开发过程文档

第一部分 前言

本项目是利用 python socket 编程实现的一个利用 SMTP 发送 MIME 协议邮件的邮件客户端。之所以使用 MIME 协议发送邮件，是因为 SMTP 服务器在接收邮件内容时，当接收到只有一个“.”字符的单独行时，就会认为邮件内容已经结束，如果一封邮件正文中正好有内容仅为一个“.”字符的单独行，SMTP 服务器就会丢弃掉该行后面的内容，从而导致信息丢失。然而本项目在发送图片或者附件时可能出现这种情况，所以就采用 MIME 协议发送邮件。SMTP 服务器使用的 RFC822 邮件格式只适合用来表达纯文本的邮件内容，所以，要使用 RFC822 邮件格式发送这些非 ASCII 码的二进制数据时，必须先采用某种编码方式将它们“编码”成可打印的 ASCII 字符后再作为 RFC822 邮件格式的内容。邮件阅读程序在读取到这种经过编码处理的邮件后，再按照相应的解码方式解码出原始的二进制数据，这样就可以借助 RFC822 邮件格式来传递多媒体数据了，也就可以实现发送图片或者附件的功能。

第二部分 开发过程报告

```
__username=''
__password=''
__recipient=''
msg = b'\r\n'
endmsg = b'\r\n.\r\n'
mailserver = ('smtp.qq.com', 465)
heloCommand = b'HELO qq.com\r\n'
loginCommand = b'AUTH login\r\n'
dataCommand = b'DATA\r\n'
quitCommand = b'QUIT\r\n'
msgsubject = b'Subject: Test E-mail\r\n'
msgtype = b"Content-Type: multipart/mixed;boundary='BOUNDARY'\r\n\r\n"
msgboundary = b'--BOUNDARY\r\n'
msgmailer = b'X-Mailer:mengqi\'s mailer\r\n'
msgMIMI = b'MIME-Version:1.0\r\n'
msgfileType = b"Content-type:application/octet-stream;charset=utf-8\r\n"
msgfilename = b"Content-Disposition: attachment; filename=''\r\n"
msgimgtype = b"Content-type:image/gif;\r\n"
msgimgname = b"Content-Disposition: attachment; filename=''\r\n"
msgtexthtmltype = b'Content-Type:text/html;\r\n'
msgimgId = b'Content-ID:<test>\r\n'
msgimgscr = b'<img src="cid:test">'
mailcontent = ''
__clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

此处首先定义了 username，password，recipient 三个量，分别代表了用户 qq 邮箱地址，授权码,以及收信人邮箱地址，注意此处的 password 指代的不是 qq 密码，而是 qq 邮箱分配的授权码，授权码的获取方式见说明文档“软件使用说明文档”的第三部分。

然后是一些主要的文件头文件体字段，此处参考网上的一些定义资料，mailserver 指的是 smtp 服务器的地址，和使用的端口号。Helocommand，logincommand，datacommand，quitcommand 就像 smtp 服务器之间的握手一样，两边建立好连接后，smtp 邮件客户端向服务器分别指出发信人和收件人的邮箱，然后开始发送邮件。

这里面的 msgtype 指的是对于某个资源的 MIME 消息，它消息头中需要指定资

源的数据类型，这里用的是 multipart/mixed 表示消息体中的内容是混合组合类型，可以是文本、声音、附件等不同邮件内容混合，然后每段数据之间使用 boundary 属性中指定的字符文本作为分隔标识符。Msgmailer 指的是该邮件从哪里发来，msgMIMI 指的是 MIME 协议版本，msgfiletype 和 msgimgtype 均指消息体的数据类型分别为文件或者图片时。img src="cid:test 指用该语句来引用图片资源。

