目录

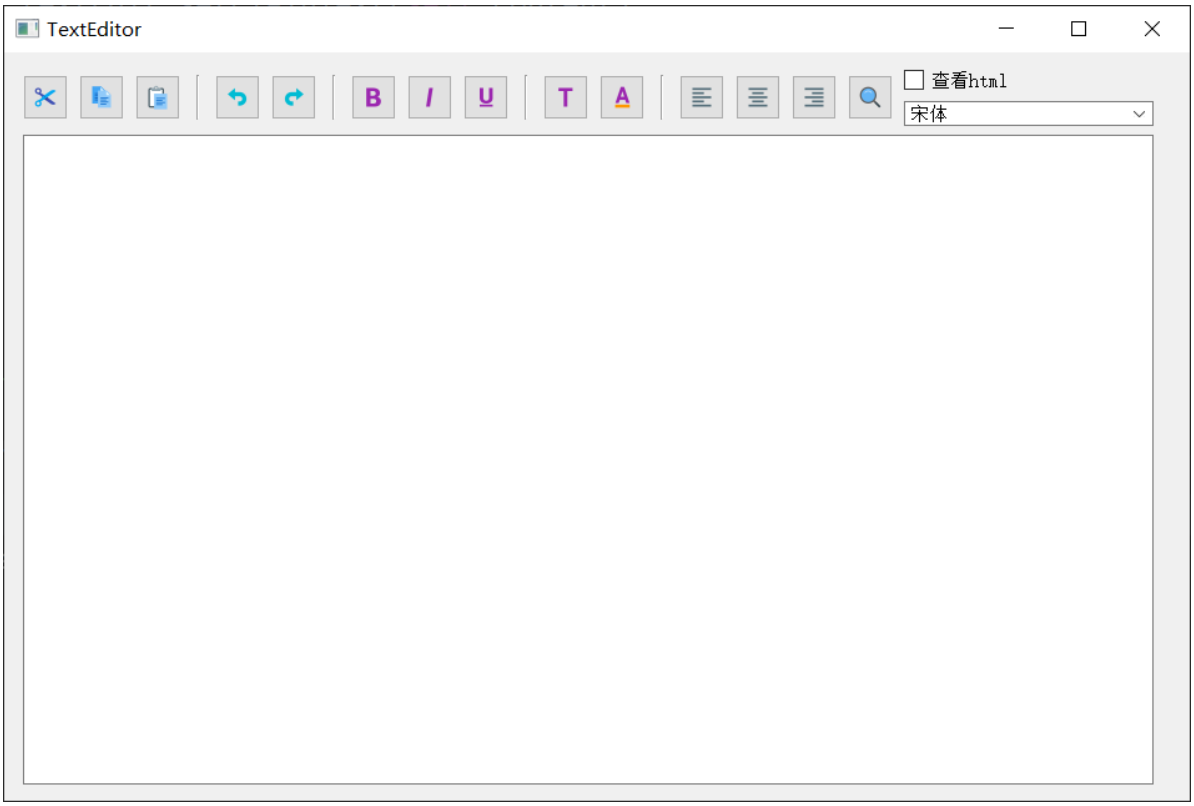
开发过程

一封电子邮件是由标题、邮件内容、发件人邮箱、收件人邮箱等部分构成。实现邮件的发送需要邮件服务器以及相对应的协议，本项目所采用的邮件协议为SMTP协议。SMTP协议属于TCP/IP协议簇，即简单邮件传输协议,它是一组用于由源地址到目的地址传送邮件的规则，由它来控制信件的中转方式，python实现发邮件也是基于此基础上进行封装的。图形用户界面(GUI)开发是基于Python的第三方库PyQt进行的。

内容编辑

UIEditor.py

在设计时，将邮件内容编辑的功能单独设计为一个模块，然后作为一个子窗口嵌套到主窗口中，提高了编辑模块代码的可复用性，保证了以后设计其他窗口时也可以套用这个模块。



在实现文本的复制、粘贴、剪切等功能时，首先考虑的是采用python监听计算机剪贴板的第三方库，例如：

```
#pyperclip模块中的copy() paste()可以向计算机的剪贴板发送文本，或从它接受文本。
# 将程序的输出发送到剪贴板，使它容易粘贴到邮件，文字处理程序或者其他软件中
import pyperclip
pyperclip.copy('hello world')#把hello world 复制到计算机的剪切板
print(pyperclip.paste())#把计算机剪切板的内容粘贴下来
```

```
import win32clipboard as wcb
import win32con as wc

# 打开复制粘贴板
wcb.OpenClipboard()
# 将内容写入复制粘贴板,第二个参数，就是我们要复制的内容，一定要传入字节
wcb.SetClipboardData(wc.CF_TEXT, "Hello World".encode("gbk"))
# 获取粘贴板内容,会返回已经使用Ctrl+C复制得到的内容
data = wcb.GetClipboardData(wc.CF_TEXT)
# 关闭复制粘贴板
wcb.CloseClipboard()
```

