

深圳大学考试答题主页

(以论文、报告等形式考核专用)

二〇一五~二〇一六学年度第二学期

课程编号 1502850003 课程名称 互联网编程 主讲教师 周义朋 评分 _____

学 号 2014150027 姓名 徐荣钦 专业年级 2014 级计算机科学与技术专业

教师评语:

项目名称: 《Android 系统的电子邮件客户端软件》

摘要:

深圳大学课程项目报告

课程名称: 互联网编程

项目名称: 《Android 系统的电子邮件客户端软件》

学 院: 计算机与软件学院

专 业: 2014 年级计算机科学与技术

任课教师: 周义朋

报 告 人: 徐荣钦 学号: 2014150027

提交时间: 2016 年 6 月 20 日

教 务 处 制

一、项目要求

以下三题任选一题。

第一题：

使用 Java 语言设计一个电子邮件客户端软件，详细要求如下：

- a) 使用 IMAP 协议获取电子邮件服务器（如 QQ 信箱）上的电子邮件。----- (10 分)
- b) 使用 SMTP 协议通过服务器发送邮件。----- (10 分)
- c) 电子邮件的附件可以下载和保存，如为图像或者文本文件等格式可以在客户端内查看。----- (10 分)
- d) 发送电子邮件时支持附件功能。----- (10 分)
- e) 在客户端操作邮件时，服务器端应保持同样操作。如客户端删除或者将邮件标注为已读，则服务器端应也删除或标注为已读。----- (10 分)
- f) 其他有用的功能，如地址簿等。----- (10 分)
- g) 在 Android 上实现该软件。----- (15 分)
- h) 代码结构合理，软件界面美观易用。----- (10 分)
- i) 项目报告能够详细，准确的描述项目内容。----- (15 分)

第二题：

使用 Java 语言设计一个 BBS 公告板系统，系统分为客户端和服务器端两部分，详细要求如下：

- a) 支持用户注册、登录、注销功能，用户个人资料管理功能。----- (10 分)
- b) 用户分为游客，普通用户和管理员用户。普通用户进一步分为正常普通用户和禁言普通用户。游客只能浏览其他用户的留言但不能发表留言，管理员用户具有权限可以禁止某个用户发言。----- (10 分)
- c) 系统分为多个讨论区，每个讨论区有独立的主题，并可以设置一到三名管理员负责管理。----- (15 分)
- d) 发布主题和跟帖功能。每个正常的注册用户和管理员可以在任一讨论区发起讨论主题，或者回复任一已发表的帖子。可以发布的内容包括文本和图片。----- (15 分)
- e) 每个用户有一个消息信箱，用于接收系统通知。每个普通用户可以查看和删除信箱

里的消息。但管理员可以群发通知消息。

----- (15 分)

f) 在 Android 上实现该软件客户端（服务器端在 PC 上实现）

----- (10 分)

g) 代码结构合理，软件界面美观易用。

----- (10 分)

h) 项目报告能够详细，准确的描述项目内容。

----- (15 分)

第三题：

自由选题：学生可以根据自己的兴趣，设计其他系统，但需要（1）使用 Java 编程语言实现，涉及互联网编程，所用技术应该为教学大纲里的主要知识点。（2）需要与任课老师沟通，方案得到任课老师的认可。

二、项目功能概述

我所选择的题目是开发一个基于 Android 平台的电子邮件客户端软件，下面将会是对软件所实现的功能以及我所做的工作的进行一个概述（所用功能都有实现，并且额外增添一些较为实用的小功能，特别是界面 UI 部分，希望能得到青睐）

1. 实现了收取邮件功能：收取邮件功能是基于 IMAP 协议的，可以在开通了 IMAP 协议的电子邮箱中获取其邮件的具体信息，以 QQ 邮箱为例，包括：

- (1)、邮件的主题、发件人、收件人（包括群发收件人）、发送时间等基本信息。
- (2)、邮件的附件内容（图像、文本、压缩文件等所有文件）。

2. 实现了电子邮件附件下载、保存、预览。

(1)、对于有附件的邮件，可以下载附件的内容（支持多格式），并且这里默认指定下载到 android 系统的 storage0/MyMail/根目录下。

(2)、对于附件是图片、文本的情况，可以在线预览，而不是直接下载。

3. 实现了可以使用 SMTP 协议通过服务器发送邮件

(1)、使用 SMTP 协议发送一封邮件，包括邮件的主题、发件人、收件人、邮件内容等完整内容。

4. 实现了发送电子邮件时支持附件的功能

(1)、实现了在发送邮件主体详细内容的同时，发送本地机器指定的附件。

(2)、支持发送多附件，并且可以发送多个不同类型的文件（图像、doc、rar 文件等各种类型）。

5、实现在客户端操作邮件时，服务器端应保持同样操作。

(1)、客户端删除或者将邮件标注为已读，则服务器端应也删除或标注为已读。下次客户端进行查看的时候，已读、删除操作会被保留。

(2)、除此之外，参考借鉴 QQ 邮箱，为了使电子邮件客户端操作更加方便，这里我增添用户可以标记电子邮件为“未读”。

6、实现了地址簿功能

(1)、实现了地址簿功能，地址簿里面保存了用户常用的邮件账户的地址，方便用户查找与使用。

(2)、实现了用户手动输入联系人的昵称、电子邮箱地址基本信息功能，并且保存到静态数据库。

7、快速回复功能

(1)、为了便捷用户快速地对当前发邮件人进行邮件回复，用户可以在阅读完邮件详细内容之后，无需手动输入，快速对发件人进行回复。

(2)、此外，用户也可以在地址簿，也就是联系人列表进行快速回复。

8、快速登录功能

(1)、为了使用户更加快捷地登录到邮件客户端，也就是借鉴微信、QQ 等快速进入客户端这样的功能，这里可以让用户选择是否记住站好和密码，从而实现快速登录。

9、联系人管理功能

- (1)、用户可以对联系人进行删除等管理功能。
- (2)、用户可以直接添加新的联系人。

10、显示 HTML 网页内容、图片、文本等内容。

(1)、若邮件正文内容不只是无格式的文字，那么邮件会自动显示为带格式的文字。如果邮件正文是网页内容，也就是一个包括文字、图片、链接的 HTML 文本，也可以正常地显示出来，并且点击网页内容的超链接有效。

- (2)、对于邮件的图片、文本也能成功显示出来。

11、注册与设置 SMTP、IMAP 协议开启功能

(1)、在联网的情况下，用户可以转到另一个页面进行邮件的注册，并且可以登录该邮箱进行开启 IMAP、SMTP 协议的设置，从而让第三方邮件客户端可以连接服务器并且获得数据。

12、界面通用、美观、多元、合理，动画效果多，动态监测网络变化

我充分学习了 Android 的 XML 界面编程技术以及 Android 动画显示技术，让界面更加有“灵性”，并且更加符合 Android 用户常用的界面操作。颜色绚丽、美艳动人。

同时，我利用第三方的开发包的 Animitation 类，可以指定控件的动画方式。

另外，通过 Android 的广播机制，动态地给用户提醒网络数据的变化。

13、电子邮件客户端主要搭建在 Android 平台，在安卓平台实现，若其他人拿到 QQ 邮箱的授权码，是可以顺利登陆该电子邮件客户端并且使用，也即是该软件具有通用性。

14、代码结构清晰易懂，利用模块化编程思想去进行 JAVA 编程，并且也有具体的注释和相应的分析，代码易读、通用！

15、仔细、完整地撰写了实验报告并且成功提交。

三、实现方法

功能 1：

1、功能阐述：

实现了收取邮件功能：收取邮件功能是基于 IMAP 协议的，可以在开通了 IMAP 协议的电子邮箱中获取其邮件的具体信息，以 QQ 邮箱为例子，包括：

- 1)、邮件的主题、发件人、收件人（包括群发收件人）、发送时间等基本信息。
- 2)、邮件的附件内容（图像、文本、压缩文件等所有文件）。

2、功能实现详解（以 QQ 邮箱为例）

前言：我的实验主要用到了 javamail.jar 这个包进行操作，这个包里面封装了包括 IMAP、SMTP、POP3 协议在内的几乎所有邮件客户端与邮件服务器端进行通信的方法。相关的包我也在我的作业中提交了，注意，我的 Android 项目是基于 Eclipse+ADT 插件+SDK 去开发的，当然 Android Studio 也是一个不错的工具，但个人习惯 Eclipse 编程平台。