```
def login(self):

    root = Tk()
    root.title("登录界面")
    frame = Frame(root)
    frame.pack(padx=8, pady=8, ipadx=4)
    self.__sslclientSocket.send(self.loginCommand)
    recv2 = self.__sslclientSocket.recv(1024).decode('utf-8')
    lab1 = Label(frame, text="邮箱号:")
    lab1.grid(row=0, column=0, padx=5, pady=5, sticky=W)
    u = StringVar()
    ent1 = Entry(frame, textvariable=u)
    ent1.grid(row=0, column=1, sticky='ew', columnspan=2)
    lab2 = Label(frame, text="授权码:")
    lab2.grid(row=1, column=0, padx=5, pady=5, sticky=W)
    p = StringVar()
    ent2 = Entry(frame, textvariable=p)
    ent2.grid(row=1, column=1, sticky='ew', columnspan=2)
    def getuser():
        self.__username = ent1.get()
        self.__password = ent2.get()
        username = b'%s\r\n' % base64.b64encode(self.__username.encode('utf-8'))
        self.__sslclientSocket.send(username)
        recv = self.__sslclientSocket.recv(1024).decode('utf-8')
        password = b'%s\r\n' % base64.b64encode(self.__password.encode('utf-8'))
        self.__sslclientSocket.send(password)
        recv = self.__sslclientSocket.recv(1024).decode('utf-8')
        if recv[:3] != '235':
            messagebox.showinfo('提示信息','登录失败：账号或密码错误，请使用授权码登录. 235 reply not received from server!')
            messagebox.showinfo('提示信息!','正在重试！')
```

```
from tkinter import *
from tkinter import messagebox
import socket
import ssl
import base64
import time
import os
import random
```

Login 函数包含了一系列功能，登录界面的实现，登录成功后出现的新界面，消

息提示等等。本项目的 UI 设计是使用 python 自带的 tkinter 库实现的，功能比较简单和单一，如图所示为登录界面，提示用户用户输入邮箱号和授权码


```
button = Button(frame, text="登录", command=getuser, default='active')
button.grid(row=2, column=1)
lab3 = Label(frame, text="")
lab3.grid(row=2, column=0, sticky=W)
button2 = Button(frame, text="退出", command=quit)
button2.grid(row=2, column=2, padx=5, pady=5)
```

此处为登录这个按钮定义了一个事件，当点击登录按钮时，触发 getuser 事件。

```
def getuser():
    self.__username = ent1.get()
    self.__password = ent2.get()
    username = b'%s\r\n' % base64.b64encode(self.__username.encode('utf-8'))
    self.__sslclientSocket.send(username)
    recv = self.__sslclientSocket.recv(1024).decode('utf-8')
    password = b'%s\r\n' % base64.b64encode(self.__password.encode('utf-8'))
    self.__sslclientSocket.send(password)
    recv = self.__sslclientSocket.recv(1024).decode('utf-8')
    if recv[:3] != '235':
        messagebox.showinfo('提示信息','登录失败：账号或密码错误，请使用授权码登录. 235 reply not received from server!')
        messagebox.showinfo('提示信息!','正在重试! ')
        root.destroy()
        self.login()
    else:
        messagebox.showinfo('提示信息!','登录成功! ')
        root.destroy()
        mailsenderCommand = b'MAIL FROM:<%s>\r\n' % self.__username.encode('utf-8')
        self.__sslclientSocket.send(mailsenderCommand)
```

Getuser 首先获取邮箱号这个文本框中的类容，然后获取授权码这个文本框中的内容，将它们赋予 username 和 password，然后使用安全套接层 sslclientsocket 发送信息。

但是在发送之前需要对其进行编码，使用 base64 的编码方式编码，返回的 recv 也相应的解码，返回的结果如果不为 235，则验证失败，需要重新登录，窗口被销毁，然后重新生成一个新窗口。反之则是登录成功。进入新的界面

```
正在连接服务器……
成功连接服务器……
正在请求服务器响应……
成功请求服务器响应……
235 Authentication successful
```

