****

Southwest University of Science and Technology

本科毕业设计（论文）

题目名称：邮件收发软件的开发与实现

学院名称

专业名称

学生姓名

学 号

指导教师

二〇二二年六月

**西南科技大学**

**本科毕业设计（论文）学术诚信声明**

本人郑重声明：所呈交的毕业设计（论文），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日期： 年 月 日

**西南科技大学**

**本科毕业设计（论文）版权使用授权书**

本毕业设计（论文）作者同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权西南科技大学可以将本毕业设计（论文）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本毕业设计（论文）。

**保密**□，在 年解密后适用本授权书。

本论文属于

**不保密**□。

（请在以上方框内打“**√**”）

作者签名： 指导教师签名：

日期： 年 月 日 日期： 年 月 日

基于JavaMail的邮件收发系统

摘 要:电子邮件在当今社会中扮演了一个很重要的角色。越来越多的人在使用它。而且用它的人数势必会继续增加。本文介绍了Javamail邮件收发系统的开发背景，对国内外现有的多种成熟的电子邮件系统进行分析和比较，总结出它们的优缺点，对Javamail技术进行深入研究，提出并设计实现了基于Javamail的邮件收发系统。本系统利用SMTP协议和POP协议从底层进行开发，利用JavaMail API为收发邮件提供与协议无关的访问。SMTP（简单邮件传输协议）是专门用来发送邮件的。POP（邮局协议）是专门用于接收邮件的。JavaMail API是一种可选的、能用于读取、编写和发送电子消息的标准扩展包。

本文主要可分为四个部分。第一部分介绍了Javamail邮件系统的意义和现状。第二部分具体介绍了几种相关协议和Javamail API。第三部分主要介绍了系统的开发工具JAVA语言、Eclipse及其安装过程与配置方法。第四部分详细阐述了Javamail邮件收发系统的实现过程。

关键词：SMTP POP JAVAMAIL 邮件收发

**MAIL SYSTEM BASED ON JAVAMAIL**

**ABSTRACT:**E-Mail play a very important role in modern times. More and more people are using it, and the number of it will larger and larger. This paper introduced the javamail system's development background, analyzed and compared some existing maturity Email system at home and abroad , summarized their advantages and disadvantages, studied deeply with the javamail technology , proposed and designed out the mail system based on javamail. This system used POP protocol and SMTP protocol to develop from the bottom, used JavaMail API to provide the visit which has nothing to do with the agreement for send and receive email. SMTP (Simple Mail Transfer Protocol) is designed to send a message. POP (Post Office Protocol) is designed to receive mail. JavaMail API is an standard expansion package which is optional, can be used to read, write and send electronic messages.

This paper can be divided into four parts. The first part introduced javamail email system's significance and status. The second part introduced several related agreements and JavaMail API. The third part mainly introduced system's development tools JAVA language , Eclipse and their installation process and configuration method. The fourth part detailed javamail email system's realization process.

**Keywords：**SMTP POP JAVAMAIL Email to send and receive

第一章 引言

本文是一项基于JavaMail的邮件系统的综合性研究，使用Sun公司提供的JavaMail API实现电子邮件的发送、电子邮件的接收等等功能。

1.1 研究背景

电子邮件(简称E-mai1)又称电子信箱、电子邮政，它是—种用电子手段提供信息交换的通信方式。它是全球多种网络上使用最普遍的一项服务。这种非交互式的通信,加速了信息的交流及数据传送,它是—个简易、快速的方法。通过连接全世界的Internet,实现各类信号的传送、接收、存贮等处理,将邮件送到世界的各个角落。到目前为止,可以说电子邮件是Internet资源使用最多的一种服务，E-mai1不只局限于信件的传递,还可用来传递文件、声音及图形、图像等不同类型的信息。

电子邮件不是一种“终端到终端”的服务,是被称为“存贮转发式”服务。这正是电子信箱系统的核心，利用存贮转发可进行非实时通信，属异步通信方式[1]。即信件发送者可随时随地发送邮件，不要求接收者同时在场，即使对方现在不在，仍可将邮件立刻送到对方的信箱内,且存储在对方的电子邮箱中。接收者可在他认为方便的时候读取信件,不受时空限制。在这里,“发送”邮件意味着将邮件放到收件人的信箱中,而“接收”邮件则意味着从自己的信箱中读取信件,信箱实际上是由文件管理系统支持的—个实体。因为电子邮件是通过邮件服务器(mai1 server)来传递档的。通常mail server是执行多任务操作系统UNIX的计算机，它提供24小时的电子邮件服务,用户只要向 mail server管理人员申请—个信箱账号,就可使用这项快速的邮件服务。

电子邮件的工作原理：

1.电子邮件系统是一种新型的信息系统,是通信技术和计算机技术结合的产物。

电子邮件的传输是通过电子邮件简单传输协议(Simple Mail Transfer Protocol,简称SMTP)这一系统软件来完成的,它是Internet下的一种电子邮件通信协议。

2.电子邮件的基本原理,是在通信网上设立“电子信箱系统”，它实际上是一个计算机系统。系统的硬件是一个高性能、大容量的计算机。硬盘作为信箱的存储介质,在硬盘上为用户分一定的存储空间作为用户的“信箱”，每位用户都有属于自己的—个电子信箱。并确定—个用户名和用户可以自己随意修改的口令。存储空间包含存放所收信件、编辑信件以及信件存盘三部分空间,用户使用口令开启自己的信箱,并进行发信、读信、编辑、转发、存档等各种操作，系统功能主要由软件实现。

3.电子邮件的通信是在信箱之间进行的。用户首先开启自己的信箱，然后通过键入命令的方式将需要发送的邮件发到对方的信箱中。邮件在信箱之间进行传递和交换,也可以与另—个邮件系统进行传递和交换。收方在取信时，使用特定账号从信箱提取。

1.2 研究意义

随着社会的发展，科技的进步，作为信息载体的计算机日益显露出其举足轻重的地位。当今社

会已步入了信息社会，知识经济将成为新世纪的主导产业。人们传统的联系方式也在不断的被改变。伴随着Internet的发展，电子邮件以其使用简易、投递迅速、收费低廉，易于保存、全球畅通无阻等等优点被广泛地应用，它使人们的交流方式得到了极大的改变。电子邮件已是人们沟通、交流的重要工具。

由于E-mail是当今人们交流的重要工具之一，因而，对E-mail系统的研究也逐渐变的热起来。基于JavaMail邮件系统具有快捷、方便的优点，吸引了众多的网络用户，使其成为E-mail研究方向的热点之一。

最初人们在使用E-mail服务时，必须使用邮件客服端，如:Outlook、Foxmail等软件，去接收邮件服务器上的邮件。由于操作系统的不同，往往需要不同的邮件接收客户端，给人们接收邮件带来很大的不便。但是基于JavaMail的邮件系统只要一个网络浏览器就可以实现E-mail的日常收发功能，而网络浏览器是任何操作系统中都自带的软件，用户在世界任何一个角落任何一台计算机都可以进行E-mail的收发工作。

此外，由于最初用户接收邮件的设置信息均保存在客服端，当用户重新安装操作系统或者更换计算机还要重新设置，而基于JavaMail的邮件收发系统的设置保存在邮件服务器上，不论你更换电脑还是系统，收发邮件的方式都是按照以前的设置进行的，使邮件的收发更合你的口味。

基于JavaMail的邮件系统让你不在为机器配置担忧，而且还为用户的邮件安全分担忧愁。

由于最初需要使用客服端软件收发软件，硬件配置低的计算机就显得“爱莫能助”了。虽然计算机的发展使得硬件问题不再是个问题，但是不强调硬件的配置也是基于JavaMail邮件系统的一个优点之一。此外，由于客户端软件维护和升级困难，网络浏览器的更新速度要远远大于邮件客户端软件的更新速度，这就给用户一个更安全的环境。

日前，基于JavaMail邮件系统的发展越来越成熟。多线程的设计方法，大大的提高了系统的通信效率和成本；采用MVC设计模式的邮件系统还大大的减少了系统的维护量[19]。

当然，基于JavaMail的邮件系统缺乏对动态页面的支持能力，没有集成有效地数据库处理功能，也存在系统扩展性差、数据处理速度稍慢等缺点。但是，这些问题已经有了很好的起步，将不再是大的问题。

电子邮件系统经过几十年的发展，已经形成了完善的技术体系。邮件服务器系统在保留了电子邮件系统最初的收发邮件、邮件存储等基本功能的同时，融入了最新的计算机与网络技术，使电子邮件系统有了全新的改观 。邮件技术的发展趋势如下：