但在阅读别人的项目和博客时，我发现PyQt的组件在底层已经实现了复制、粘贴、剪切、字体选择、颜色选择等功能。我们只需要在其功能上进行扩展即可，不必重复造轮子。

```
    def fileCopy(self):
        self.textEdit.copy()

    def fileCut(self):
        self.textEdit.cut()

    def filePaste(self):
        self.textEdit.paste()

    def fileFontBox(self):
        font, okPressed = QFontDialog.getFont()
        if okPressed:
            self.textEdit.setCurrentFont(font)
```

邮件发送

emailSender.py

邮件发送的功能是基于python自带的模块——smtplib和emai，协议采用的是SMTP协议。

smtplib用法相对来说很简单，就是分为两步。

- 创建SMTP的操作对象并连接smtp目标服务器，可以是163、QQ等
- 根据自己的账号登录目标服务器（自己的邮箱地址和邮箱授权码）
- 调用对象中的方法，发送邮件到目标地址

python与smtp服务器之间的具体交互的通用代码：

```
# 采用SSL进行连接时，端口号为465
smtpObj = smtplib.SMTP_SSL(mail_host, 465)
# smtpObj = smtplib.SMTP(mail_host, 25) 采用普通连接的端口号则是25
smtpObj.login(fmail, password)
smtpObj.sendmail(fmail, tmail_list, message.as_string())
```

需要注意的是，当我们与SMTP服务器进行联系时，服务器需要我们的邮箱和授权码(而不是邮箱密码)进行验证。

设置邮件头的内容

```
# 创建邮件
msg = email.message.EmailMessage()
msg['subject'] = subject
msg['from'] = sender + '<' + fmail + '>'
msg['to'] = ','.join(tmail_list)
msg['date'] = email.utils.formatdate(None, True, True)
msg['Content-Type'] = 'text/html'
```