```
messagebox.showinfo('提示信息!','登录成功! ')
root.destroy()
mailsenderCommand = b'MAIL FROM:<%s>\r\n' % self.__username.encode('utf-8')
self.__sslclientSocket.send(mailsenderCommand)
root2 = Tk()
root2.title("发送界面")
root2.wm_geometry('400x400')
frame2 = Frame(root2)
frame2.pack(padx=8, pady=8, ipadx=4)
lab10 = Label(frame2, text="收件人邮箱号:")
lab10.grid(row=0, column=0, padx=5, pady=5, sticky=W)
v = StringVar()
ent3 = Entry(frame2, textvariable=v)
ent3.grid(row=0, column=1, sticky='ew', columnspan=2)
lab11 = Label(frame2,text="邮件主题:")
lab11.grid(row=1, column=0, padx=5, pady=5, sticky=W)
w = StringVar()
ent4 = Entry(frame2, textvariable=w)
ent4.grid(row=1, column=1, sticky='ew', columnspan=2)
lab12 = Label(frame2,text="邮件正文:")
lab12.grid(row=2, column=0, padx=5, pady=5, sticky=W)
x = StringVar()
ent5 = Entry(frame2,text="textvariable=x")
ent5.grid(row=2, column=1, sticky='ew', columnspan=2)
lab13 = Label(frame2,text="文件路径:")
lab13.grid(row=3, column=0, padx=5, pady=5, sticky=W)
y = StringVar()
ent6 = Entry(frame2, textvariable=y)
ent6.grid(row=3, column=1, sticky='ew', columnspan=2)
filepath =""
lab14 = Label(frame2,text="图片路径:")
lab14.grid(row=4, column=0, padx=5, pady=5, sticky=W)
z = StringVar()
ent7 = Entry(frame2, textvariable=z)
ent7.grid(row=4, column=1, sticky='ew', columnspan=2)
imgpath=""
```

发送界面如图所示，用户需要输入收件人的邮箱号，邮件主题和邮件正文，是否

需要添加附件或者添加图片则由用户自己决定，若需要添加附件或者添加图片，则输入相应附件的绝对路径或者图片的绝对路径即可，点击发送就可以发送邮件信息。同样的为发送按钮定义了一个事件 sendmail，当点击发送按钮后就会触发这个事件，内部实现如下：

```
def sendmaill():
    self.__recipient = ent3.get()
    subject = ent4.get()
    filepath = ent6.get()
    imgpath = ent7.get()
    #print(self.__recipient)
    #print(subject)
    #print(filepath)
    #print(imgpath)
    if filepath=="":
      if imgpath=="":
        mailrecipientCommand = b'RCPT TO:<%s>\r\n' % self.__recipient.encode('utf-8')
        self.__sslclientSocket.send(mailrecipientCommand)
        recv = self.__sslclientSocket.recv(1024).decode('utf-8')
        self.__sslclientSocket.send(self.dataCommand)
        recv = self.__sslclientSocket.recv(1024).decode('utf-8')
        self.msgsubject = b'Subject: %s\r\n' % subject.encode('utf-8')
        self.__sslclientSocket.send(self.msgsubject)
        self.__sslclientSocket.send(self.msgmailer)
        self.__sslclientSocket.send(self.msgtype)
        self.__sslclientSocket.send(b'Content-Transfer-Encoding:7bit\r\n\r\n')
        self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
        self.__sslclientSocket.send(b'Content-Type: text/html;charset=utf-8\r\n')
        self.__sslclientSocket.send(b'Content-Transfer-Encoding:7bit\r\n\r\n')
        self.mailcontent = ent5.get()
        self.__sslclientSocket.sendall(b'%s\r\n'%self.mailcontent.encode('utf-8'))
        self.__sslclientSocket.send(self.endmsg)
```

If filepath==“”&&if imgpath==“”这就表示用户既不需要发送附件也不需要发送图片，只需要发送纯文本的邮件。首先需要将收件人的邮箱编码，然后使用安全套接字层 sslsocket 发送，邮件主题编码发送，将邮件主题，发件人邮箱和邮件类别发送后开始接收正文部分，Content-Transfer-Encoding 头字段用于指定 MIME 消息体中的内容所采用的邮件编码方式，7bit 指消息体内容全部是没有经过编码的 ASCII 字符。content-Type 指定资源的数据类型，此处是 text 和 html 型，发送纯文本型邮件，准备完成后，获取邮件正文里面的内容，即 self.mailcontent =ent5.get()，然后将接收到的文本内容全部发送，并且发送结束信息。