1.Web邮件技术

随着中国Internet应用的日益普及和逐步深入，这些数字仍在不断增长。但是对于电子邮件系统来说，单纯使用邮件客户端程序进行邮件的收发已经不能满足用户移动办公的需要。Web邮件技术的出现，彻底解决了用户办公的需要，使得邮件用户可以在任何地方使用浏览器登录邮件服务器收发邮件，而不用配置邮件客户端程序。

2.多域邮件服务

所谓多域邮件服务，即是一台物理服务器为多个独立注册Internet域名的企业或单位提供电子邮件的服务，在逻辑上，这些企业和单位拥有自己独立的邮件服务器，也可以称为虚拟邮件服务器技术。对于ISP提供商和企业集团公司来说，多域邮件服务器的支持能力是选择邮件服务器的一个重要考虑因素。它可以方便地扩展其横向邮件服务能力。

3. Linux邮件服务器

Linux操作系统作为目前应用最为广泛的开源操作系统，具有性能稳定、可靠性高和价格低廉的特点。使用Linux作为邮件服务器，主要是可以与Sendmail、MySQL等开源软件共同使用，在满足用户需求的基础上降低了系统价格。

4.安全防护

现在的邮件服务器在安全防护技术上有了较大的提高，包括数据身份认证、传输加密、垃圾邮件过滤、邮件病毒过滤、安全审计等的多项安全技术在邮件服务器中都得到了很好的应用。

5.多语言

目前光中文就有若干字符集，如GB-18030、GB-2312、Big5等等，虽然我们可以统一标准，但是，在实际的过程中，我们不可能统一所有的邮件客户端，因此我们只能要求邮件服务器支持多语言的环境，使“我们的沟通无障碍”。

6.远程监控和性能调整

由于目前许多邮件服务器处于电信托管等方式，不可能经常进行本地操作，因此目前邮件服务器均提供了远程邮件监控的功能。我们可以通过Web方式，监控邮件服务器的工作状态，包括在线用户数、邮件处理数量和速度、存储空间使用率等，并且可以随时对出现的发信高峰和网络攻击进行远程处理[2]。

总之，计算机的广泛应用，Internet的不断发展，E-mail在日常通讯中的地位越来越重要。虽然传统的C/S模式的邮件系统仍占有重要地位，但基于JavaMail的邮件收发系统会以其快捷、方便的等等优点为越来越多的人所接收，广泛的应用到各个领域，成为一名后起之秀。

1.3 研究现状

E-mail是信息化时代最重要的联系工具之一，在日常的工作学习中具有非常重要的作用。对E-mail的研究也层出不穷，基于JavaMail邮件系统的设计与实现大多数都是研究和使用Sun公司提供的JavaMail API去实现其邮件发送和接收功能。

1.3.1 国外研究现状

在国外，“邮件收发”方面的研究涉及到更多的技术和应用场景，例如人工智能、区块链、大数据等，以实现更加智能化、安全化和高效化的邮件收发服务。

例如，Google公司在其邮件服务Gmail中使用了人工智能技术，通过机器学习等算法对用户的邮件进行智能分类和管理，提供了更加智能化的邮件收发服务。同时，Gmail还使用了DMARC、SPF和DKIM等安全协议，保障了邮件传输的安全性。

此外，英国的研究人员也在其论文中提出了一种基于区块链的安全邮件系统，通过将邮件内容和元数据加密存储在区块链上，确保邮件的机密性和不可篡改性。该系统使用了PGP和S/MIME等安全协议，提供了更加安全的邮件传输和存储服务。

另外，美国的研究人员在其论文中提出了一种基于大数据和机器学习的邮件反垃圾方案，通过对邮件元数据和内容进行分析和分类，减少垃圾邮件对用户的干扰和影响。

综上所述，国外的研究人员在“邮件收发”方面，主要关注于技术创新和应用场景拓展，使用人工智能、区块链、大数据等技术，提供更加智能化、安全化和高效化的邮件收发服务，以满足用户的不同需求。

1.3.2 国内研究现状

在“邮件收发”方面，国内研究人员主要关注于邮件客户端的开发和实现，以提供更加便捷、安全和高效的邮件收发服务。

例如，中国科学院计算技术研究所的研究人员在其论文中提出了一种基于Web的邮件客户端的设计和实现方法。该方法使用HTML、CSS和JavaScript等Web技术，将邮件客户端实现为一个网页应用程序，用户可以通过浏览器访问该网页，完成邮件收发的操作。该方法相较于传统的邮件客户端具有更加灵活的操作方式，可以随时随地进行邮件收发操作，具有较好的可移植性和跨平台性。

此外，南京大学的研究人员也在其论文中提出了一种基于移动终端的邮件客户端设计方案，该方案主要针对移动设备用户的邮件收发需求，提供了一些特色功能，例如支持离线邮件查看和回复、自动邮件分类等。该方案还使用了SSL和TLS等安全协议，加强了邮件传输的安全性。

综上所述，国内的研究人员在“邮件收发”方面，主要关注于邮件客户端的开发和实现，使用Web技术和移动技术等，提供更加便捷、安全和高效的邮件收发服务，以满足用户的不同需求。

1.4 研究内容及目标

本研究的主要内容和目标是设计和实现一个基于JavaMail的电子邮件系统，重点关注其可扩展性、安全性、和稳定性。具体包含以下几个方面：

1.4.1 研究内容

邮件发送和接收功能的实现：研究和使用Sun公司提供的JavaMail API，实现邮件发送和接收功能，包括文本邮件和附件邮件的发送和接收。

邮件存储和检索：研究如何有效存储和检索邮件，以提高邮件系统的效率和用户体验。

邮件系统的安全性：研究如何提高邮件系统的安全性，防止邮件被恶意攻击和滥用。

邮件系统的扩展性：研究如何提高邮件系统的扩展性，使邮件系统可以适应更大的用户群和更复杂的应用环境。

邮件系统的稳定性：研究如何提高邮件系统的稳定性，使邮件系统在各种网络环境下都能稳定运行。

1.4.2 研究目标

实现基于JavaMail的电子邮件系统：利用JavaMail API，完成电子邮件系统的设计和实现，满足用户的基本邮件发送和接收需求。

提高邮件系统的安全性：设计和实现有效的安全机制，防止邮件系统被恶意攻击和滥用。

提高邮件系统的扩展性：设计和实现有效的扩展机制，使邮件系统可以适应更大的用户群和更复杂的应用环境。

提高邮件系统的稳定性：通过优化邮件存储和处理机制，提高邮件系统的稳定性，使邮件系统在各种网络环境下都能稳定运行。

提供优秀的用户体验：通过优化邮件检索和显示机制，提供优秀的用户体验，使用户可以方便快捷地使用邮件系统。

1.5 论文组织结构（后面回来写）

1.6 本章小节

本章主要介绍了本研究的背景和目标，以及电子邮件系统的研究现状。

首先，我们详细阐述了电子邮件在信息化时代中的重要性，以及基于JavaMail的电子邮件系统设计与实现的必要性。我们讨论了现有电子邮件系统的一些问题，例如安全性、稳定性和可扩展性等。

接下来，我们对国内外的电子邮件系统研究现状进行了详细的分析。在国际范围内，我们主要研究了Unix平台下的Sendmail和Windows平台的Microsoft Exchange。在国内，我们主要研究了电子邮件系统的稳定性、扩展性和安全性等问题。

最后，我们明确了本研究的内容和目标，包括实现基于JavaMail的电子邮件系统，提高邮件系统的安全性、扩展性和稳定性，以及提供优秀的用户体验。

第2章 相关理论基础和技术简介

2.1 相关邮件协议

邮件的传送要依赖于具体的传输协议，主要有：SMTP、POP3、MIME。

（1）SMTP：简单邮件传输协议，它是一组用于由源地址到目的地传送邮件的规则，或者说是由它来控制信件传输的一种中转方式[13]。SMTP协议属于TCP/IP协议族，它帮助每台计算机在发送或中转邮件时找到下一个目的地。通过SMTP协议所指定的服务器，我们就可以把Email寄到收信人的服务器上了，整个过程只需要几分钟。SMTP的开发最初是为了在封闭的网络中传送相对来说不太重要的简短信件，而不是为了在互联网中传送重要而敏感的信息，因此最初通过SMTP传输信件时 ，安全性不高。

