Python Gmail Email Program OOP

Contents

Time required: 30 minutes

Let's learn how to send email using Python and Gmail. Being able to send an email from Python opens up all sorts of possibilities.

Setting up a Gmail Account for Email

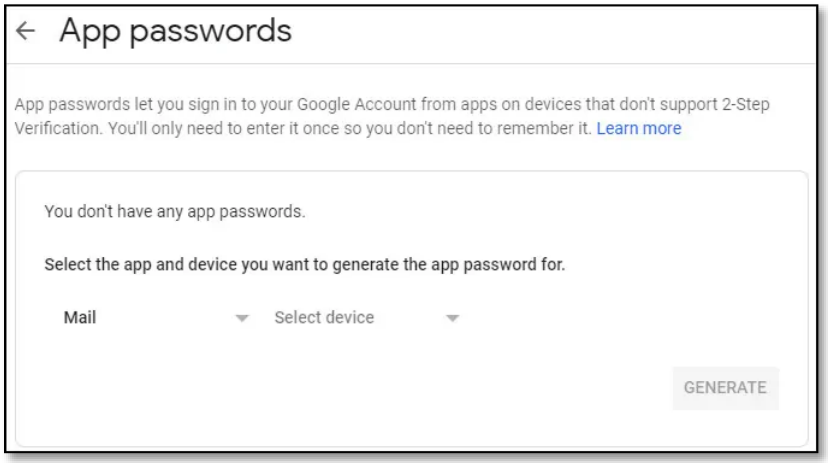
If you don't have a Gmail account, you will want to create one.

Let's enable your Gmail account to receive connections from external programs, like Python.

1. Open your browser and access your Gmail account.
2. On the login screen → enter your Gmail username and password.
3. After the login → access the following URL: https://myaccount.google.com/signinoptions/two-step-verification
4. Enable the two-step verification on this account.

Create an Application Password

1. Access the following URL: https://security.google.com/settings/security/apppasswords
2. Select Gmail application and the type of device: **Other**.



App passwords

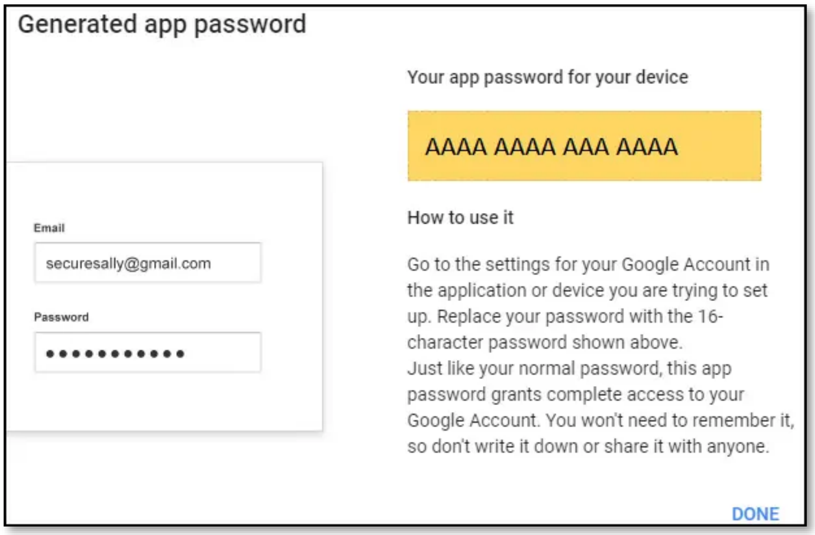
App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. Learn more

You don't have any app passwords.

Select the app and device you want to generate the app password for.