另外，由于发邮件这部分对下面几个功能的实现都十分重要，因此，我在这里会对代码、实现方法、效果等内容做一个十分详细的解释！

(1)、首先一定要在 QQ 邮箱那里获得授权码，并且开通 IMAP/SMTP 服务，这是重点，不然不能访问邮件服务器，这里的操作，如下图 1：



图 1

(2)、开通了邮箱服务之后，我们需要获得授权码与账号密码（这里以 1251931015@qq.com 作为测试）。要连接到邮箱服务器，就要先配置好相应的连接配置以及认证配置，才能通过邮箱服务器的认证，如下图 2 所示：

```
Properties props = System.getProperties();
props=System.getProperties();
props.put("mail.store.protocol", "imap");
props.put("mail.imap.auth", "true"); //这样才能通过验证
props.put("mail.imap.host", "imap.qq.com");
props.put("mail.imap.port", "993");
props.put("mail.imap.ssl.enable", "true");
props.put("mail.imap.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
props.put("mail.imap.socketFactory.fallback", "false");
Session session=Session.getInstance(props, null);
Store store;
Folder folder;
Message message[] = null;
```

图 2

我的分析与实现说明：

首先由于 JavaMail 需要和邮件服务器进行通信，这就要求程序提供许多诸如服务器地址、端口、用户名、密码等信息，JavaMail 通过 Properties 对象封装这些属性信息。如上图

所示的代码封装了这些属性信息，通过 Properties 类的 put 方法以名值对的方式把属性配置好，注意，几乎大多数邮箱都是支持 SSL 连接的，因此这里需要配置 SSL 参数属性，这样才能在更多的邮箱成功地连接到服务器。注意，这里用的是 IMAP 协议（通过 Properties 对象指定要用的协议即可），并且指定服务器为 imap.qq.com，而且 IMAP 协议的端口号是 993（IMAP 的 SSL 连接），这样就可以通过 IMAP 协议收取邮件了！

(3)、接着，就可以通过 Session 接收各种配置属性信息：通过 Properties 对象设置的属性信息，并且同时根据 JavaMail 的配置文件，初始化 JavaMail 环境，以便通过 Session 对象创建其他重要类的实例。如下图 3 所示：

```
Session session=Session.getInstance(props, null);
Store store;
Folder folder;
Message message[] = null;
try {
    store=session.getStore("imap");
    store.connect(name,mima);
    folder=store.getFolder("INBOX");
    folder.open(Folder.READ_WRITE);
    message= folder.getMessages();
    message = folder.getMessages();
    //store.isConnected()可用于判断是否
} catch (NoSuchProviderException e) {
    // TODO 自动生成的 catch 块
    e.printStackTrace();
} catch (MessagingException e) {
    // TODO 自动生成的 catch 块
    e.printStackTrace();
}
```

图 3

我的分析与说明：

由 Session 对象得到 Store 对象，Store 对象的作用是在有邮箱密码和用户名的前提下与邮箱服务器进行连接，（当然，也可以在构造 Session 对象的时候把 Authentication 的继承子类作为参数传入，构建一个包含密码和用户名的认证类，来与服务器进行交流），然后，连接成功后，得到邮箱的 INBOX 收信箱文件夹也就是 Folder 对象。

最后，调用 Folder 对象的 open 与 getMessages 方法，就可以得到一个 Message 数组对象（每一个 Message 对象就包含了一封邮件）；这里要特别注意 Message 对象，因为 JavaMail 把一封邮件所有的信息，包括基本信息（主题、收件人、发送日期、内容、收件人等）与附件等信息，包含在了 Message 对象里面，因此，后面的大多数操作都是居于 Message 对象或者数组进行操作的！

接着，为了让代码可读性高、清晰并且有条理、模块化程度高、这里我专门封装了一个专门针对某一封邮件进行操作（保存或预览附件、获得内容、判断是否包含附件、收件人详细信息、发件人详细信息等等）的类 MyGetMail。这个类只需要传入 MimeMessage 或 Message 对象，也就是传入一个包含一封邮件所有信息的 Message 或 MimeMessage 对象。

因此，如下图所示，这里用一个循环处理收信箱里面的邮件，获得了邮件的发件人、时间、主题三个用户最关心的信息，如下图 4 所示：

```

for (int i = message.length-1; i >= message.length-len; i--) {
    pmm = new MyGetMail((MimeMessage) message[i]);
    try {
        item=new MailItemUtil(i,pmm.getSubject(),
            pmm.getSentDate(),
            pmm.getReplySign(),
            pmm.isNew(),
            true, //这里不要调用判断附件是否存在的逻辑，会耗流量！
            pmm.getFrom(),
            pmm.getMessageId(),
            GetMailActivity.getImageId(pmm.getFrom()),pmm.getMailAddress("to"),(""));
        android.os.Message m=new android.os.Message();
        m.obj=item;
        m.what=1;
        handler.sendMessage(m);
    } catch (MessagingException e) {
        // TODO 自动生的 catch 语句
    }
}

```

图 4

我的说明与分析：

调用刚才所说的 MyGetMail 的 getSentDate、getFrom、get messageId、getSubject 等方法获得发送时间、发件人、当前邮件的 ID（这个对后面的操作区别于当前时哪一封邮件有很重要的作用）、主题信息，并且通过 ListView 的适配器 Adapter 把这些信息作为列表的形式在软件界面上显示出来。

注意，这里利用了异步操作的思想，因为在 Android 中，耗时的网络操作不可以更改 UI 现成的，因此这里通过 Hanlder 类以及 Handler 类的 android.os.Message 类来进行异步操作，让得到了邮件的详细信息之后，回到主线程去，更改 UI。

由于篇幅关系，下面举例说明我是如何获得某一封邮件的主题、发送日期的，其余的方法都在 MyGetMail 这个类里面，并且有十分详细的注释！如下图 5 所示：

```

// 获得邮件主题

public String getSubject() throws MessagingException {
    String subject = "";
    try {
        subject = MimeUtility.decodeText(mimeMessage.getSubject());
        if (subject == null)
            subject = "";
    } catch (Exception exce) {}
    return subject;
}

//获得邮件发送日期，并且把日期编程格式化
public String getSentDate() throws Exception {
    Date sentdate = mimeMessage.getSentDate();
    SimpleDateFormat format = new SimpleDateFormat(dateformat);
    return format.format(sentdate);
}

```

图 5

分析：这里特别说明一点，邮件的附件名字、主题等明文信息都是加密传输的，这里需要 MimeUtility.decodeText 方法进行解码！然后调用 MimeMessge 对象的 getSubject、getSentDate 就可以轻松获得详细信息。

(4)、最后，按照以上的步骤获得每一封邮件的具体信息之后，就可以通过 ListView 控件以及相关适配器的方法把每一封邮件的详细信息显示出来，如下图 6、7 所示



图 6



图 7

我的实现与分析：

在这里，要特别说明在 android 平台上如何通过 ListView 控件把每一个邮箱 item 通过自定义的布局显示出来，并且可以下拉查看屏幕之外的邮件。首先定义好该 Activity 的布局（包含一个 ListView 对象）和每一个单项 item 的布局，然后针对每一个 item 定义一个包含 item 各个变量的封装类，最后定义一个继承于 Adapter 类的适配器类，这个适配器类主要作用把每一个 item 的图片、文字等信息都显示在 ListView 上。