```
mailrecipientCommand = b'RCPT TO:<%s>\r\n' % self.__recipient.encode('utf-8')
self.__sslclientSocket.send(mailrecipientCommand)
recv = self.__sslclientSocket.recv(1024).decode('utf-8')
self.__sslclientSocket.send(self.dataCommand)
recv = self.__sslclientSocket.recv(1024).decode('utf-8')
self.msgsubject = b'Subject: %s\r\n' % subject.encode('utf-8')
self.__sslclientSocket.send(self.msgsubject)
self.__sslclientSocket.send(self.msgmailer)
self.__sslclientSocket.send(self.msgtype)
self.__sslclientSocket.send(b'Content-Transfer-Encoding:7bit\r\n\r\n')
self.mailcontent = ent5.get()
if os.path.isfile(imgpath):
  time.sleep(0.1)
  filename = os.path.basename(imgpath)
  randomid = filename.split('.')[1]+str(random.randint(1000, 9999))
  time.sleep(0.1)
  self.msgimgId = b'Content-ID:%s\r\n' % randomid.encode('utf-8')
  self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
  self.__sslclientSocket.send(self.msgimgtype)
  self.__sslclientSocket.send(self.msgimgId)
  self.msgimgname = b"Content-Disposition: attachment; filename='%s'\r\n" % filename.encode('utf-8')
  self.__sslclientSocket.send(self.msgfilename)
  time.sleep(0.1)
  self.__sslclientSocket.send(b'Content-Transfer-Encoding:base64\r\n\r\n')
  self.__sslclientSocket.send(self.msg)
  fb = open(imgpath, 'rb')
```

```
fb = open(imgpath, 'rb')
while True:
    filedata = fb.read(1024)
    if not filedata:
        break
    self.__sslclientSocket.send(base64.b64encode(filedata))
    time.sleep(0.1)
fb.close()
time.sleep(0.1)
self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
self.__sslclientSocket.send(self.msgtexthtmltype)
self.__sslclientSocket.send(b'Content-Transfer-Encoding:8bit\r\n\r\n')
msgimgscr = b'<img src="cid:%s">'%randomid.encode('utf-8')
time.sleep(0.1)
self.__sslclientSocket.send(msgimgscr)
time.sleep(0.1)
self.__sslclientSocket.sendall(b'%s' % self.mailcontent.encode('utf-8'))

time.sleep(0.1)
self.__sslclientSocket.send(self.endmsg)
```

If filepath ==""else 表示用户不需要发送附件，但是需要发送图片文件。

同样的需要将收件人的邮箱编码，然后使用安全套接字层 sslsocket 发送，邮件主题编码发送，将邮件主题，发件人邮箱和邮件类别发送后开始接收正文部分，以及打开图片，代码中的 time.sleep（0.1）是为了给系统反映时间，否则发送的图片可能存在显示不完全的情况。发送的方式均为先将相应内容编码成可打印的

ASCII 字符后再发送。