3. Name the device: **Python**

4. Click on the Generate button and take note of the randomly generated password.



Generated app password

Email: securesally@gmail.com

Password: ●●●●●●●●●●●

Your app password for your device

AAAA AAAA AAA AAAA

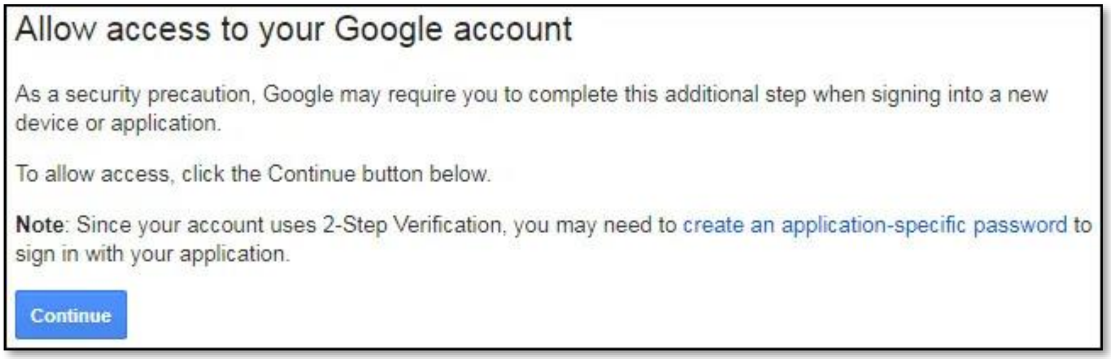
How to use it

Go to the settings for your Google Account in the application or device you are trying to set up. Replace your password with the 16-character password shown above.
Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone.

5. Access the following URL:

6. https://accounts.google.com/DisplayUnlockCaptcha

7. Click on the **Continue** button to enable external access to your Google account.



Allow access to your Google account

As a security precaution, Google may require you to complete this additional step when signing into a new device or application.

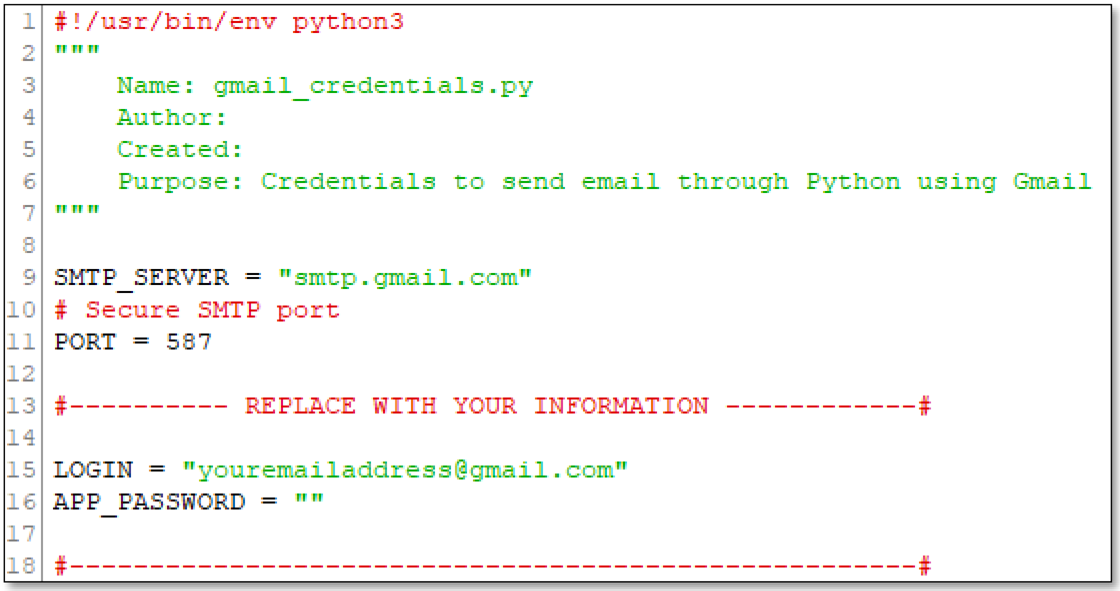
To allow access, click the Continue button below.

Note: Since your account uses 2-Step Verification, you may need to create an application-specific password to sign in with your application.

You have finished the required steps for the Gmail integration.