（2）POP：邮局协议[5]，当前的版本为3，也称作POP3。POP是一种允许用户从邮件服务器收发邮件的协议，它有两种版本，即POP2和POP3，都具有简单的电子邮件存储转发功能。POP2和POP3本质上类似，都属于离线式工作协议，但是由于使用了不通的协议端口，两者并不兼容。与SMTP协议相结合，POP3是目前最常用的电子邮件服务协议。在离线工作方式下，用户收发邮件时，首先通过POP3客户程序登录到支持POP3协议的邮件服务器，然后发送邮件及附件；接着，邮件服务器为该用户收存的邮件传送给POP3客户程序，并将这些邮件从服务器上删除；最后，邮件服务器将用户提交的发送邮件，转发到运行SMTP协议的计算机中，通过它实现邮件的最终发送。在为用户从邮件服务器收取邮件时，POP3是以该用户当前存储在服务器上全部邮件为对象进行操作的，并一次性将它们下载到用户端计算机中。一旦客户的邮件下载完毕，邮件服务器对这些邮件的暂存托管即告完毕。使用POP3，用户不能对他们贮存在邮件服务器上的邮件进行部分传输，离线工作方式适合那些从固定计算机上收发邮件的用户使用。

（3）MIME[13]: 多用途网际邮件扩充协议，是最初的互联网电子邮件协议的一个扩展，说明了如何安排消息格式使消息在不同的邮件系统内进行交换。 MIME 的格式灵活，使人们能使用这一协议在互联网上交换不同类型的数据文件：音频，视频，图像，应用软件和其他类型的文件，也包括在最初的协议—简单邮件传送协议(SMTP)中的ASCII文本。

（4）IMAP：IMAP使用在接收信息的高级协议，目前版本为第4版，所以也被称为IMAP4。需要注意的是在使用IMAP时，邮件服务器必须支持该协议[21]。从这个方面讲，我们并不能完全使用IMAP来替代POP，不能期待IMAP在任何地方都被支持。假如邮件服务器支持IMAP，那么我们的邮件程序将能够具有以下被IMAP所支持的特性[6]：每个用户在服务器上可具有多个目录，这些目录能在多个用户之间共享。其与POP相比高级之处显而易见，但是在尝试采取IMAP时，我们认识到它并不是十分完美的。由于IMAP需要从其它服务器上接收新信息，将这些信息递送给用户，维护每个用户的多个目录，这都为邮件服务器带来了高负载。并且IMAP与POP的一个不同之处是POP用户在接收邮件时将从邮件服务器上下载邮件，而IMAP允许用户直接访问邮件目录，所以在邮件服务器进行备份作业时，由于每个长期使用此邮件系统的用户所用的邮件目录会占有很大的空间，这将直接导致邮件服务器上磁盘空间暴涨。

（5）NNTP和其它的第三方协议：正因为JavaMail API在设计时考虑到与第三方协议实现提供商之间的分离，故我们可以很容易的添加一些第三方协议[4]。SUN维护着一个第三方协议实现提供商的列表：<http://java.sun.com/products/javamail/Third_Party.html>，通过此列表我们可以找到所需要的而又不被SUN提供支持的第三方协议：比如NNTP这个新闻组协议和S/MIME这个安全的MIME协议。

2.2 JavaMail API

2.2.1 JavaMail API简介

JavaMail API是SUN为Java开发者提供对电子邮件处理的延伸，能用于读取、编写和发送电子消息的包（标准扩展）。为管理电子邮件提供了统一的应用编程接口，它可以处理各种e-mail格式（包括IMAP、POP、SMTP以及MIME）和其他Internet相关的讯息通信协定。

用户可使用这种包创建邮件用户代理（Mail User Agent，MUA）类型的程序，它类似于Foxmail、Eudora以及Miceosoft Outlook这些邮件程序。其主要目的不是像发送邮件或其他邮件传输代理（Mail Transfer Agent，MTA）类型的程序那样用于传输、发送和转发消息[7]。换句话说，用户可以与MUA类型的程序交互，以阅读和撰写电子邮件。MUA依靠MTA处理实际的发送任务。

2.2.2 JavaMail API核心类

（1）Session：Session类定义了基本的邮件会话。就像Http会话那样，我们进行收发邮件的工作都是基于这个会话的。Session对象利用了java.util.Properties对象获得了邮件服务器、用户名、密码信息和整个应用程序都要使用到的共享信息。

Session类的构造方法是私有的，所以我们可以使用Session类提供的getDefaultInstance()这个静态工厂方法获得一个默认的Session对象：

Properties props = new Properties();

// fill props with any information

Session session = Session.getDefaultInstance(props, null);

或者使用getInstance()这个静态工厂方法[8]获得自定义的Session:

Properties props = new Properties();

// fill props with any information

Session session = Session.getInstance(props, null);

从上面的两个例子中不难发现，getDefaultInstance()和getInstance()方法的第二个参数都是null，这是因为在上面的例子中并没有使用到邮件授权。在对mail server进行访问的过程中使用共享的Session是足够的，即使是工作在多个用户邮箱的模式下也不例外。

（2）Message ：一旦获得Session对象，就可以继续创建要发送的信息，在这里SUN提供了Message类型来帮助开发者完成这项工作。由于Message是一个抽象类，大多数情况下，我们使用javax.mail.internet.MimeMessage这个子类，该类是使用MIME类型和头的电子邮件信息。

信息头只能使用US-ASCII字符，而非ASCII字符将通过编码转换为ASCII的方式使用。为了建立一个MimeMessage对象，我们必须将Session对象作为MimeMessage构造方法的参数传入：

MimeMessage message = new MimeMessage(session);

注意：对于MimeMessage类来讲存在着多种构造方法，比如使用输入流作为参数的构造方法。

在建立了MimeMessage对象后，我们需要设置它的各个part，对于MimeMessage类来说，这些part就是MimePart接口。最基本的信息内容设置方法就是通过表示信息内容和MIME类型的参数调用setContent()方法：message.setContent("Hello", "text/plain")[9];然而，如果我们所使用的MimeMessage中信息内容是文本的话，我们便可以直接使用setText()方法来方便的设置文本内容。message.setText("Hello");前面所讲的两种方法，对于文本信息，后者更为合适。而对于其它的一些信息类型，比如HTML信息，则要使用前者。使用setSubject()方法对邮件设置邮件主题：message.setSubject("First");

（3）Address：到这里，我们已经建立了Session和Message，像Message一样，Address类也是一个抽象类，所以我们将使用javax.mail.internet.InternetAddress这个子类。

通过传入代表邮件地址的字符串，我们可以建立一个邮件地址类：

Address address = new InternetAddress("president@whitehouse.gov");

如果要在邮件地址后面增加名字的话，可以通过传递两个参数：代表邮件地址和名字的字符串来建立一个具有邮件地址和名字的邮件地址类：

Address address = new InternetAddress("president@whitehouse.gov", "George Bush");

本文在这里所讲的邮件地址类是为了设置邮件信息的发信人和收信人而准备的，在建立了邮件地址类后，我们通过message的setFrom()和setReplyTo()两种方法设置邮件的发信人：

message.setFrom(address);

message.setReplyTo(address);

若在邮件中存在多个发信人地址，我们可用addForm()方法增加发信人：

Address address[] = ...;

message.addFrom(address);

为了设置收信人，我们使用addRecipient()方法增加收信人，此方法需要使用Message.RecipientType的常量来区分收信人的类型：

message.addRecipient(type, address)

下面是Message.RecipientType的三个常量:

Message.RecipientType.TO

Message.RecipientType.CC

Message.RecipientType.BCC

因此，如果我们要发送邮件给总统，并发用一个副本给第一夫人的话，下面的方法[10]将被用到：

Address toAddress = new InternetAddress("vice.president@whitehouse.gov");

Address ccAddress = new InternetAddress("first.lady@whitehouse.gov");

message.addRecipient(Message.RecipientType.TO, toAddress);

message.addRecipient(Message.RecipientType.CC, ccAddress);

JavaMail API并没有提供检查邮件地址有效性的机制。当然我们可以自己完成这个功能：验证邮件地址的字符是否按照RFC822规定的格式书写或者通过DNS服务器上的MX记录验证等。

（4）Authenticator： JavaMail API也可以利用Authenticator通过用户名和密码访问受保护的资源。对于JavaMail API来说，这些资源就是邮件服务器。JavaMail Authenticator在javax.mail包中，要使用Authenticator，先创建一个抽象类的子类，并从getPasswordAuthentication()方法中返回PassaordAuthentication实例。创建完成后，必须向Session注册Authenticator。然后，在需要认证的时候，就会通知Authenticator。