此外，这里有一些小技巧，比如说，会根据发件人的邮箱后缀来获得相应的图标，这样就让用户更加清晰知道邮箱的来源，并且让界面更加丰富好看。

(5)、ListView 中每一个 item 的点击事件，如下图所示，这里根据点击的是哪一封邮件（就是刚才步骤所说的 ID），把该封邮件的主题、id 号、收件人、发件人、日期等信息获取，以便调用 Intent 跳到另一个显示邮件详细信息的 Activity 时快速取得主要数据，具体实现如下图 8 所示：

```

//监听点击ListView的item事件
private OnItemClickListener itemListener =new OnItemClickListener(){
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        // TODO 自动生成的方法存根
        MailItemUtil item=list.get(position);
        Intent i=new Intent(mContext,DetailActivity.class);
        //把主题，发件人，收件人，时间，附件等内容通过intent都发送到DetailActivity
        i.putExtra("subject", item.getSub());
        i.putExtra("from", item.getFrom());
        i.putExtra("to", item.getTo());
        i.putExtra("date", item.getDate());
        i.putExtra("attach", "附件");
        i.putExtra("id",item.getMsg_id()+"");
        startActivity(i);
    }
}

```

图 8

(6)、获得并且显示邮件的详细内容以及基本内容（具体在 DetailActivity.java 中），参考 QQ 邮箱，这里按照以下的效果图进行显示，美观大方，而且一目了然。

1)、首先讲如何获得邮箱的主题内容 content 与附件的文件名，如下图 9、10 所示

```

public void getMailContent(Part part) throws Exception {
    String contenttype = part.getContentType();
    int nameindex = contenttype.indexOf("name");
    boolean conname = false;
    if (nameindex != -1)
        conname = true;
    // System.out.println("CONTENTTYPE: " + contenttype); //从IMAP协议返回的报文头
    if (part.isMimeType("text/plain") && !conname) {
        bodytext.append((String) part.getContent());
    } else if (part.isMimeType("text/html") && !conname) {
        bodytext.append((String) part.getContent());
    } else if (part.isMimeType("multipart/*")) { //如果该邮件是包含多种文件的MIME类型
        Multipart multipart = (Multipart) part.getContent();
        int counts = multipart.getCount();
        for (int i = 0; i < counts; i++) {
            getMailContent(multipart.getBodyPart(i));
        }
    } else if (part.isMimeType("message/rfc822")) {
        getMailContent((Part) part.getContent());
    } else {}
}

```

图 9

我的实现方法与分析：

把特定的那一封邮件的 Message 对象传到这个函数里面并且转换为 Part 类型的参数，然后就可以读取邮件的主体内容（包括 HTML 文本内容），读取的内容方法 String bodytext 这个变量。

如下图 10，是获取文件名字的操作，这里要特别注意，需要对文件名进行解码才能可读，这里给出最关键的代码详细可以看源代码注释：

```

fileName = mpart.getFileName(); //对文件名进行解码
if (fileName.toLowerCase().indexOf("gb") != -1) { //解码
    fileName = MimeUtility.decodeText(fileName);
}

```

图 10

2)、然后获得邮箱的所有细节的数据之后，我们就可以就利用 Android 控件提供的 set 方法把我们的内容都在控件上显示出来，如下图 11、12、13、14 所示：

首先是点击 ListView 某一个 item 进入等待，且没连接会提醒，如下图 11 所示。获取内容成功后我们可以进入到所有具体内容的界面，如下图 12 所示。

注意，这里巧妙地利用了 ScrollView 控件（使更多的内容可以通过手势的滑动来查看），以及 WebView 控件（可以把文字大小、粗细、颜色等格式显示出来，同时网页、文档格式也可以完美地显示出来！）。此外，也巧妙第利用了 TableLayout 与 TableRow 等控件实现了界面以图 12 的方格与行并在的格式显示出来，这都是 UI 前端的内容，需要花很多去做好，这里不在累赘阐述。

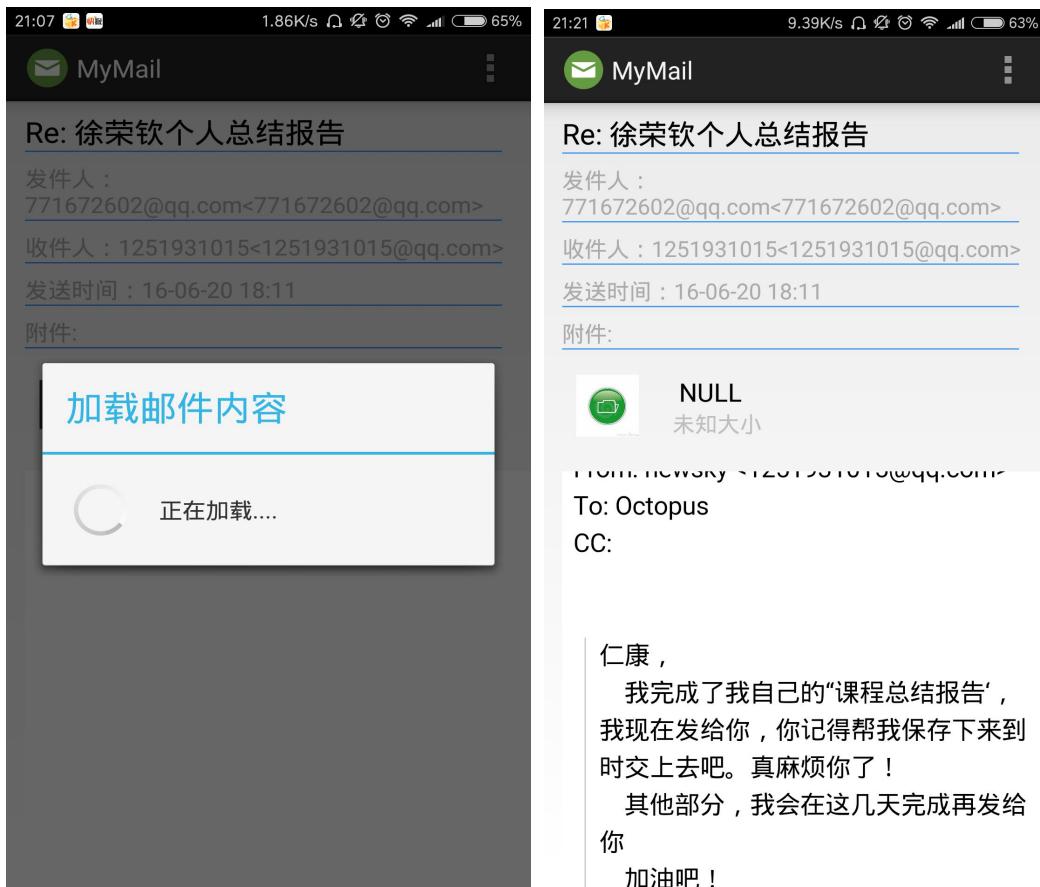


图 11

图 12

此外，带附件的邮件也可以详细地显示出来，并且，可以通过文件的后缀名是图片、文档、压缩文件、其他文件把图标显示出来，并且能显示文件名字（解码后的），如下图 13、14 所示：

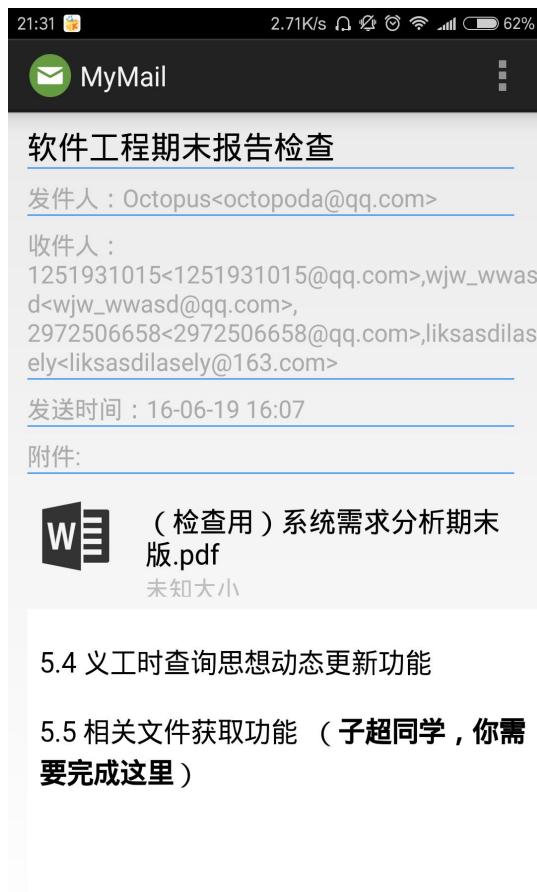


图 13



图 14

至此，获取邮件这个最主要的、最重要的功能的实现方法详细分析以及最终效果图都完美地展示在上面了，由于篇幅关系（大概有 8000 行代码），具体的代码以及注释在项目代码里面。