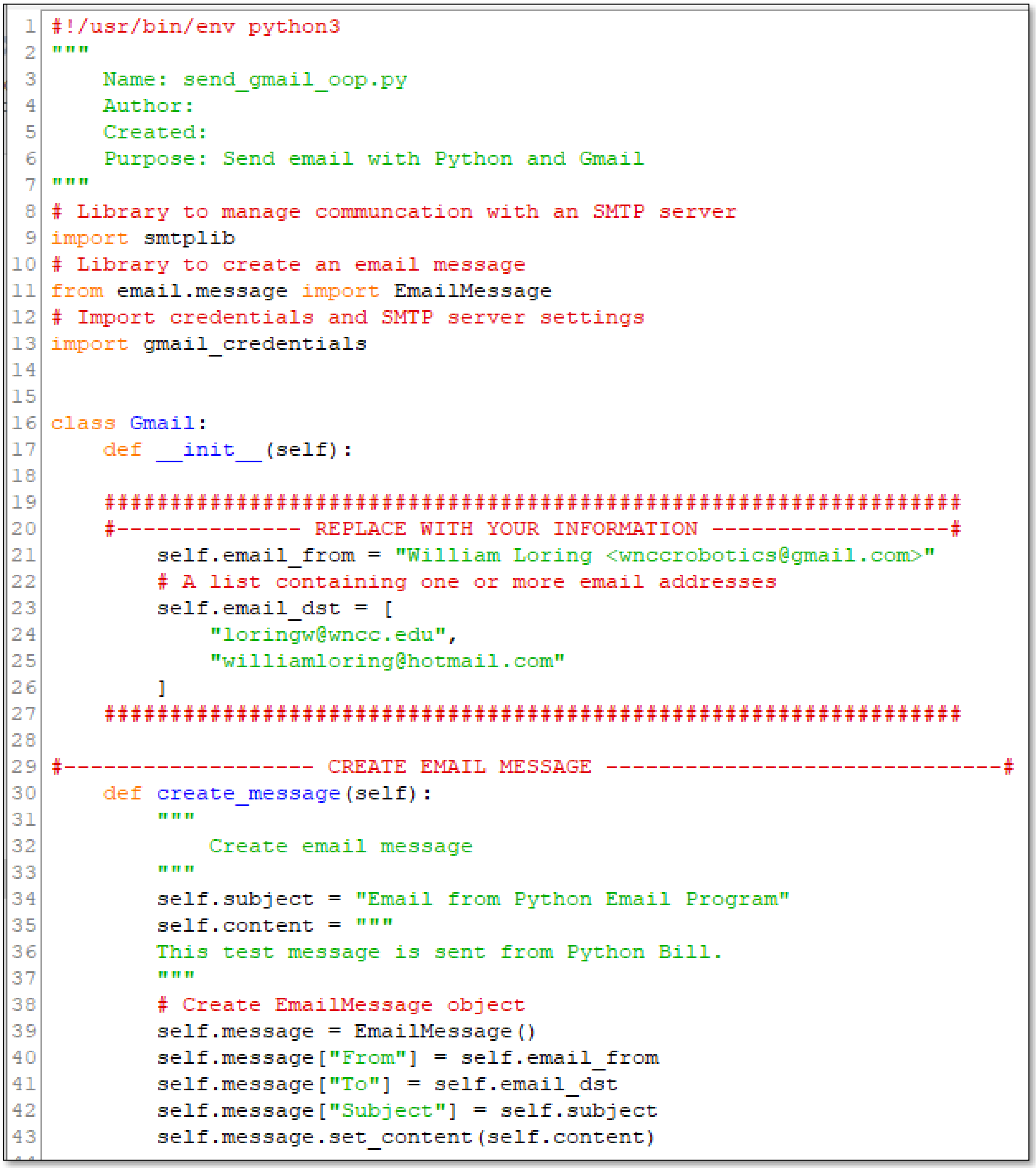
Tutorial 1 - Send Email with Python

Create a Python program named: **gmail_credentials.py**

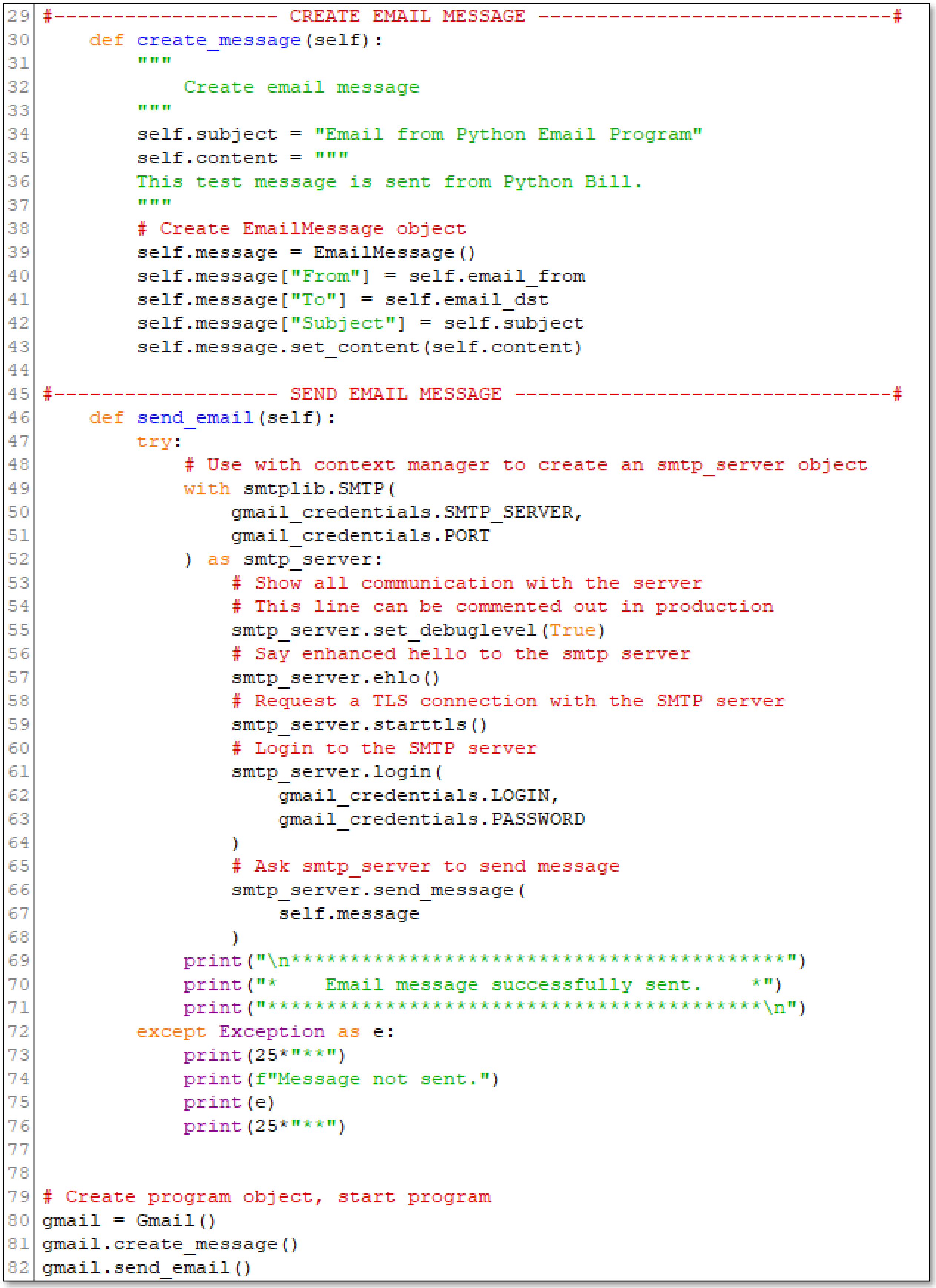


```
#!/usr/bin/env python3
"""
    Name: gmail_credentials.py
    Author:
    Created:
    Purpose: Credentials to send email through Python using Gmail
"""

SMTP_SERVER = "smtp.gmail.com"
# Secure SMTP port
PORT = 587

#---------- REPLACE WITH YOUR INFORMATION ------------#

LOGIN = "youremailaddress@gmail.com"
APP_PASSWORD = ""

#-----------------------------------------------------#
```