（5）Transport：像java.net类那样，JavaMail API通过使用授权者类（Authenticator）以用户名、密码的方式访问那些受到保护的资源，在这里“资源”就是指邮件服务器。在javax.mail包中可以找到这个JavaMail的授权者类（Authenticator）。

在使用Authenticator这个抽象类时，我们必须采用继承该抽象类的方式，并且该继承类必须具有返回PasswordAuthentication对象（用于存储认证时要用到的用户名、密码）getPasswordAuthentication()方法。并且要在Session中进行注册，使Session能够了解在认证时该使用哪个类。

下面代码片断中的MyAuthenticator就是一个Authenticator的子类。

Properties props = new Properties();

// fill props with any information

Authenticator auth = new MyAuthenticator();

Session session = Session.getDefaultInstance(props, auth);

（6）Folder和Store ：Store类实现特定邮件协议上的读、写、监视、查找等操作。通过javax.mail.Store类可以访问javax.mail.Folder类。

Folder类用于分级组织邮件，并提供javax.mail.Message格式访问e-mail的能力。

2.2.3 JavaMail API的应用

（1）阅读电子邮件

对于阅读邮件来说，首先要获取一个会话，然后获取并连接到一个相应的用于您的收件箱的存储上，接着打开相应的文件夹，再获取消息。同时，不要忘记了操作完成后关闭连接。

步骤如下：

* + - 1. 创建一个Properties 对象，Session 对象通过Properties 对象来接受选项，在这种情况下，没有选项，但仍然需要Properties 对象。
      2. 创建Session 对象，并不直接对Session 实例化，而是通过调用getDefaultInstance
      3. 使用Session.getStore 来获取服务器的POP3存取，也可以指定IMAP。
      4. 使用store.connect 来创建一个从store 对象到实际服务器上的活连接。
      5. 用store.getFolder 找到INBOX（收件箱）文件夹（POP3 存储里唯一的文件夹），对IMAP 服务器而言，可以访问不同的文件夹。用folder.open 以只读方式打开文件夹。

（2）发送电子邮件

发送电子邮件消息涉及到获取会话、创建和填充消息并发送消息这些操作。在获取Session 时，通过为要传递的Properties 对象设置mail.smtp.host属性来指定所用的SMTP 服务器。

（3）转发电子邮件

转发消息涉及的内容要稍微多一点，没有一个专门用于转发消息的方法，可以通过处理组成消息的各个部分来创建要转发的消息。一条邮件消息可由多个部分组成，每一部分是一个BodyPart（报文部分），或更特殊一点，在操作MIME 消息时则是MimeBodyPart [11]。不同的报文部分组合到一个称为Multipart 的容器中，或者更特殊一点，是一个MimeMultipart 容器。要转发消息，用户需要创建一个用于消息文本的部分，和用于要转发的消息的第二个部分，并将这两个部分组合成一个Multipart（多个部分）。然后， 可以把这个Multipart添加到一个合适的注明地址的消息中并发送它。

这就是转发消息的过程。要把一条消息的内容复制给另一条消息，只需通过它的ataHandler 类复制即可，它是出自于JavaBeans Activation Framework的一个类。

上面着重分析和探讨基于JavaMail API 的邮件服务系统的设计和实现的相关技术。为系统实际开发过程中充分利用Java 编程语言的开放性、平台无关性等优点，极大地提高了系统的效率和性能。

2.3 本章小节

在本章中，我们详细介绍了JavaMail API的基本概念和应用，通过深入的分析和探讨了基于JavaMail API的邮件服务系统设计和实现的相关技术。

首先，我们介绍了JavaMail API的基本结构和主要组件，包括Session、Message、Address、Authenticator、Transport、Folder和Store等。我们解释了如何通过这些组件创建、发送和接收电子邮件。

然后，我们详细讨论了JavaMail API的应用，包括阅读电子邮件、发送电子邮件以及转发电子邮件的方法。我们展示了如何创建一个Properties对象，获取并建立与邮件服务器的连接，打开邮件文件夹，获取和处理消息。我们还介绍了如何使用SMTP服务器发送电子邮件，以及如何通过处理组成消息的各个部分来转发电子邮件。

最后，我们着重分析了基于JavaMail API的邮件服务系统的设计和实现。我们讨论了如何充分利用Java编程语言的开放性和平台无关性来提高系统的效率和性能。

第3章 JavaMail系统分析

3.1 系统需求分析

在本章节中，我们将对邮件收发软件的需求进行详细的分析，包括功能需求、非功能性需求、可靠性需求和用户体验需求。

3.1.1 功能性需求分析

3.1.1 功能性需求分析

邮件收发软件的主要功能包括邮件的发送、接收和管理。用户需要能够方便地发送邮件，并能够在软件中接收、查看和管理已发送和已接收的邮件。此外，用户还需要能够对邮件进行删除、下载附件等操作，以提高邮件处理的效率。具体如图3.1所示[17]。

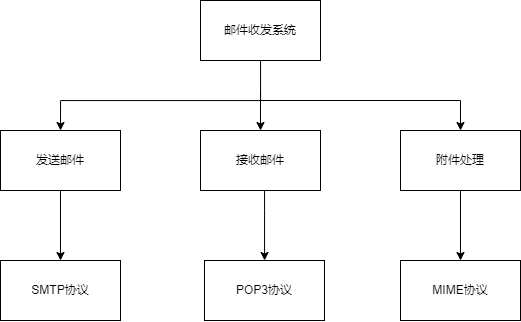


图3.1 系统功能模块组织图

3.1.2 非功能性需求分析

非功能性需求是指不直接与系统的具体功能相关，但对系统的运行和使用至关重要的需求。在邮件收发软件中，非功能性需求主要包括安全性、可靠性和用户体验等方面。

首先，我们分析安全性需求。为了确保我们设计的邮件收发软件在使用时更加安全，我们需要满足以下要求：

安全的邮件传输协议[18]：软件应该支持SMTPS、POP3S等安全的邮件传输协议，以保证邮件在传输过程中的安全性。

数据加密：软件需要支持SSL和TLS协议，这样可以加密邮件传输过程中的数据，以避免数据泄露或者被恶意拦截，防止数据被窃取或篡改。同时，软件应该支持对邮件内容进行加密，以防止邮件内容被窃取或篡改。

身份验证机制：软件应该支持邮件收发过程中的身份验证机制，以确保只有合法用户可以收发邮件，防止邮件被恶意使用。

用户隐私保护：软件应该提供用户隐私保护功能，例如用户可以选择不公开自己的邮件地址，以保护用户的隐私安全。

其次，我们分析可靠性需求。邮件是用户进行通信和交流的重要工具，因此邮件收发软件需要保障其可靠性，避免邮件在传输过程中丢失或丢失部分内容。因此，软件需要使用可靠的邮件传输协议和机制，例如SMTP的重传机制和POP3的删除标记机制，以确保邮件能够可靠地传输和接收[19]。

最后，我们分析用户体验需求。用户体验是邮件收发软件的重要方面。软件需要提供用户友好的界面和操作体验，使用户能够轻松地发送、接收和管理邮件，同时也需要保障软件的稳定性和性能，以提高用户的满意度。

3.2 可行性研究

在启动邮件收发软件的开发项目之前，我们必须对其可行性进行全面的评估。这个过程包括对技术实现的可能性、法律合规性等因素的考量，确保我们的项目在后续的开发阶段不会遇到不可预见的难题，为项目的成功提供一个坚实的基础。

3.2.1 技术可行性

邮件收发软件的实现核心依托于SMTP、POP3、MIME等关键通信协议和技术。因此，这个毕业设计项目的成功，需要我具有足够的理论知识和实际操作能力来处理这些协议。

通过自我评估和导师的指导，我发现我已经掌握了这些协议的基本工作原理，而且我能够成功地将这些协议应用到我的毕业设计项目中。此外，我也已经具备了解决可能出现的技术问题和挑战的能力，这对于保证软件的稳定性和性能至关重要。

总的来说，通过对自身的技术理解和实践能力的全面评估，我相信我可以成功地完成这个毕业设计项目，实现邮件收发软件的核心功能，证明这个项目在技术上是完全可行的。

3.2.2 法律可行性

在开发邮件收发软件的过程中，我们需要遵守所有相关的法律法规，包括但不限于网络安全法、个人信息保护法等。这些法律法规为我们的软件设定了一系列的法律框架，我们必须在这个框架内进行开发和实现。