功能 2:

1、功能阐述:

实现了电子邮件附件下载、保存、预览。

1)、对于有附件的邮件，可以下载附件的内容（支持多格式），并且这里默认指定下载到 android 系统的 storage0/MyMail/根目录下。

2)、对于附件是图片、文本的情况，可以在线预览，而不是直接下载。当然，想要下载的话，我也是提供了十分方便的操作，在下面会有展示。

2、功能实现详解（以 QQ 邮箱为例）

前言：为了更加方便于用户进行操作，使软件更加易用，接近 Android 用户使用习惯，这里用户可以直接在刚才的详细邮箱内容展示 Activity 界面进行操作，这也是借鉴 QQ 邮件进行人性化设置的。

(1)、首先是下载邮件中的附件，关键代码与实现方法如下图 15 所示：

1)、同样地，这里会根据当前邮件第几封邮件（这个在上面收邮件有阐述，会在 DetailActivity 中保存当前邮件的 ID，这个很重要），然后连接 IMAP 服务器（这个也在上面收邮件有详细描述如何通过 Properties 配置参数，并且通过 Session 与 Store 通过服务器验证，这里不在赘述），获得 Message 数组对象之后，通过当前邮件的 ID 获取当前是哪一个邮件，然后通过该邮件的 Message 对象调用封装在 MyGetMail 里面的下载方法。如下图 15 所示：

```
pmm = new MyGetMail((MimeMessage) message[Integer.parseInt(id)]);
try {
    pmm.saveAttachment((Part) message[Integer.parseInt(id)]);
} catch (NumberFormatException e) {
```

图 15

2)、下图 16 是下载附件的详细分析，这里解释关键部分：

首先把邮件的 Message 对象转换为 Part 对象传入，然后通过 getContent() 对象获得一个 Multipart 对象。Multipart 对象十分重要，它包含了一个邮件所有的附件，都可以从这个对象解析出来。

然后递归地从 Multipart 对象中得到 BodyPart 对象，BodyPart 对象包含了某一个附件（因为一封邮件有多个附件）的所有信息。然后从 BodyPart 对象中获得文件的文件名（要解码！）。

最后调用 BodyPart 对象的 getInputStream 方法获取附件的输入流，我们就可以利用输入流去保存文件了！

```

public void saveAttachment(Part part) throws Exception {
    String fileName;
    if (part.isMimeType("multipart/*")) {
        Multipart mp = (Multipart) part.getContent();
        for (int i = 0; i < mp.getCount(); i++) {
            BodyPart mpart = mp.getBodyPart(i);
            String disposition = mpart.getDisposition();
            if ((disposition != null)
                && ((disposition.equals(Part.ATTACHMENT)) || (disposition
                    .equals(Part.INLINE)))) {
                fileName = mpart.getFileName(); //对文件名进行解码
                if (fileName.toLowerCase().indexOf("gb") != -1) { //解码
                    fileName = MimeUtility.decodeText(fileName);
                }
                saveFile(fileName, mpart.getInputStream()); //单文件附件时
            }
        else if (mpart.isMimeType("multipart/*")) {
            saveAttachment(mpart); //表明该邮件包含多附件
        }
    }
}

```

图 16

3）、获得附件的输入流 InputStream 后，调用 MyGetMail 的 saveFile 方法，把附件保存在 android 手机的内存卡的 /MyMail 目录下，也就是 storage0/emulate/0/MyMail/根目录下面，关键的下载分析如下图 17 所示：

```

if(!storefile.exists()){
    BufferedOutputStream bos = null;
    BufferedInputStream bis = null;
    try {
        bos = new BufferedOutputStream(new FileOutputStream(storefile));
        bis = new BufferedInputStream(in);
        byte c[] = new byte[4096];
        while (bis.read(c) != -1) {
            bos.write(c);
            bos.flush();
            bos.flush();
        }
    } catch (Exception exception) {
        exception.printStackTrace();
        throw new Exception("文件保存失败!");
    } finally {
        bis.close();
        bos.close();
    }
}

```

图 17

分析：首先创建一个目录 MyMail。这里根据附件的文件名，在该根目录下新建一个文件，并且获得该文件的输出流；然后定义一个字节数组用于缓冲，文件输入流读取到的字节内容先缓存到 byte 数组，然后通过文件输出流，把字节内容写到文件里面去。这样循环下去，直到文件的所有内容都写到该文件，就完成了文件的下载。

4）、下载的效果图，下载支持文档（doc、xls、ppt、txt等）、图片、压缩文件等几乎所有文件，下面是效果图。长按附件按钮，然后弹出 AlertDialog 对话框（是一个常用的控件），点击下载按钮，就可以下载该文件了，如下图 18、19、20、21、22、23 所示，分别下载了 2 个文件作为尝试。



图 18



图 19



图 20 (下载成功可以打开查看!)



图 21



图 22



图 23

(2)、预览图像

1)、预览图像文件。点击了上述 AlertDialog 的“查看”按钮后，就进入预览图像的 j 活动 NetImageActivity。在各界面中，同样地，利用上述方法与 IMAP 协议的邮箱服务器进行连接，然后根据当前是哪一封邮件（邮件的 ID），去获得图像的输入流，如下图 24 所示：

```
pmm = new MyGetMail((MimeMessage) message[msg_id]);
InputStream is=null;
try {
    is = pmm.getImageInputStream((Part) message[msg_id]);
    bitmap = BitmapFactory.decodeStream(is);
} catch (Exception e) {
    // TODO 自动生成的 catch 块
    e.printStackTrace();
}
finally{
    try {
        is.close();
    } catch (IOException e) {
        // TODO 自动生成的 catch 块
        e.printStackTrace();
    }
}
android.os.Message m=new android.os.Message();
m.obj=bitmap;
m.what=1;
handler.sendMessage(m);
```

图 24

我的实现方法与分析：

首先，这里要获得该邮件的图片附件的输入流，上面在下载的方法中已经详细阐述过，主要是通过该封邮件的 Message 对象获得 Mltipart 对象，再从 Multipart 对象这获取附件对应的 BodyPart 对象，该 BodyPart 对象就可以调用 getInputStream 获得图片的输入流变量 is。

然后，关键的是把 InputStream 对象传入 BitmapFactory.decodeStream 方法中（对输入流进行解码），返回一个 Bitmap 对象。

最后，通过异步的方式，把 Bitmap 对象作为图像显示控件 ImageView 的传入参数，就可显示图片，如下图 25 所示：

```
Bitmap image=(Bitmap)msg.obj;
if(image!=null)
    iv.setImageBitmap(image);
```