```
else :
    if imgpath=="":
        mailrecipientCommand = b'RCPT TO:<%s>\r\n' % self.__recipient.encode('utf-8')
        self.__sslclientSocket.send(mailrecipientCommand)
        recv = self.__sslclientSocket.recv(1024).decode('utf-8')
        self.__sslclientSocket.send(self.dataCommand)
        recv = self.__sslclientSocket.recv(1024).decode('utf-8')
        self.msgsubject = b'Subject: %s\r\n' % subject.encode('utf-8')
        self.__sslclientSocket.send(self.msgsubject)
        self.__sslclientSocket.send(self.msgmailer)
        self.__sslclientSocket.send(self.msgtype)
        self.__sslclientSocket.send(b'Content-Transfer-Encoding:7bit\r\n\r\n')
        self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
        self.__sslclientSocket.send(b'Content-Type: text/html;charset=utf-8\r\n')
        self.__sslclientSocket.send(b'Content-Transfer-Encoding:7bit\r\n\r\n')
        self.mailcontent = ent5.get()
        self.__sslclientSocket.sendall(b'%s\r\n'%self.mailcontent.encode('utf-8'))
        if os.path.isfile(filepath):
            filename = os.path.basename(filepath)
            self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
            self.__sslclientSocket.send(self.msgfileType)
            self.msgfilename = b"Content-Disposition: attachment; filename='%s'\r\n" % filename.encode('utf-
            self.__sslclientSocket.send(self.msgfilename)
            self.__sslclientSocket.send(b'Content-Transfer-Encoding:base64\r\n\r\n')
            self.__sslclientSocket.send(self.msg)
```

```
            fb = open(filepath,'rb')
            while True:
                filedata = fb.read(1024)
                # print(filedata)
                if not filedata:
                    break
                self.__sslclientSocket.send(base64.b64encode(filedata))
                time.sleep(1)
            fb.close()
            time.sleep(0.1)
            self.__sslclientSocket.send(self.endmsg)
```

```
else :
    mailrecipientCommand = b'RCPT TO:<%s>\r\n' % self.__recipient.encode('utf-8')
    self.__sslclientSocket.send(mailrecipientCommand)
    recv = self.__sslclientSocket.recv(1024).decode('utf-8')
    self.__sslclientSocket.send(self.dataCommand)
    recv = self.__sslclientSocket.recv(1024).decode('utf-8')
    self.msgsubject = b'Subject: %s\r\n' % subject.encode('utf-8')
    self.__sslclientSocket.send(self.msgsubject)
    self.__sslclientSocket.send(self.msgmailer)
    self.__sslclientSocket.send(self.msgtype)
    self.__sslclientSocket.send(b'Content-Transfer-Encoding:7bit\r\n\r\n')
    self.mailcontent = ent5.get()
    if os.path.isfile(imgpath):
      time.sleep(0.1)
      filename = os.path.basename(imgpath)
      randomid = filename.split('.')[1]+str(random.randint(1000, 9999))
      time.sleep(0.1)
      self.msgimgId = b'Content-ID:%s\r\n' % randomid.encode('utf-8')
      self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
      self.__sslclientSocket.send(self.msgimgtype)
      self.__sslclientSocket.send(self.msgimgId)
      self.msgimgname = b"Content-Disposition: attachment; filename='%s'\r\n" % filename.encode('utf-8')
      self.__sslclientSocket.send(self.msgfilename)
      time.sleep(0.1)
      self.__sslclientSocket.send(b'Content-Transfer-Encoding:base64\r\n\r\n')
      self.__sslclientSocket.send(self.msg)
```

```
fb = open(imgpath, 'rb')
while True:
    filedata = fb.read(1024)
    if not filedata:
        break
    self.__sslclientSocket.send(base64.b64encode(filedata))
    time.sleep(0.1)
fb.close()
time.sleep(0.1)
self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
self.__sslclientSocket.send(self.msgtexthtmltype)
self.__sslclientSocket.send(b'Content-Transfer-Encoding:8bit\r\n\r\n')
msgimgscr = b'<img src="cid:%s">'%randomid.encode('utf-8')
time.sleep(0.1)
self.__sslclientSocket.send(msgimgscr)
time.sleep(0.1)
self.__sslclientSocket.sendall(b'%s' % self.mailcontent.encode('utf-8'))
time.sleep(0.1)
if os.path.isfile(filepath):
    filename = os.path.basename(filepath)
    self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
    self.__sslclientSocket.send(self.msgfileType)
    self.msgfilename = b"Content-Disposition: attachment; filename='%s'\r\n" % filename.encode('utf-
    self.__sslclientSocket.send(self.msgfilename)
    self.__sslclientSocket.send(b'Content-Transfer-Encoding:base64\r\n\r\n')
    self.__sslclientSocket.send(self.msg)
    fb = open(filepath,'rb')
    while True:
        filedata = fb.read(1024)

        if not filedata:
            break
        self.__sslclientSocket.send(base64.b64encode(filedata))
        time.sleep(1)
    fb.close()
```

以上分别是需要发送附件但不需要发送图片和既需要发送附件也需要发送图片的情况。