因此，我们需要深入了解这些法律法规，理解它们对我们的软件可能产生的影响，并在软件的设计和实现过程中尽可能地遵守这些规定。这不仅可以确保我们的软件在法律上的合规性，也可以帮助我们避免可能的法律风险。

综上，通过对相关法律法规和政策要求的深入理解和分析，我们可以确定邮件收发软件在法律上的可行性。

3.3 本章小结

本章主要介绍了邮件收发系统的系统分析。首先进行了需求分析，包括功能性需求分析和非功能性需求分析并在此基础上进行了可行性分析，包括技术可行性和法律可行性。

在功能需求方面，系统需要支持邮件的发送和接收，同时需要支持邮件附件的添加和处理，并支持多种邮件格式以及安全传输协议。在非功能性需求分析方面，系统需要保障邮件的机密性和完整性，并保护用户的隐私。在可靠性需求方面，系统需要保证邮件的可靠性和稳定性，避免邮件丢失和延迟等问题。在用户体验需求方面，系统需要提供简洁、直观、易用的界面，方便用户操作和管理邮件。

在可行性分析方面，本章主要从技术可行性和法律可行性两个方面进行了探讨。在技术可行性方面，系统基于JavaMail，并使用SMTP、POP3和MIME等通信协议，具有良好的可扩展性和兼容性，可以满足系统的需求。在法律可行性方面，系统需要遵循相关的法律法规和规范，保护用户的合法权益，避免违反相关法规而导致的风险和损失[21]。

第4章 JavaMail系统总体设计

4.1 系统结构设计

本系统采用三层架构模式具体架构如下：

表示层：提供用户界面，实现与用户的交互。采用Java Swing技术实现，包括主窗口、邮件列表窗口、邮件编辑窗口等。

业务逻辑层：处理业务逻辑和系统控制，负责数据处理和状态管理。主要包括邮件发送、接收、解析、显示等功能，采用JavaMail、SMTP、POP3和MIME协议实现。

数据访问层：负责数据的存储和访问，采用文件系统或数据库存储邮件数据，提供数据操作的接口。

系统结构设计图如图4.1所示[22]。

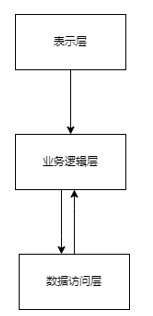


图4.1 系统结构设计框图

4.2 功能模块设计

本系统主要包括以下功能模块[23]：

邮件发送模块，邮件接收模块，邮件解析模块，邮件显示模块，邮件设置模块。

以上功能模块通过MVC（Model-View-Controller）设计模式实现，其中Model层负责数据的存储和处理，View层负责用户界面的显示和交互，Controller层负责业务逻辑的处理和控制。通过分层设计和模块化开发，实现了系统的可扩展性和可维护性。

总体结构图如下：

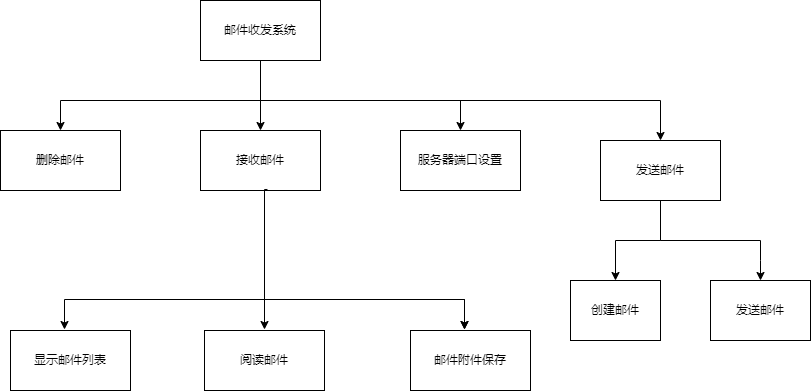


图4.2 功能框图

通过以上设计，实现了邮件客户端程序的基本功能，同时保障了邮件在传输过程中的安全性，提高了用户的使用体验。未来可以在此基础上继续完善功能，提升软件的性能和稳定性。

4.2.1 发送邮件模块设计

发送邮件模块是整个系统的核心功能之一，需要支持多种邮件格式（如纯文本邮件、HTML邮件等）和邮件安全传输协议（如SMTPS、SSL、TLS等），并能够添加和处理邮件附件。该模块主要包括以下功能模块[24]：

邮件发送器：负责发送邮件，包括设置邮件内容、添加附件、设置收件人、主题、发送时间等信息。

SMTP客户端：负责与SMTP服务器建立连接，并通过SMTP协议发送邮件消息。

MIME编码器：负责将邮件内容转换为HTML或纯文本格式，以及对附件进行编码。

邮件安全传输器：负责保障邮件的安全传输，如使用SMTPS、SSL、TLS等协议。

4.2.2 接收邮件模块设计

接收邮件模块是系统的另一个核心功能，需要支持多种邮件格式（如纯文本邮件、HTML邮件等）和邮件安全传输协议（如POP3S、SSL、TLS等），并能够处理邮件附件。该模块主要包括以下功能模块：

邮件接收器：负责接收邮件，包括设置邮件内容、保存附件、解析邮件头部信息。

POP3客户端：负责与POP3服务器建立连接，并通过POP3协议接收邮件消息。

邮件安全传输器：负责保障邮件的安全传输，如使用POP3S、SSL、TLS等协议。

以上功能模块都需要与JavaMail API进行交互，利用JavaMail API提供的邮件收发相关类实现具体功能。同时，还需要使用Java的多线程技术，使得邮件发送和接收功能可以在后台异步运行，提高系统的响应速度和性能[25]。

4.2.3 附件处理模块设计

附件处理模块是邮件收发系统中的一个重要模块，负责对邮件附件的添加、处理和下载。系统需要支持各种类型的附件，包括文本文件、图片、视频等。在实现该模块时，需要考虑到安全性、可靠性和用户体验等方面[26]。

该模块的具体设计如下：

添加附件功能：用户可以通过系统界面选择本地文件并添加到待发送邮件中，也可以通过拖拽操作将文件添加到待发送邮件中。

附件处理功能：当邮件被接收后，系统需要对附件进行解析和处理。对于文本文件和图片等常见类型的附件，系统需要提供在线预览和下载功能；对于视频等大型文件，系统需要提供下载链接或者引导用户通过云存储等方式进行下载。

安全性保障：在邮件附件的传输和处理过程中，系统需要采用加密技术和安全传输协议，确保数据的安全性和保密性。

4.2.4 界面设计

界面设计是邮件收发系统中的重要组成部分，直接影响用户的使用体验。系统需要提供清晰、简洁、易用的界面，方便用户进行邮件的发送、接收和管理。在设计界面时，需要考虑到以下因素：

功能布局：界面需要将不同的功能模块进行合理布局，使用户可以快速找到需要的功能，同时避免界面过于复杂和拥挤。

操作便捷性：系统需要提供简单易懂的操作指引，让用户能够快速上手并熟练使用系统。

界面美观度：系统需要采用统一的UI设计风格，保证界面的美观度和协调性。

用户体验：界面需要考虑到用户的使用习惯和心理需求，提供更加贴近用户需求的设计。例如，可以提供主题换肤功能、快捷键等用户体验优化功能。

在总体设计阶段，需要综合考虑系统的需求和目标，对各个功能模块进行细致的设计和规划，为后续的系统开发和实现提供有力支持。

4.3 数据库设计

数据库设计是整个系统设计的重要部分，它决定了数据的存储、查询、更新和管理如何进行。邮件收发系统需要存储用户信息、邮件信息和附件信息等数据，因此需要设计一个高效、安全、可扩展的数据库。

4.3.1 用户信息表设计

用户信息表用于存储用户的基本信息，包括用户名、密码、邮箱地址等。下表是用户信息表的设计：

表4.3.1 用户信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 说明 |
| UserID | INT | 用户ID，主键，自增长 |
| UserName | VARCHAR(50) | 用户名 |
| Password | VARCHAR(50) | 用户密码，需加密存储 |
| Email | VARCHAR(50) | 用户的邮箱地址 |

4.3.2 邮件信息表设计

邮件信息表用于存储邮件的基本信息，包括邮件的发送者、接收者、主题、发送时间等。下表是邮件信息表的设计：

表4.3.2 邮件信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 说明 |
| MailID | INT | 邮件ID，主键，自增长 |
| Sender | VARCHAR(50) | 发送者邮箱地址 |
| Receiver | VARCHAR(50) | 接收者邮箱地址 |
| Subject | VARCHAR(100) | 邮件主题 |
| SendTime | DATETIME | 邮件发送时间 |