图 25

接着，这里给出预览图片效果图，图如下图 26、27 所示：

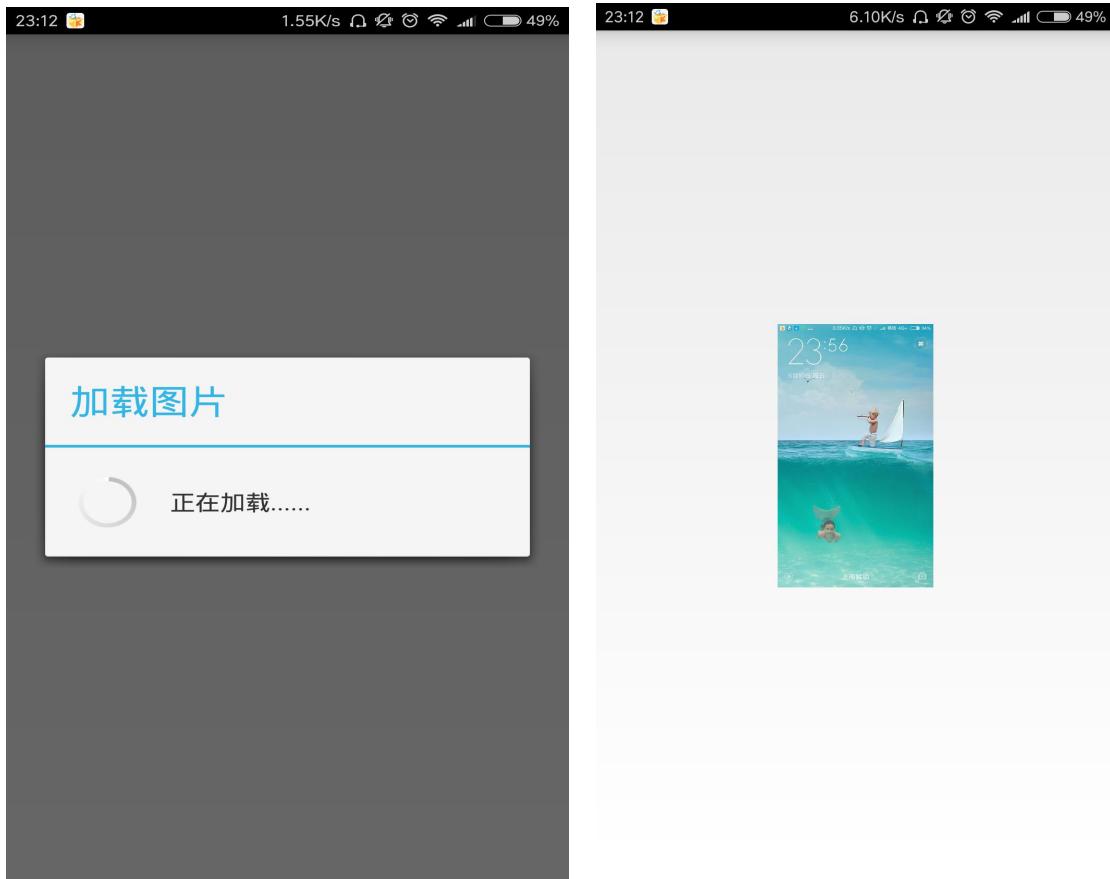


图 26

图 27

(3)、预览文本

1)、这里和预览图像是一样的道理，只不过是另外定义一个 NetTXTActivity，这个活动里面只有一个控件 TextView 控件，并且在 DetailActivity 通过判断文件名字判断是预览图像还是文本；这里获得文本的输入流之后，写到控件 TextView 上面，效果图如下图 28、29 所示：



图 28

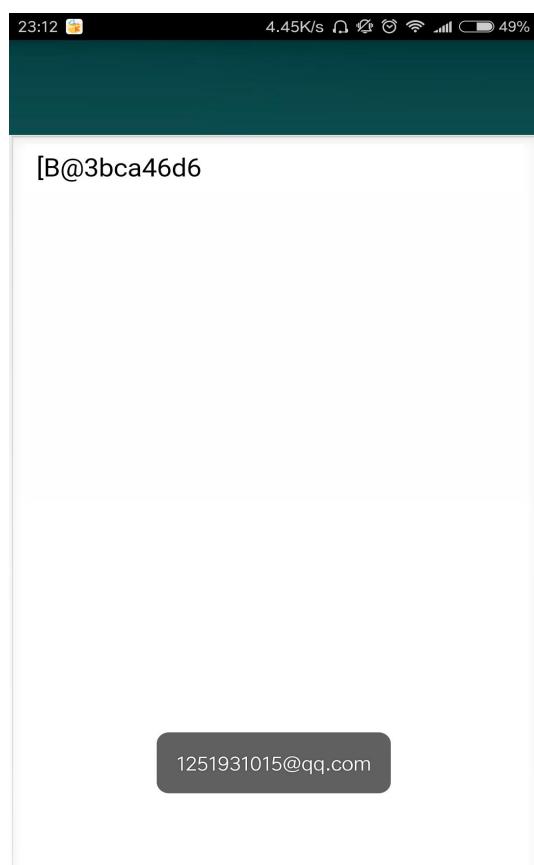


图 29

功能 3:

1、功能阐述:

实现在客户端操作邮件时，服务器端应保持同样操作。

(1)、客户端删除或者将邮件标注为已读，则服务器端应也删除或标注为已读。下次客户端进行查看的时候，**已读、删除**操作会被保留。

(2)、除此之外，参考借鉴 QQ 邮箱，为了使电子邮件客户端操作更加方便，这里我增添用户可以标记电子邮件为“**未读**”。

2、功能实现详解（以 QQ 邮箱为例）

(1)、这里，为了更加方便于用户进行操作，使软件更加易用，接近 Android 用户使用习惯，这里用户可以直接在刚才的详细邮箱内容展示界面进行操作未读、删除、已读的操作，并且通过菜单 Meun 以及 Item 控件、Item 时间的监听事件来进行相应的操作，这样让屏幕空间不那么累赘。如下图 30 所示，把标记未读、标记已读、删除邮件都集成在菜单栏：



图 30

(2)、然后点击菜单的每一个 item，都可以响应事件。我把这些响应时间的操作都封装在一个继承 Thread 的线程类 MyThread 里面，可以根据操作指令向服务器发送标记未读、标记已读、删除邮件等操作。这里只展示标价已读的具体实现操作，其他的类似，如下图 31 所示：

```
if(type.equalsIgnoreCase("hasRead")){
    try {
        message[msg_id].setFlag(Flag.SEEN, true);
    } catch (MessagingException e) {
        // TODO 自动生成的 catch 块
        e.printStackTrace();
        sign="0";
        android.os.Message m=new android.os.Message();
        m.obj=sign;
        m.what=5;//已读
        handler.sendMessage(m);
    }
    sign="1";
    android.os.Message m=new android.os.Message();
    m.obj=sign;
    m.what=6;//已读
    handler.sendMessage(m);
}
```

图 31

分析：同样地，还是按照上面的方式通过 IMAP 协议访问 QQ 邮箱服务器 imap.qq.com，然后根据邮件的 ID 获得该邮件对应的 Message 对象，然后调用 Message 对象的 setFlag 方法把邮件标记为“已读（Flag.SEEN）”，然后服务器就会把该邮件标记为已读。删除、未读操

作差不多，也就是传入的参数不同，下面图 32、33 是效果图：



图 32

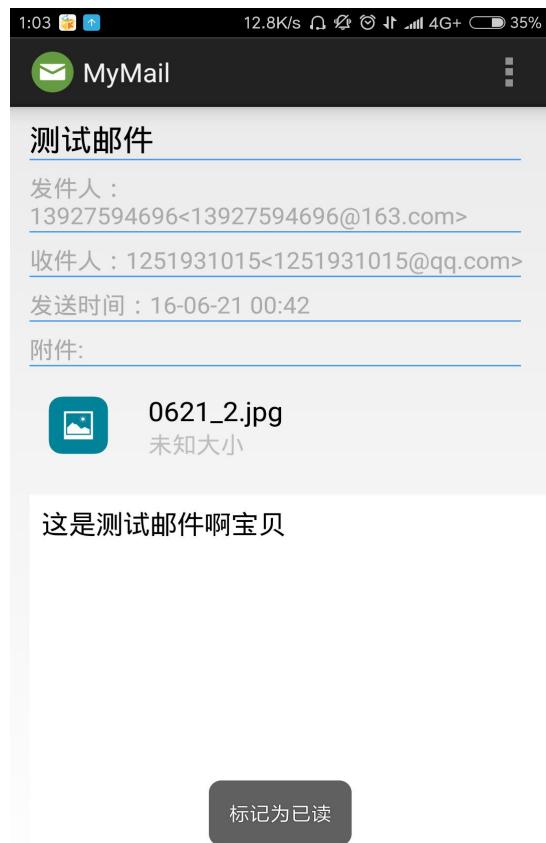


图 33

功能 4:

1、功能阐述：

实现了可以使用 SMTP 协议通过服务器发送邮件

(1)、使用 SMTP 协议发送一封邮件，包括邮件的主题、发件人、收件人、邮件内容等完整内容。

2、功能实现详解（以 QQ 邮箱与 163 邮箱为例）

前言：由于 QQ 邮箱或者网易邮箱的特殊性，我们所做的第三方邮箱通过 SMTP 协议发邮件，会被系统当做垃圾邮箱来处理（这个概率还是挺大的），又或者，过于频繁地通过第三方客户端发邮件的话，会导致邮箱服务器任务这是恶意的广告或攻击；又或者安卓系统本身问题。因此，充分考虑到在 Android 客户端发邮件发不出去的时候，可以考虑用我所做的一个桌面的得.java 文件去发送邮件（这个会稍微稳定一点），这个.java 文件也可以发基本信息与多附件的，功能也十分完善。