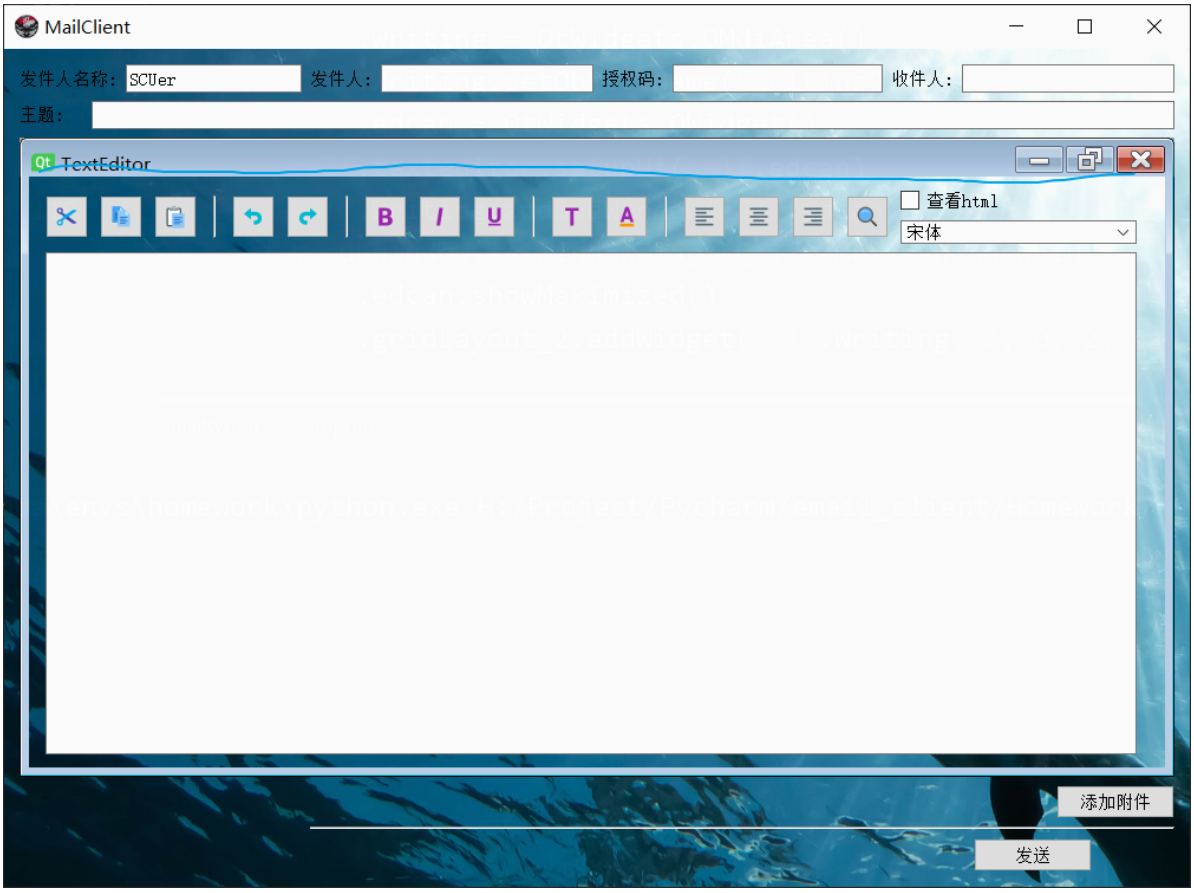
处理附件的添加，采用email的MIMEMultipart类的对象，此模块主要是通过attach方法把以二进制读入的文件传入到邮件的整体内容中。

```
subfile = email.mime.multipart.MIMEMultipart()
subfile['Content-Type'] = 'multipart/related'
for path in pathlist:
    try:
        data = open(path, 'rb')
        except:
            return '附件不存在或为文件夹\n'
        filetype = path.split('.')[-1]
        datafile = email.mime.multipart.MIMEBase(filetype, filetype)
        datafile.set_payload(data.read())
        data.close()
        email.encoders.encode_base64(datafile)
        basename = os.path.basename(path)
        datafile.add_header('Content-Disposition', 'attachment',
filename=basename)
        subfile.attach(datafile)
```