4.3.3 附件信息表设计

附件信息表用于存储与邮件相关的附件信息，包括附件的名称、类型、大小等。下表是附件信息表的设计：

表4.3.3 附件信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 说明 |
| AttachID | INT | 附件ID，主键，自增长 |
| MailID | INT | 对应的邮件ID，外键 |
| FileName | VARCHAR(50) | 附件名称 |
| FileType | VARCHAR(20) | 附件类型 |
| FileSize | INT | 附件大小 |

4.4 本章小结

本章详细介绍了邮件收发系统的总体设计。首先，我们根据MVC模式设计了系统结构，明确了模型层、视图层和控制层的作用和职责。接着，在功能模块设计中，我们对发送邮件模块、接收邮件模块、附件处理模块进行了详细的设计和描述。发送邮件模块主要包括邮件信息的构建、邮件内容的格式化和发送邮件的实现；接收邮件模块主要包括邮件服务器的配置、连接和接收邮件的实现；附件处理模块主要负责添加和处理邮件附件。而界面设计则主要通过Java Swing实现，以提供友好的用户交互界面。

此外，我们还设计了系统所需的数据库结构，这包括用户信息表、邮件信息表和附件信息表，以便有效地存储和管理系统中的关键数据。

通过本章的设计，系统具备了完整的邮件收发功能、附件处理功能，以及有效的数据管理能力。同时，我们还提供了一个友好的用户交互界面，满足了在需求分析中提出的功能需求和用户体验需求。在后续的实现和测试阶段，我们将按照本章的设计要求进行具体的实现和测试，以确保达到系统设计目标。

第5章 JavaMail系统实现

5.1 JavaMail系统实现

5.1.1 JavaMail的实现

导入JavaMail库

JavaMail是一个独立的库，需要在项目中引入该库才能使用JavaMail API。从JavaMail官网下载JavaMail库，或者使用Maven或Gradle等构建工具进行依赖管理。

创建Session对象

Session是JavaMail API的核心对象，表示一个邮件会话，包含了与SMTP服务器的连接信息、身份验证信息等。可以使用javax.mail.Session.getDefaultInstance()方法创建默认的邮件会话，或者使用javax.mail.Session.getInstance(Properties props, Authenticator authenticator)方法创建一个指定的邮件会话。

创建Message对象

Message是JavaMail API中表示邮件的对象，可以设置邮件的发送者、接收者、主题、内容、附件等信息。可以使用javax.mail.internet.MimeMessage类创建Message对象，也可以使用javax.mail.Message子类创建不同类型的邮件对象。

使用Transport.send(Message message)方法发送邮件，该方法会自动将Message对象转换为符合SMTP协议的数据，并通过SMTP服务器发送邮件。发送邮件前需要设置SMTP服务器的信息和身份验证信息。

5.1.2 SMTP协议的实现

JavaMail提供了SMTP协议的实现方法。在JavaMail中，可以使用javax.mail包中的SMTPTransport类来实现SMTP协议。使用SMTPTransport类可以连接SMTP服务器、发送邮件和接收响应信息等操作。以下是JavaMail中实现SMTP协议的基本步骤：

创建Session对象：使用Session.getDefaultInstance()方法创建Session对象，设置SMTP服务器地址、端口号等属性。

创建Message对象：创建MimeMessage对象，设置邮件的发件人、收件人、主题、内容等信息。

创建Transport对象：使用Session对象创建SMTPTransport对象。

连接SMTP服务器：使用SMTPTransport对象的connect()方法连接SMTP服务器。

发送邮件：使用SMTPTransport对象的sendMessage()方法发送邮件。

关闭连接：使用SMTPTransport对象的close()方法关闭连接。

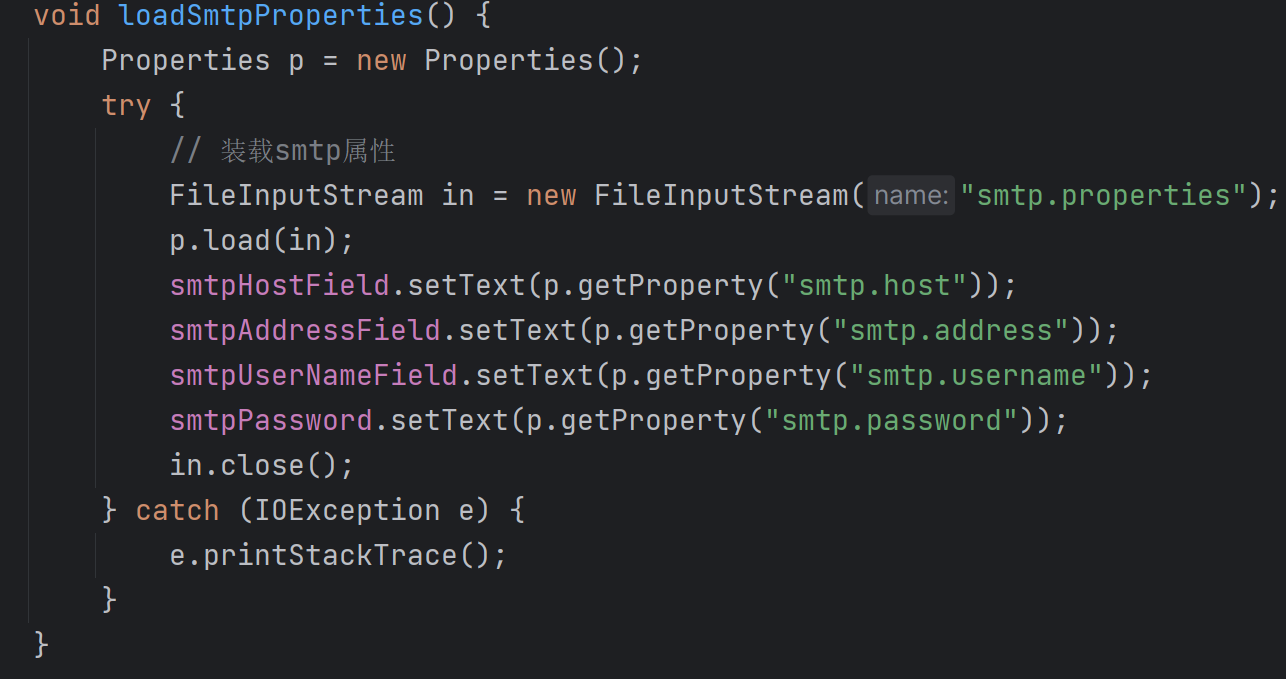


图5.1.2 SMTP加载关键代码

5.1.3 POP3协议实现

服务器端的实现

POP3服务器的实现通常由邮件服务器提供商完成。服务器需要提供基本的POP3功能，如接受来自客户端的连接请求、进行身份验证、检查并下载邮件、删除邮件等。POP3服务器通常会使用TCP协议来与客户端进行通信，并且需要一个独立的端口（默认是110）。

客户端的实现

POP3客户端的实现通常是邮件客户端程序。客户端程序需要提供基本的POP3功能，如连接服务器、进行身份验证、获取邮件、删除邮件等。JavaMail API提供了POP3客户端的实现，使得开发者可以使用Java语言来开发邮件客户端应用程序。

首先需要设置POP3服务器的地址、用户名和密码等参数。然后创建一个Properties对象来存储这些参数。接下来，通过Session.getInstance()方法创建一个会话对象，该方法需要传递Properties对象和一个Authenticator对象，用于进行身份验证。在会话对象中，使用getStore()方法获取存储对象，并调用connect()方法连接服务器。