(1)、首先，必须要与 SMTP 协议的服务器进行认证与连接，关键代码如下图 34 所示：

```
//首先要配置好连接到服务器的属性和参数
Properties props = new Properties();
props.put("mail.smtp.host", "smtp.qq.com");
props.put("mail.smtp.auth", true);
props.put("mail.smtp.port", 465);
props.put("mail.smtp.ssl.enable", true);
props.put("mail.smtp.auth.login.disable", true);
props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
props.put("mail.smtp.socketFactory.fallback", "false");
props.put("mail.transport.protocol", "smtp");

try {
    Session session = Session.getDefaultInstance(props, new Authenticator() {//default也可
        @Override
        protected PasswordAuthentication getPasswordAuthentication() { // //创建一个认证类
            return new PasswordAuthentication(name, mima); //打开与服务器连接的“窗口”
        }
    });
    session.setDebug(true);
    MimeMessage message = new MimeMessage(session);
```

图 34

我的详细分析：

与上述发邮件的思想一样，必须先指定服务器的端口、地址、SSL 连接、认证所需的账号和密码、传输的协议（SMTP）等信息，然后获得 Session 对象，并且通过 Session 对象构造出来一个 MimeMessage 对象，我们就可以从这个 MimeMessage 对象把主题、收件人、发件人、内容、多附件等信息封在这个 MimeMessage 里面，然后就可以通过 Transport 类的 send() 方法把邮件发给对方。如下图 35、36 所示，都有现成的方法可以发送邮件的。

```
Address addressFrom = new InternetAddress(name); //设置发件人的地址以及昵称。
Address addressTo = new InternetAddress(to); //设置收件人的地址。
message.setText(content); //设置发件人想要发的文字内容
message.setSubject(subject);
message.setFrom(addressFrom); //发件人作为MimeMessage的参数传入
message.addRecipient(Message.RecipientType.TO, addressTo); //设置收件人地址
//message.saveChanges();
```

图 35

```
// 最后把包含附件的邮件内容加到MimeMessage对象里面
message.setContent(multipart); //Multipart对象里面包含了多个附件组成
//加上附件
Transport transport = session.getTransport("smtp");
transport.connect("smtp.qq.com", name, mima); //传入账号和密码，然后与SMTP服务器进行连接
transport.send(message); //利用Transport类把MimeMessage对象传出去，里面包括了邮件的主体内容、
transport.close();
... ...
```

图 36

(2)、因此，通过上面的分析，我们就可以发送一封比较简单的邮件，编辑好后点击“发送”按钮，就可以把邮件发送出去。android 界面以及发送操作如下图 36、37、38、39 所示：



图 36



图 37



图 38



图 39

(3)、另外，我也做了一个 PC 端的.java 文件，同样也可以通过 SMTP 协议来发送邮件，原理是一样的，这样我们就可以在个人计算机的发送邮件了操作与效果图如下图 40、41 所示。这里以 163 邮箱为例子，发送邮件到 1251931015@qq.com 邮箱，在命令行下面输入主题、内容、收件人、以及附件的路径（会保存在一个 List<String> 泛型类对象里面），就可以实现在 PC 发送邮件并且发送附件了！

```
<已终止> SendMail [Java 应用程序] D:\JDK\bin\javaw.exe (2016年6月21日 下午1:21:45)
请输入收件人地址:
1251931015@qq.com
请输入邮件主题:
这是用PC端发邮件的主题
请输入邮件内容:
这是用PC端发邮件的内容
请输入邮件附件数目:
3
请输入第1个附件的绝对路径:
D://java2//1.jpg
请输入第2个附件的绝对路径:
D://java2//2.jpg
请输入第3个附件的绝对路径:
D://java2//3.jpg
发送完毕!
```

图 40

然后到我自己开发的邮箱下面去看，果然，邮件成功发送了！如下图 41、42 所示：



图 41



图 42

功能 5：

1、功能阐述：

实现了发送电子邮件时支持附件的功能

(1)、实现了在发送邮件主体详细内容的同时，发送本地机器指定的附件。

(2)、支持发送多附件，并且可以发送多个不同类型的文件（图像、doc、rar 文件等各类型）。

2、功能实现详解（以 QQ 邮箱与 163 邮箱为例）

(1)、首先要连接到 SMTP 协议的服务器，在上面已经详细阐述，这里只重点侧重如何发附件以及多附件，如下图 43 所示：

```
//加上附件
Multipart multipart = new MimeMultipart(); //这个Multipart对象可以添加多个附件实体
for(int i=0;i<list.size();i++){// //根据List指向的内容来把指定目录下的文件作为附件加到MultiPart对象里面
BodyPart messageBodyPart = new MimeBodyPart(); // BodyPart对象作为附件信息的一个封装
DataSource source = new FileDataSource(list.get(i));
messageBodyPart.setDataHandler(new DataHandler(source));
File file=new File(list.get(i));
messageBodyPart.setFileName(file.getName()); //得到文件的名字
multipart.addBodyPart(messageBodyPart);
}
// 最后把包含附件的邮件内容加到MimeMessage对象里面
message.setContent(multipart); //Multipart对象里面包含了多个附件组成
//加上附件
Transport transport = session.getTransport("smtp");
transport.connect("smtp.qq.com", name,mima); //传入账号和密码，然后与SMTP服务器进行连接
```

图 43

我的分析与详细说明：

这里通过 Javamail 提供的 Multipart 类可以把多个附件（每个附件都用 BodyPart 包裹起来）包裹起来，然后调用邮件对象 Message 的 setContent 方法就可以把包括附件在内的内容组装到一起。

这里为了实现多附件，这里我用的方法是定义一个 List<String 泛型类>，然后把所有的文件的绝对路径都放到里面（通过一个 for 循环来构造多个 BodyPart，也就是多个附件），然后就可以对包含多附件的邮件进行所有内容的组装了。

(2)、对多附件组装完毕，就可以发送邮件，PC 端也是一样的道理，处理并发送多附件的思路方法已经在上面演示。这里只演示在 Android 下的附件发送方法，如下图 44、45、46、47、48 所示。注意，为了让操作更加方便，这里点击附件按钮，就可以跳到系统的文件目录下，并且能显示附件的路径以及名字，十分方便！。



图 44

多附件测试哦哦哦

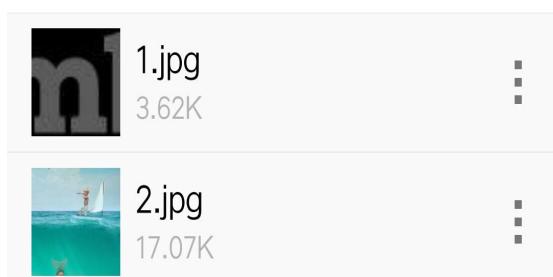


图 45 (QQ 有有邮箱可以看到附件发送成功)

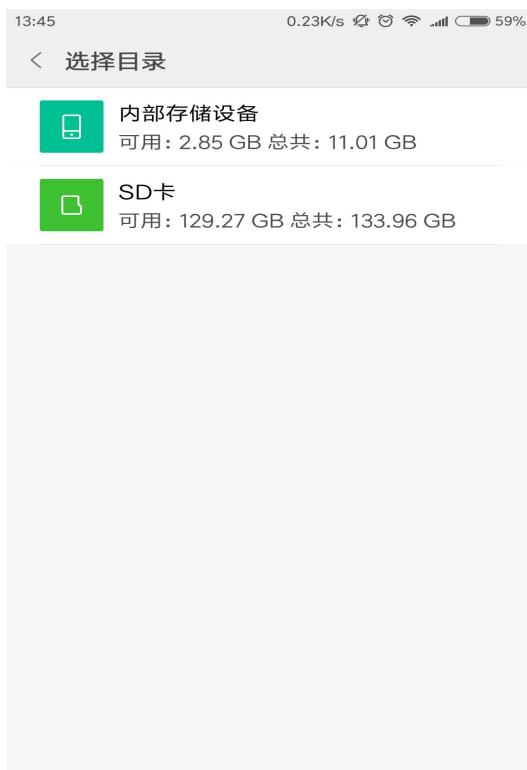


图 46 (系统界面)