```
'使用socket套接字连接qq邮箱服务器，并设置ssl验证'
def socketconnet(self):
    print("正在连接服务器……")
    self.__sslclientSocket = ssl.wrap_socket(self.__clientSocket, cert_reqs=ssl.CERT_NONE,
                                             ssl_version=ssl.PROTOCOL_SSLv23)
    self.__sslclientSocket.connect(self.mailserver)
    recv = self.__sslclientSocket.recv(1024).decode('utf-8')
    if recv[:3] != '220':
        print('服务器连接失败：220 reply not received from server.')
        print('正在重试……')
        self.socketconnet()
    print("成功连接服务器……")
    print("正在请求服务器响应……")
    self.__sslclientSocket.send(self.heloCommand)
    recv1 = self.__sslclientSocket.recv(1024).decode('utf-8')
    if recv1[:3] != '250':
        print('服务器响应失败：250 replay not received from server')
        time.sleep(2)
        print('正在重试……')
        self.socketconnet()
    print("成功请求服务器响应……")
```

实现以上操作之前必须要先连接 qq 邮箱服务器，使用 socket 套接字连接，通过

安全套接字层验证。

```
if __name__ == '__main__':
    try:
        sendmail = SendMail()
        sendmail.socketconnet()
        sendmail.login()
        sendmail.quitconnect()

    except Exception:
            print(Exception)
    finally:
        exit(0)
```

最后调用主函数，运行系统。

第三部分 测试结果

1.发送文本邮件

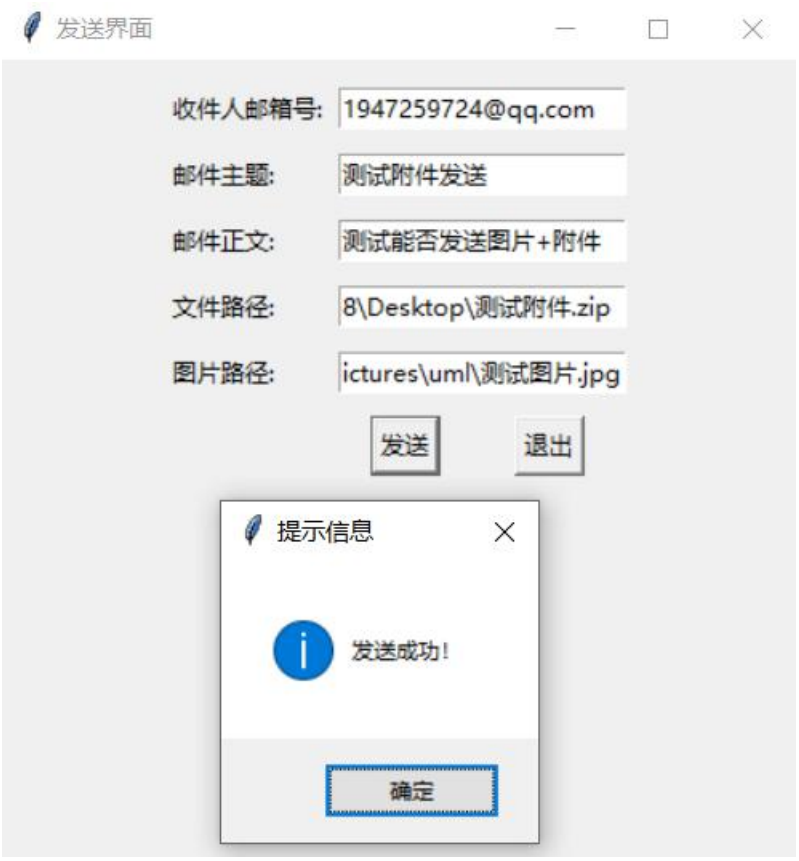
测试能否发送正文

可以看到发送成功，该邮件客服端使用授权码登录，所以并不是直接由发件人 qq 发送，而是由发件人 qq 代发送。

2.发送文本加图片

成功发送正文+图片

3.发送正文+图片+附件


发送界面
收件人邮箱:
1947259724@qq.com
邮件主题:
测试附件发送
邮件正文:
测试能否发送图片+附件
文件路径:
8\Desktop\测试附件.zip
图片路径:
ictures\uml\测试图片.jpg
发送
退出
提示信息
发送成功!
确定


测试附件发送
发件人: <>
(由 769839806@qq.com 代发)
时 间: 2021年6月30日 (星期三) 上午11 : 34
附 件: 1 个 (测试附件.zip)
UML
测试能否发送图片+附件
附件(1 个)
普通附件 (已通过电脑管家云查杀引擎扫描)
测试附件.zip (19K)
预览 下载 收藏 转存

成功发送图片+附件

第四部分 存在的问题

1.图片显示不全问题

刚开始在发送图片的模块测试时，有时图片比较大，然后会出现图片只能显示一部分的情况，原因在于添加图片时，客户端还需要通过路径去寻找图片所在的位置，然后上传，但是程序运行后导致图片只上传部分，其余部分丢失了导致只显示一半或者三分之一，所以在程序块中添加适量的time.sleep后解决了这个问题。