通过存储对象获取邮件列表，并遍历每一封邮件，并打印邮件的主题、发送人、内容等信息。最后，关闭文件夹和存储对象，释放资源。



图5.1.3 POP3加载关键代码

5.1.5 邮件收发模块

邮件收发模块的实现主要分为两个部分：邮件发送和邮件接收。下面分别介绍这两部分的实现。

邮件发送

邮件发送模块的实现包括两个部分：SMTP协议的通信和邮件消息的构建。下面分别介绍这两部分的实现。

SMTP协议的通信：

使用JavaMail API实现SMTP协议的通信需要以下步骤：

创建一个Properties对象，用于设置SMTP服务器的地址和端口号、是否需要身份验证等信息；

创建一个Session对象，将Properties对象传入，用于与SMTP服务器建立连接；

创建一个MimeMessage对象，用于构建邮件消息；

使用Transport对象的send方法将MimeMessage发送给SMTP服务器。

邮件消息的构建：

具体实现步骤如下：

创建一个MimeMessage对象；

设置邮件的发件人地址、收件人地址、主题、发送时间等信息；

创建一个MimeMultipart对象，用于存储邮件的文本和附件内容；

创建一个MimeBodyPart对象，设置邮件的文本内容；

将MimeBodyPart对象添加到MimeMultipart对象中；

遍历邮件的附件列表，为每个附件创建一个MimeBodyPart对象，并将其添加到MimeMultipart对象中；

将MimeMultipart对象设置为MimeMessage的内容；

调用Transport对象的send方法将MimeMessage对象发送给SMTP服务器。

以上是JavaMail API实现邮件发送的一般步骤，具体实现还需根据具体的业务需求进行调整。

邮件接收

邮件接收的实现需要使用POP3协议，通过JavaMail提供的API可以方便地实现邮件接收。邮件接收的步骤如下：

创建JavaMail的Session对象；

使用Store对象连接到POP3服务器；

打开Folder对象，并读取邮件；

遍历邮件列表，获取每个邮件的主题、发件人、收件人、时间等信息；

根据需求获取邮件正文和附件，并进行解码；

关闭Folder和Store对象。

需要注意的是，在进行邮件接收时，要先连接到邮件服务器并打开相应的邮件文

件夹，才能读取其中的邮件。在获取邮件的正文和附件时，需要使用MIME协议进行解码，以得到可读的邮件内容和附件文件。

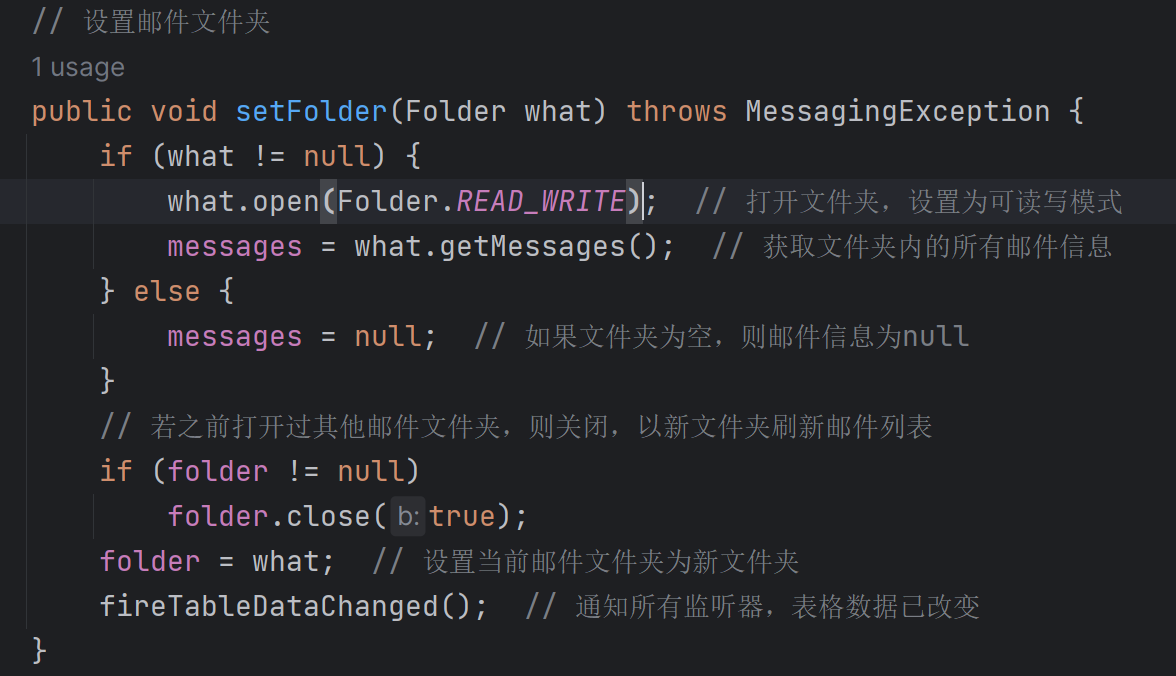


图5.1.5 邮件接受模块关键代码



图5.1.5 邮件发送模块关键代码

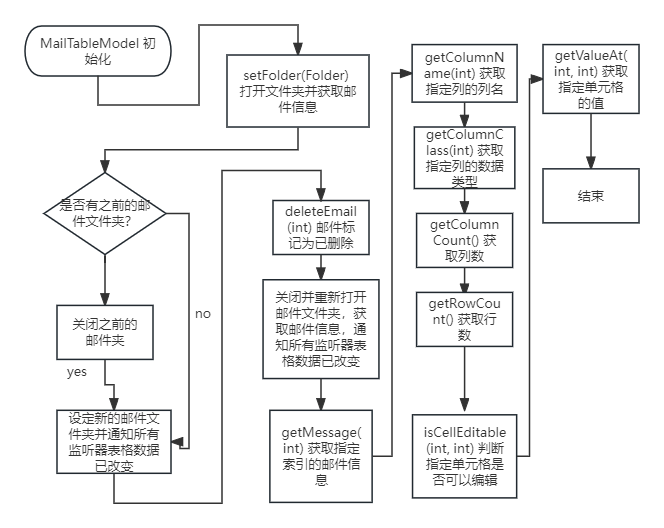


图5.1.5 接受邮件流程图

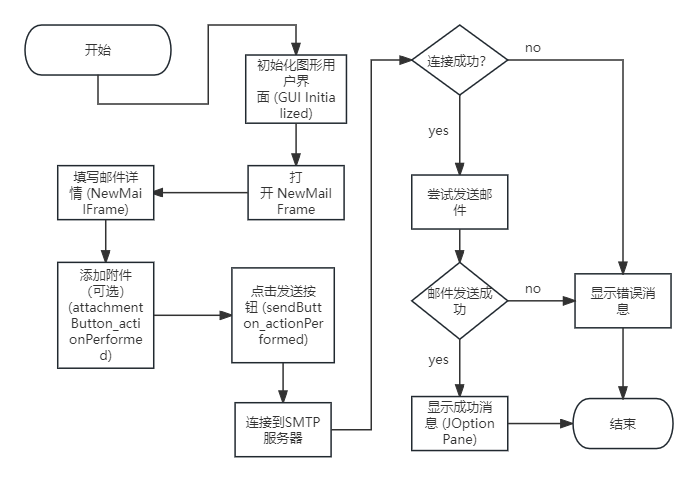


图5.1.5 邮件接受流程图

5.1.6 邮件删除模块

邮件删除模块的流程和工作原理如下：

用户在系统主窗体中选择一个或多个邮件，并点击删除按钮。

系统获取选中邮件的ID或其他唯一标识符。

邮件删除模块通过调用邮件管理模块提供的API，向邮件服务器发送删除请求，并传递选中邮件的ID或其他唯一标识符。

邮件服务器接收到请求后，将选中的邮件从服务器上删除。

邮件删除模块通过调用邮件管理模块提供的API，将选中邮件从本地邮件数据库中删除。

邮件删除模块刷新邮件显示表格，更新显示的邮件列表。

用户可以在表格中看到已删除的邮件已经消失。

需要注意的是，在删除邮件之前，用户需要对删除操作进行确认，以免误删重要邮件。



图5.1.6 邮件删除模块关键代码

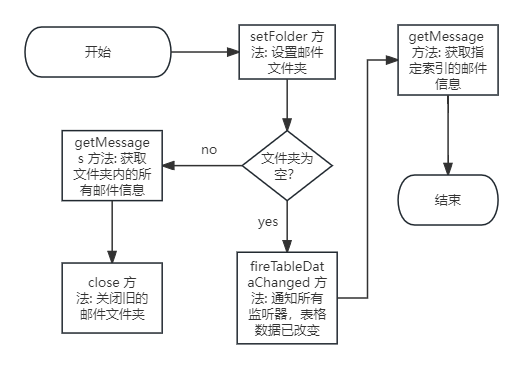


图5.1.6 邮件删除模块流程图

5.1.7 邮件端口模块

系统提供了一个属性文件smtpPop3.properties来存储这些信息。该属性文件包含以下几个属性：

pop3host：POP3邮件服务器地址，如pop3.sina.com.cn；

smtphost：SMTP邮件服务器地址，如smtp.sina.com.cn；

address：收发邮件帐户，如key@sina.com；

username：账户用户名，如key；

password：账户密码，如password。

系统可以通过Java的Properties类来读取该属性文件，并使用setProperty(), getProperty()方法来进行存取属性内容。在获取用户设置的邮件服务器和账户信息后，系统可以使用该信息来实现邮件的发送和接收功能。