图 47



图 48

至此，所有的通过 SMTP 协议发送邮件的操作就成功完成了。

功能 6:

1、功能阐述：

实现了地址簿功能

(1)、实现了地址簿功能，地址簿里面保存了用户常用的邮件账户的地址，方便用户查找与使用。

(2)、实现了用户手动输入联系人的昵称、电子邮箱地址基本信息功能，并且保存到静态数据库。

2、功能实现详解（以 QQ 邮箱与 163 邮箱为例）

(1)、地址簿功能都是基于安卓本地数据库的功能，所有的数据都会用安卓下的 SQLite 数据库保存起来。

首先要创建一个继承自 SQLiteOpenHelper 类的子类 MyDatabaseHelper 类，这个类的作用是用户第一次使用软件的时候在其 onCreate() 方法中创建一个名为 mail.db 的数据库，这个数据库可以放很多张表。如下图 49 所示是 MyDatabaseHelper 类所做的工作：

```

@Override
public void onCreate(SQLiteDatabase db) {
    // TODO 自动生成的方法存根
    db.execSQL(create_contact);
    db.execSQL(user_msg);
    //组装
    ContentValues values=new ContentValues();
    values.put("id", "1");
    values.put("account", "13927594696");
    values.put("password", "1995328mmmm");
    db.insert("user_msg", null, values);
    values.clear();
    Toast.makeText(mContext, "成功创建数据表", Toast.LENGTH_SHORT).show();
}

}

```

图 49

下面是建数据表的语句：

```

public static String create_contact="create table contact (
        +name text primary key,
        +email text)";

```

图 50

建立数据库：

```

public void createDatabase(){
    dbhelper=new MyDatabaseHelper(mContext,"mail.db",null,1);
    dbhelper.getWritableDatabase();
    Toast.makeText(mContext, "DB OK",Toast.LENGTH_SHORT).show();
}

```

图 51

然后往数据表里里面加入数据，利用 ContentValues 这个类就可以把组装好的数据，按照行键值的形式把联系人、联系人的邮箱这些数据保存在数据库里面（当然，我们也会把用户的密码账号保存起来），如下图 51 所示。

```

public void updatemsg(){
    dbhelper=new MyDatabaseHelper(mContext,"mail.db",null,1);
    SQLiteDatabase db=dbhelper.getWritableDatabase();
    ContentValues values=new ContentValues();
    values.put("account", acc);
    values.put("password", pw);
    db.update("user_msg", values, "id=?", new String[]{"1"});
    values.clear();
    db.close();
}

```

图 52

至此，基本的数据库的建数据、建数据表、组装和插入、删除数据的操作就完成了，这个对后面的联系人管理有很大的帮助。

(2)、接着，就是从数据库的某一张表读取数据，由于我把联系人以及对应的邮箱是保存

在一个数据表里面，因此，可以利用刚才的 MyDatabaseHelper 类访问数据库里面的数据表，然后利用 Cursor 类从第一行联系人数据起，遍历到最后一个联系人，然后把这些遍历数据通过 ListView 的形式显示在一个活动中（ListView 这里不再多说）。遍历数据如下图 53 所示：

```
private void init(){//从数据表contact里面拿出数据
    dbhelper=new MyDatabaseHelper(mContext,"mail.db",null,1);
    SQLiteDatabase db=dbhelper.getWritableDatabase();
    //得到所有contact数据
    Cursor cursor =db.query("contact", null, null, null,null,null,null);
    Contact_item item;
    if(cursor.moveToFirst()){
        do{
            //遍历所有数据
            String name=cursor.getString(cursor.getColumnIndex("name"));
            String email=cursor.getString(cursor.getColumnIndex("email"));
            item=new Contact_item(name,email,R.drawable.user);
            list.add(item);
        }
        while(cursor.moveToNext());
    }
    cursor.close();
}
```

图 53

(3)、接着就是效果图的展示，如下图所示 54、55 所示，点击主界面的“地址簿”从数据库中可以得到地址簿列表：



图 54



图 55

功能 7:

1、功能阐述:

实现了联系人管理功能

(1)、用户可以对联系人进行删除等管理功能。

(2)、用户可以直接添加新的联系人。

2、功能实现详解

(1)、地址簿功能都是基于安卓本地数据库的功能，所有的数据都会用安卓下的 SQLite 数据库保存起来。关于数据库的操作不再赘述，下面是用户点击添加“联系人”选项就可以进入编辑界面。

输入联系人的名字以及邮箱并且检查合法了之后，就可以插入到联系人的数据表里面去，实现方法如下图 56 所示：

```
// 口/本机/应用/我的CONTACTS 表的插入操作  
if(useremail.contains("@")&&useremail.contains("."))&&useremail.contains("com")){  
    dbhelper=new MyDatabaseHelper(mContext,"mail.db",null,1);  
    SQLiteDatabase db=dbhelper.getWritableDatabase();  
    //组装联系人数据  
    ContentValues values=new ContentValues();  
    values.put("name", username);  
    values.put("email", useremail);  
    db.insert("contact", null, values);  
    values.clear();  
    //清空，并且回到主界面  
    name.setText("");  
    email.setText("");  
    Intent i=new Intent(mContext,MainActivity.class);  
    startActivity(i);  
    Toast.makeText(mContext, "成功加入联系人数据库",Toast.LENGTH_SHORT).show();  
}  
}
```

图 56

分析：

同样是利用 MyDatabaseHelper 去访问数据库，然后开始组装数据，通过 ContentValues 组装联系人姓名、邮箱地址数据，然后调用 SQLiteDatabase 对象的 insert 方法把这数据插到这个数据表里面去，。

(2)、接着，用户可以在“联系人地址”界面通过长按的方式选择是否删除联系人，删除联系人的逻辑如下图 57 所示：

```
@Override  
public void onClick(DialogInterface dialog, int which) {  
    // TODO 自动生成的方法存根  
    dbhelper=new MyDatabaseHelper(mContext,"mail.db",null,1);  
    SQLiteDatabase db=dbhelper.getWritableDatabase();  
    db.delete("contact", "name=?", new String[]{item.getUser()});  
    //  
    list.remove(position);  
    showData();  
}
```

图 57

详细分析：

这个很简单，判断当前是 ListView 哪一个 item 之后，就可以打开数据库的数据表，接着先删除 Key 为当前联系人名字所对应的那一条数据（delete 方法），然后再把数据重新刷新一下。

(3)、最后，展示用户自己手动添加联系人的界面操作以及删除联系人的界面操作，如下图 58、59 所示：



图 58



图 59

功能 8：

1、功能阐述：

快速回复功能

(1)、为了便捷用户快速地对当前发邮件人进行邮件回复，用户可以在阅读完邮件详细内容之后，无需手动输入，快速对发件人进行回复。

(2)、此外，用户也可以在地址簿，也就是联系人列表进行快速回复。

2、功能实现详解

(1)、其实这里主要就是对字符串的处理，比如说在邮件详细内容界面（DetailActivity）就可以把发件人所在的 TextView 的邮件地址获取，然后通过 Intent 的 put 方法把发件人的数据传到发送邮件的 Activity 中并且设置。

(2)、同样地，点击“联系人列表”后可以立即转到发邮件的 Activity，同时把点击 ListView 的该 item 的邮件地址作为字符串数据传到发邮件的 Activity 并且设置在对应的 Activity。