```
fb = open(imgpath, 'rb')
while True:
    filedata = fb.read(1024)
    if not filedata:
        break
    self.__sslclientSocket.send(base64.b64encode(filedata))
    time.sleep(0.1)
fb.close()
time.sleep(0.1)
self.__sslclientSocket.send(b'\r\n\r\n' + self.msgboundary)
self.__sslclientSocket.send(self.msgtexthtmltype)
self.__sslclientSocket.send(b'Content-Transfer-Encoding:8bit\r\n\r\n')
msgimgscr = b'<img src="cid:%s">'%randomid.encode('utf-8')
time.sleep(0.1)
self.__sslclientSocket.send(msgimgscr)
time.sleep(0.1)
self.__sslclientSocket.sendall(b'%s' % self.mailcontent.encode('utf-8'))

time.sleep(0.1)
self.__sslclientSocket.send(self.endmsg)
```

同样的，在添加文件时也有相应操作，通过添加 time.sleep 后可以成功发送。

2.收件人错误问题

在发送邮件时，需要先指定收件人邮箱，然而用户可能存在收件人邮箱输入错误的情况，并且可能是邮箱信息错误，比如输入的 qq 号码不正确，也有可能是格式错误，但这点本客户端无法判断究竟是哪一种错误，所以即使错误，也会提示发送成功，但相应的由于是使用 qq 邮箱代理，所以 qq 邮箱会返回发送失败的信息，本邮件客户端对于本问题尚未解决。

3.发送带有图片或者附件的邮件时，界面未响应的问题。

每一次发送带有附件或者图片的邮件时，由于本客户端系统尚不成熟的原因，添加图片和附件的速度比较慢，所以在发送带有图片或附件的邮件时会出现发送界面未响应的问题，此时是客户端添加文件和图片的过程，但是添加的过程可能和窗口界面发送了一定冲突，所以出现未响应，这是正常情况，需要用户等几秒可以看到发送成功的信息，对于这个问题，初步估计是因为窗口界面函数包含了太多数据的编码解码运算以及获取被编码的数据操作，导致窗口运行缓冲区不够的情况，归根结底就是没有把项目的体系结构设计好，让主窗口函数显得十分冗余，

导致程序效率不高，在后续会逐渐将其完善。