模块整合

UIWriter.py

首先是在主窗口中嵌套TextEditor窗口。

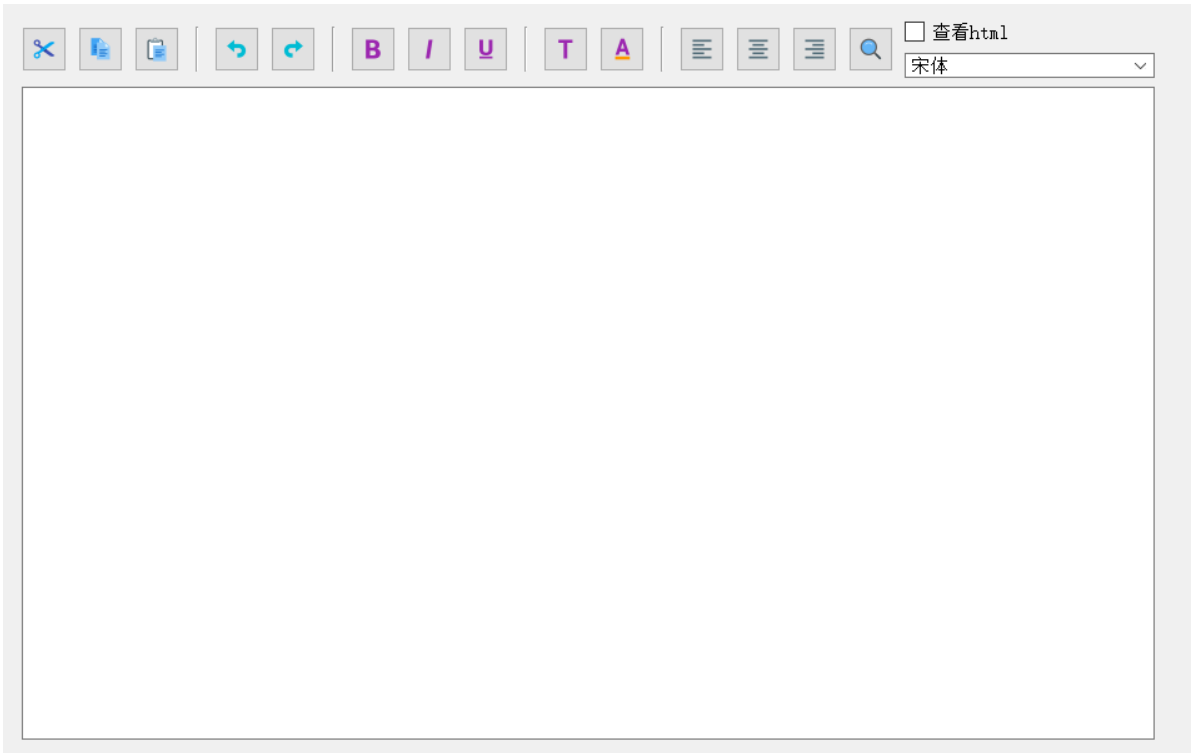


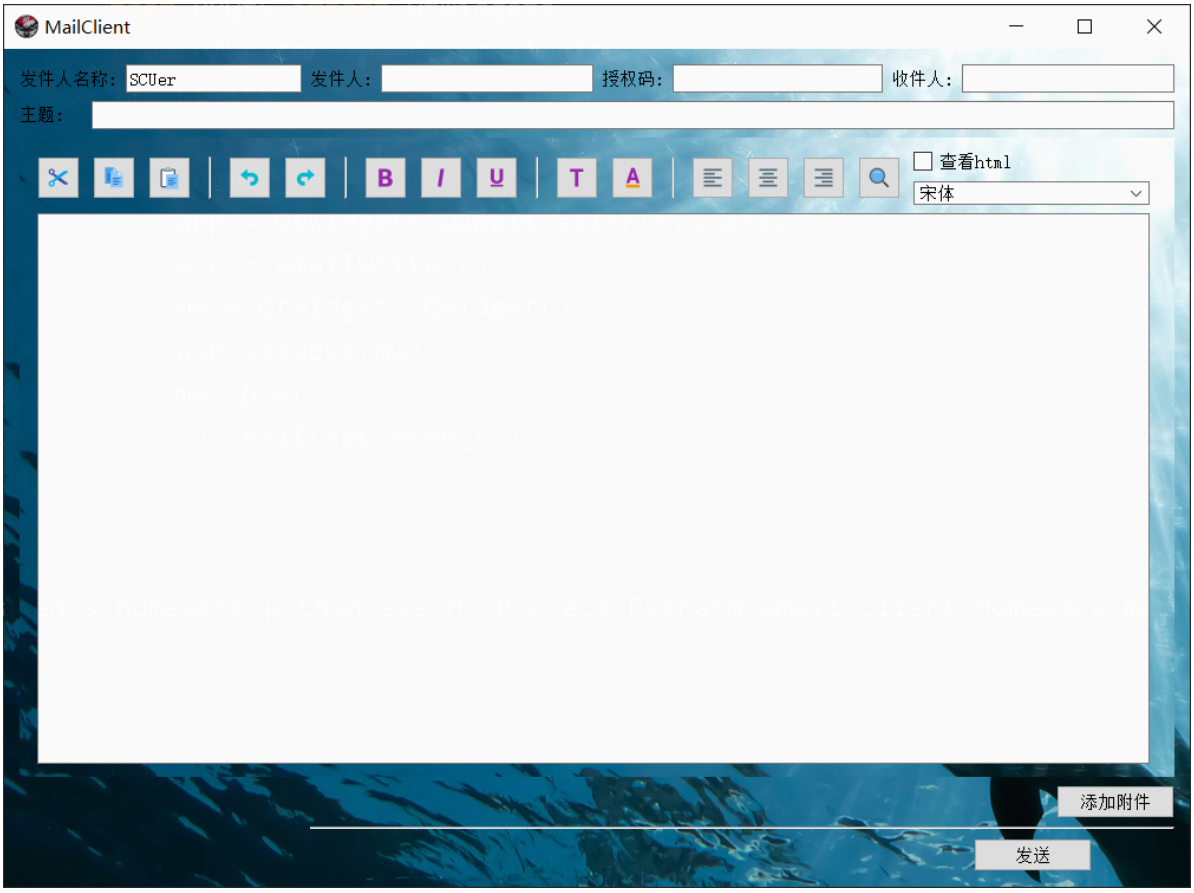
TextEditor窗口虽然成功嵌入主窗口中，但是仍有边框以及菜单栏，可以自由调整大小。