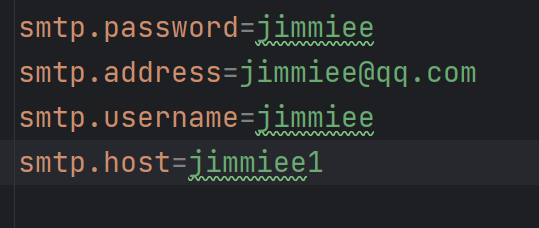


图5.1.7 SMTP配置文件

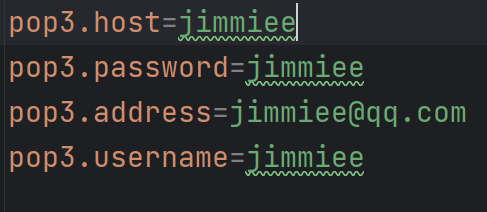


图5.1.7 POP3配置文件

5.2 界面设计

在邮件客户端的界面设计中，主要包括了邮件列表区域、邮件显示区域以及功能按钮等几个关键部分。

邮件列表区域是一个JTable，它用于展示邮件的主题、发件人、收件日期等信息。用户可以通过在列表中点击某一行来选中并查看某一封邮件的具体内容。

邮件显示区域是一个JPanel，它用于展示用户在邮件列表中选中的邮件的详细内容。当用户在邮件列表中选中一封邮件时，该邮件的内容就会在此区域中展示。

除了邮件的展示区域，界面上还设置了一系列的功能按钮，具体如下：

创建邮件按钮：用户点击后，将打开一个新的窗口，在这个窗口中，用户可以编写新的邮件并发送。

接收邮件按钮：用户点击后，将从邮件服务器上接收新的邮件并刷新邮件列表。

删除邮件按钮：用户点击后，将删除邮件列表中被选中的邮件。

POP3/SMTP设置按钮：用户点击后，将打开一个新的窗口，在这个窗口中，用户可以设置POP3/SMTP的相关参数。

当用户点击这些功能按钮时，会触发对应的事件处理函数。例如，当点击删除邮件按钮时，就会触发deleteEmailButton\_actionPerformed函数。这个函数会获取用户在邮件列表中选中的邮件的索引，然后调用MailTableModel类中的删除方法，将邮件从邮件列表中删除。类似地，点击其他的功能按钮也会触发对应的事件处理函数，完成相应的操作。

为了处理这些按钮的点击事件，我们定义了四个适配器类，分别是MainFrame\_receiveEmailButton\_actionAdapter，MainFrame\_propertiesSetButton\_actionAdapter、MainFrame\_createEmailButton\_actionAdapter和MainFrame\_deleteEmailButton\_actionAdapter。

这些适配器类实现了java.awt.event.ActionListener接口，并在actionPerformed方法中调用了主界面类的相应方法，从而完成了对按钮点击事件的处理。

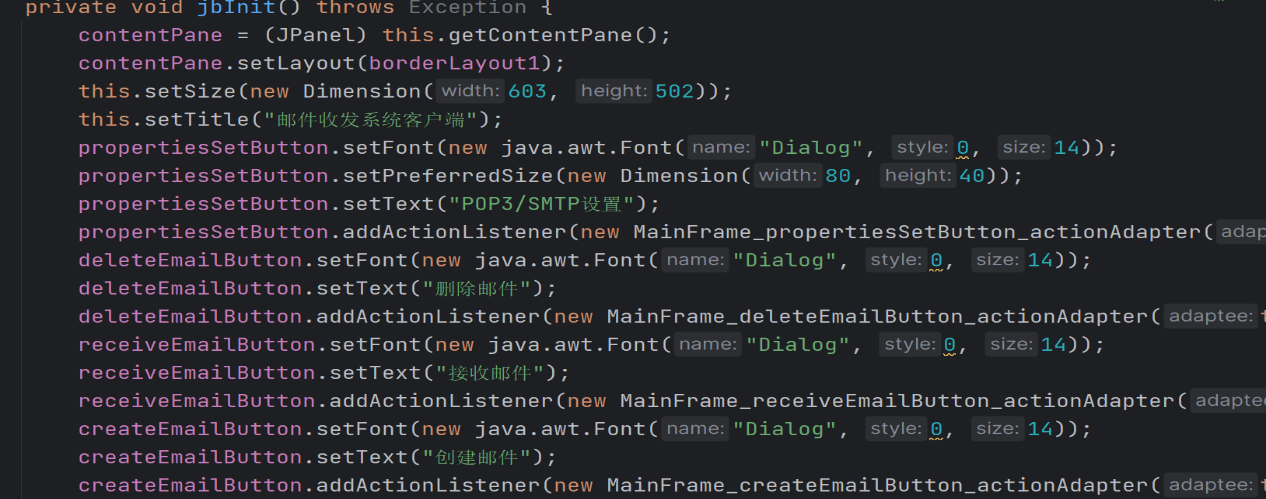


图5.2 界面设计实现关键代码

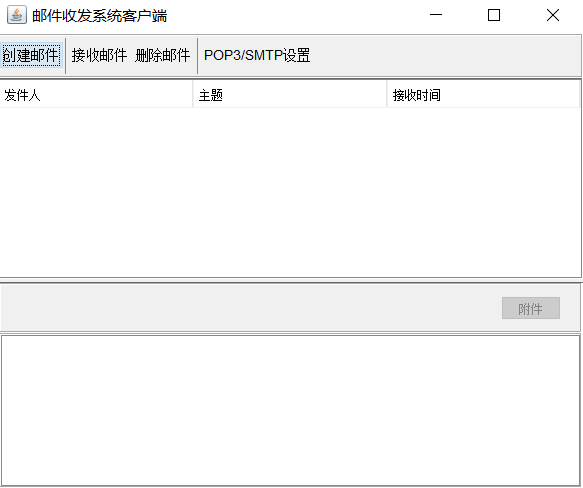


图5.2 主界面

5.3 本章小结

本章主要介绍了邮件收发软件的实现过程，包括系统设计和系统实现两个方面。在系统设计方面，采用了JavaMail和SMTP、POP3、MIME等通信协议，设计了系统的功能模块，表示了各模块之间的关系。在系统实现方面，详细介绍了JavaMail、SMTP、POP3和MIME协议的实现过程，并重点介绍了邮件收发、邮件删除和邮件端口等功能模块的实现。通过本章的实现过程，成功实现了一个基于JavaMail的邮件收发软件，并实现了其基本的功能模块。

第6章 系统测试

6.1 系统总体测试

当系统进行部署之后，整个系统运行正常，如图6-1所示，各个交互界面运行正常：

6.3 总结

本文研究了邮件收发软件的开发与实现，基于JavaMail使用SMTP、POP3、MIME的通信协议。在需求分析的基础上，对JavaMail的使用方法、SMTP、POP3协议的工作原理和实现方法，以及MIME协议的实现方法进行了详细的介绍和分析。

在设计和实现邮件收发模块时，本文采用了MVC架构，通过JavaMail库和相关协议实现了邮件的发送、接收和删除等功能，并对邮件的附件发送和接收进行了测试。测试结果表明，邮件收发软件的功能实现正常，可以满足用户的基本需求。

6.4 未来的展望

随着互联网的发展和普及，电子邮件作为一种便捷、快速、可靠的通信方式，正在被越来越多的人所使用。邮件收发软件的开发和优化仍然具有重要的意义和价值。

未来的研究方向可以从以下几个方面进行探索和改进：

加强用户体验。可以进一步改进界面设计和操作体验，提高软件的易用性和用户满意度。

提高邮件的安全性和保密性。可以通过加密算法等技术手段，保障邮件传输和存储过程中的安全性和保密性。

提高邮件的处理效率。可以通过优化算法和数据结构等技术手段，提高邮件的处理效率和响应速度。

集成其他通信方式。可以将邮件收发软件与其他通信方式（如短信、即时通讯等）进行集成，提供更全面的通信服务。

综上所述，随着技术的不断发展和创新，邮件收发软件将会不断完善和优化，更好地满足用户的需求和期望。