(3)、下面是效果图，快速回复，如图 60、61 所示：



图 60



图 61

功能 9：

1、功能阐述：

快速登录功能

(1)、为了使用户更加快捷地登录到邮件客户端，也就是借鉴微信、QQ 等快速进入客户端这样的功能，这里可以让用户选择是否记住站好和密码，从而实现快速登录。

2、功能实现详解

(1)、这里用到的是安卓数据库的思想，不过用的是较轻量级的 SharedPreferences，而不是上面所说的 SQLite 数据库。

(2)、这里给用户选择，若用户选择记住密码，就执行把密码以键值对的形式保存起来，下次打开软件的时候就可以直接把账号和密码显示出来，不需要用户多次手动输入。

```
if(box.isChecked()){
    editor=pref.edit();
    editor.putBoolean("r_p",true);
    editor.putString("account", acc.getText().toString());
    editor.putString("password", pw.getText().toString());
}
else{
    editor=pref.edit();
    editor.putBoolean("r_p",false);
    editor.putString("account", "");
    editor.putString("password", "");
    editor.clear();
}
```

图 62

分析，检测到 box 被打勾，就可以把数据保存到在 SharedPreferences 数据库。这个比较简单，这里不在累赘阐述。

(3)、如下图 63 所示，选择记住密码就可以保存密码与账户。



图 63

功能 10:

1、功能阐述：

显示 HTML 网页内容、图片、文本等内容。

(1)、若邮件正文内容不只是无格式的文字，那么邮件会自动显示为带格式的文字。如果邮件正文是网页内容，也就是一个包括文字、图片、链接的 HTML 文本，也可以正常地显示出来，并且点击网页内容的超链接有效。

(2)、对于邮件的图片、文本也能成功显示出来。

2、功能实现详解

(1)、开始的时候，我用 EditText 显示邮件内容的时候，一些带格式的文字或者 HTML 网页时显示不出来的，因此，改用 WebView 控件来显示之后，就可以显示了，非常完美，核心代码分析如下图 64:

```
Data c=(Data)msg.obj;
dc.loadDataWithBaseURL(null, c.getC().trim().toString(), "text/html", "utf-8", null)
dc.getSettings().setJavaScriptEnabled(true);
dc.setWebChromeClient(new WebChromeClient());
```

图 64

分析：

首先是从服务器获得邮件内容的数据，这里把这些内容都转换成字符串，并且包装在自定义的 Data 类中，然后异步传给 WebView 对象 dc，调用其 loadDataWithBaseURL 方法并且指定其 MIME 类型，接着调用以上的两个方法就可以把 HTML 内容显示出来。

(2)、下图 65、66 是效果图，分别显示了带格式文字与网页的显示：

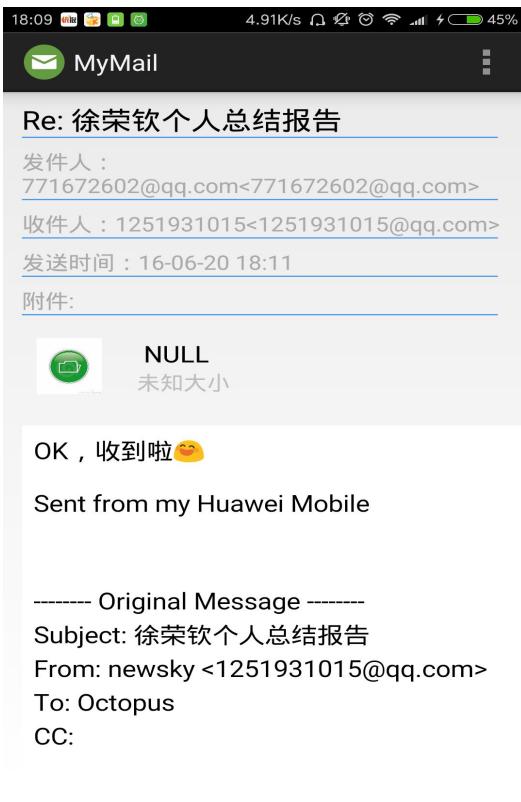


图 65



图 66

功能 11:

1、功能阐述:

注册与设置 SMTP、IMAP 协议开启功能

(1)、在联网的情况下，用户可以转到另一个页面进行邮件的注册，并且可以登录该邮箱进行开启 IMAP、SMTP 协议的设置，从而让第三方邮件客户端可以连接服务器并且获得数据。

2、功能实现详解

(1)、这个主要是利用 WebView 指定一个 URL,，然后在联网的前提下就可以赚到 163 网易邮箱的注册界面

(2)、效果图如下图 67



[先进入网页版邮箱](#)

图 67

功能 12:

1、界面显示美观、富有动画、动态监测网络变化

(1)、我充分学习了 Android 的 XML 界面编程技术以及 Android 动画显示技术，让界面更加有“灵性”，并且更加符合 Android 用户常用的界面操作。颜色绚丽、美艳动人。

(2)、同时，我利用第三方的开发包的 Animation 类，可以指定控件的动画方式。

(3)、另外，通过 Android 的广播机制，动态地给用户提醒网络数据的变化。

2、功能实现详解

- (1)、可通过 XML 的嵌套编程思想去编出非常好看的界面，这是需要特别多的时间去研究和实现的，并且要适合各个场合下的 Android 系统机器。最重要的是利用好 ListView、LinearLayout、ScrollView、TableLayout 等几个控件与布局。
- (2)、对于控件动画显示技术，这里我用了第三方的开发包的 Animation 类，可以指定控件的动画方式。
- (3)、android 广播机制，这里我利用 android 的广播机制动态监测网络的变化，在有需要的 Activity 中通过广播接收类动态地检测网络变化，当网络突然无连接或者突然连接都可以 Toast 显示出来。
- (4)、下面展示几个好看的而又没有展示过的欢迎界面、登录界面、主界面与动态监测网络的例子，动画可以真机感受。如下图 68、69、70、71 所示：

18:28 45%



18:28 45%



图 68

用户必读

注册

图 69



图 70



图 71

其他

1、代码结构合理、格式清晰有条理。

我写代码准循着模块化的思想，不仅让代码可读性增强，而且有利于调试，让项目更好地管理与开发。

另外，代码中的关键处也有十分详细的注释。有利于后续开发。

2、基于 Android 平台，我的软件是基于 android 平台的，软件做到简单易用、功能强大、可开发性强。

3、报告的撰写十分详细，每个功能都有具体的实现分析。另外，可以在项目里面找到 APK 文件去做实际的使用。

四、项目总结

1、这次大作业真的学到太多太多了，在安卓编程上与 JAVA 网络编程上的能力都大大增强，充分学学会了安卓的各种组件以及各种与网络打交道的方法与技巧。

2、充分地学会了如何利用 IMAP 协议获取邮件以及 SMTP 协议发送邮件，基本的访问邮箱服务器、接收邮件、发送带附件邮件等功能都会实现，并且搞懂了原理。

3、我也遇到了不少的困难，比如说，

(1)、在显示邮件内容的时候，出现很多乱码，并且 HTML 图片不能显示，这个让我痛苦很久，后来知道可以用 WebView 控件去处理。

(2)、在与邮箱服务器进行连接并且获取邮箱的多个邮件的时候，开始遇到获取不到的情况。后来发现原因是没有在 Properties 属性里面把 SSL 连接这个属性配置好。

(3)、另外，在 android 上用 ListView 控件显示邮箱的详细内容时，经常遇到显示不出来的情况，后来发现是在适配器类的处理时太粗心了，没有把邮件的字符串传进去。

(4)、XML 界面编写方面，由于没有 XML 基础，因此在编写界面实花费了很多时间，并且也遇到了界面很丑的情况，后来花时间去研究才编写出精美的界面。这需要时间去好好研究的。

(5)、发邮件的时候，经常遇到邮件发布出去的情况，或者附件无法发送的问题。后来仔细 DEBUG，才发现也是属性配置没弄好。

(6)、在开发过程中，安卓程序经常闪退，原因多数是 NullPointerException，这都是贬称过程中粗心的表现。

(7)、在数据库的删除、插入、修改等操作中容易误操作，这是因为没有数据库基础。后来，仔细研究 android 自带的 SQLite 数据库后才搞定。

4、在设计和实现过程里，我充分做到用户友好型，并且要处理连接服务器和接受邮件等过程中所发生的各种程序异常，因为手机网络不稳定，我需要对这种不稳定环境下进行处理和考虑等等。

5、我更加熟悉了面向对象的编程思想，充分利用 JAVA 面向对象的思想和方法：在 连接到邮箱服务器、交互、程序之间的数据交流和交换的过程中利用面向对象的编程思想和编程技术去实现关键功能。