通过阅读博客以及查看项目的源码，可以通过以下代码实现窗口边框的消除

```
subwindow.setWindowFlags(Qt.FramelessWindowHint) #消除窗口边框
```

效果图如下：





通过python自带的正则表达式库re来实现对邮箱格式初步检测，提高代码的健壮性

```
email_regex = re.compile('\\w[-\\w.+]*@([A-Za-z0-9][-A-Za-z0-9]+\\.)+[A-Za-z]
{2,14}')
if not email_regex.findall(self.fmail):
    self.caution(text='发件人邮箱格式不正确呢  ')
    return
```

附件的选择

```
def bappendfileClicked(self):
    # 添加附件的类型
    filetype = "All Files (*);;Text Files (*.txt);;Jpeg (*.jpg);;Png
(*.png);;Doc Files(*.doc);;Pdf Files(*.pdf)"
    msgBox = QtWidgets.QWidget()
    files, ok1 = QFileDialog.getOpenFileNames(msgBox, "多文件选择", "C:/",
filetype)
    for path in files:
        self.bappendfileClickedOne(path)
```

处理BUG

问题1

在实现向多个用户发送邮件时，程序报错

```
 File "/usr/lib/python3.8/email/message.py", line 407, in __setitem__
    raise ValueError("There may be at most {} {} headers "
ValueError: There may be at most 1 To headers in a message
```

在stackoverflow中搜索出了报错原因以及解决方案

python - ValueError: There may be at most 1 To headers in a message - Stack Overflow

> Not knowing the inner working of the EmailMessage class, what I can assume is that every call to `__setitem__` writes to the head of the email message, so by calling it in a loop, the header is being written multiple times, what I'd recommend is that you make an email message for every email you'll send, but create only one server:
>
> ```
> server = smtplib.SMTP(host='smtp.gmail.com', port=587)
> server.starttls()
> server.login("myemail@gmail.com", "mypassword")
> email_list = ["xyz@gmail.com", "abc@gmail.com"]
> for email in email_list:
>     msg = EmailMessage()
>     msg.set_content("Test message.")
>     msg['Subject'] = "Test Subject!!!"
>     msg['From'] = "myemail@gmail.com"
>     msg['To'] = email
>     server.send_message(msg)
> server.quit()
> ```
>
> Only if you need for the messages to be sent **separately**. If you want to send the same message to everyone at the same time you could do something like
>
> ```
> msg['To'] = ', '.join(email_list)
> ```

根据分析，导致报错的原因可能是——在循环中，多次对msg['to']进行引用和修改。

原来的代码如下：

```
msg['to'] = tmail_list[0]
for tmail in tmail_list:
    if not msg['to'].find(tmail):
        msg['to'] += ',' + tmail
```

解决方案:

```
msg['To'] = ', '.join(tmail_list)
```

先调用python字符串的join方法拼接字符串，再将拼接好的字符串赋值给msg['To']

问题2

在实现文件读入时，程序报错

```
file = open('z/img/background.png', 'r')
print(file.read())
```

```
Traceback (most recent call last):
  File "H:/Project/Pycharm/email_client/test.py", line 3, in <module>
    print(file.read())
UnicodeDecodeError: 'gbk' codec can't decode byte 0xff in position 0: illegal
multibyte sequence
```

在网上查询后，发现图片等文件非纯文本数据只能通过字节流读写。

字符流按字符(一个字符占两个字节)读数据：一次读两个字节，返回了这两个字节所对应的字符的int型数值(编码)。写入文件时把这两个字节的内容解码成这个字符在Unicode码下对应的二进制数据写入。即把原始文件中的二进制数据以字符形式读出,再将字符以二进制形式写入,所以得到的文件以字符方式存储。字符流只能够处理纯文本数据。'r'以字符流的方式打开文件，而'rb'以二进制格式即字符流的方式打开文件。

将读入模式切换为'rb'后，程序运行成功。此时，代码不仅可以处理文本文件，还能处理非文本文件。

```
file = open('z/img/background.png', 'rb')
print(file.read())
```

但是，如果仅仅只是处理文本文件，字符流('r')处理文本要比字节流('rb')处理文本要方便

分析与总结

在编写项目代码中，我们学会了smtplib、email库的基本用法，同时也学习了如何使用PyQt库构建一个具有一定功能的窗口程序，了解了SMTP协议是如何运行以及一封电子邮件所具有的属性。

同时，本项目还有许多不足之处，代码还有不少需要优化的地方，例如没有实现账号登录功能、实现通讯录功能等。