Create a Python program named: **send_gmail.py**



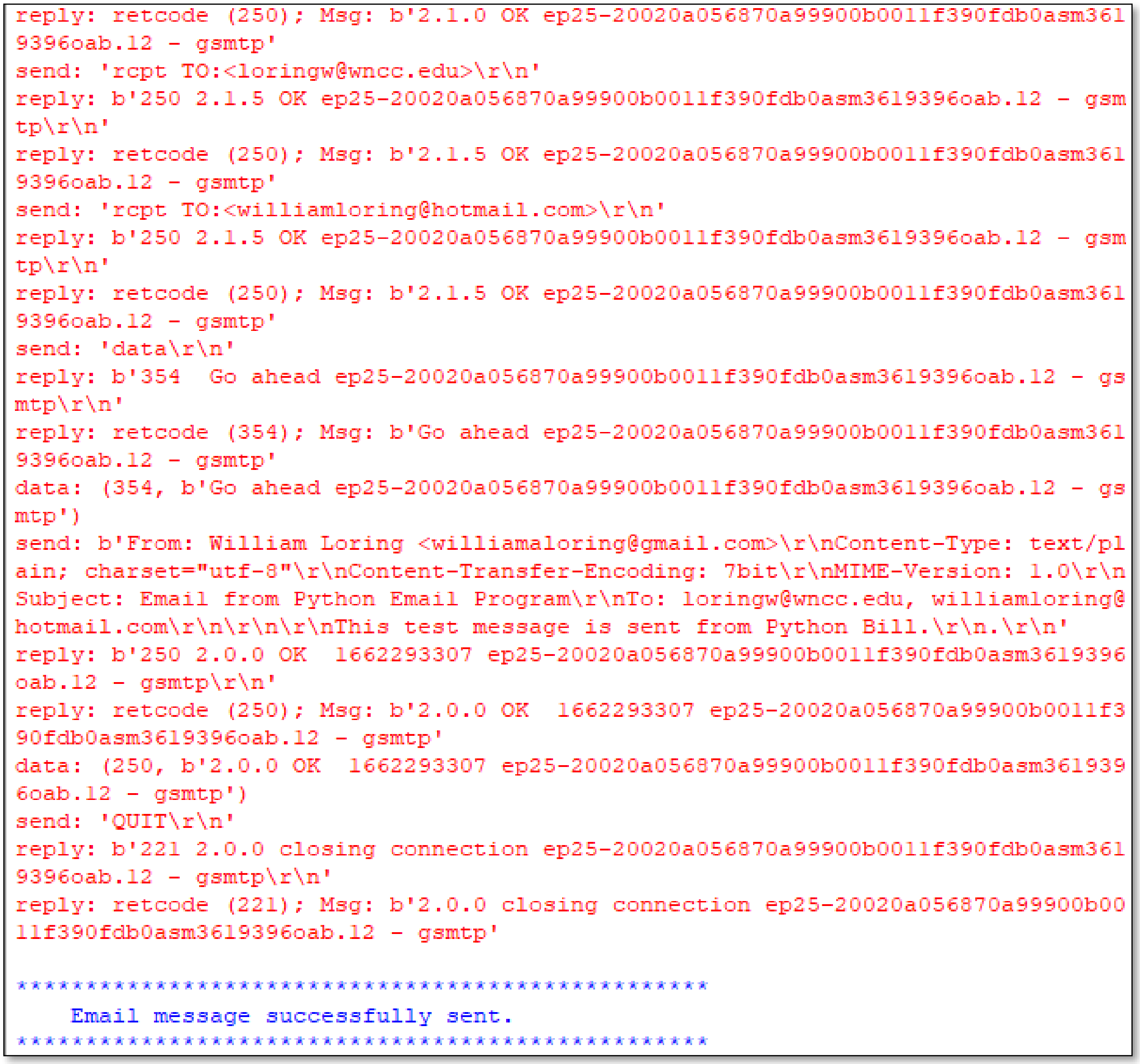
```
#!/usr/bin/env python3
"""
    Name: send_gmail_oop.py
    Author:
    Created:
    Purpose: Send email with Python and Gmail
"""
# Library to manage communcation with an SMTP server
import smtplib
# Library to create an email message
from email.message import EmailMessage
# Import credentials and SMTP server settings
import gmail_credentials


class Gmail:
    def __init__(self):

    ##################################################################
    #-------------- REPLACE WITH YOUR INFORMATION -------------------#
        self.email_from = "William Loring <wnccrobotics@gmail.com>"
        # A list containing one or more email addresses
        self.email_dst = [
            "loringw@wncc.edu",
            "williamloring@hotmail.com"
        ]
    ##################################################################

#------------------- CREATE EMAIL MESSAGE -------------------------------#
    def create_message(self):
        """
            Create email message
        """
        self.subject = "Email from Python Email Program"
        self.content = """
        This test message is sent from Python Bill.
        """
        # Create EmailMessage object
        self.message = EmailMessage()
        self.message["From"] = self.email_from
        self.message["To"] = self.email_dst
        self.message["Subject"] = self.subject
        self.message.set_content(self.content)
```



```python
#-------------------- CREATE EMAIL MESSAGE ------------------------------#
    def create_message(self):
        """
            Create email message
        """
        self.subject = "Email from Python Email Program"
        self.content = """
        This test message is sent from Python Bill.
        """
        # Create EmailMessage object
        self.message = EmailMessage()
        self.message["From"] = self.email_from
        self.message["To"] = self.email_dst
        self.message["Subject"] = self.subject
        self.message.set_content(self.content)

#-------------------- SEND EMAIL MESSAGE --------------------------------#
    def send_email(self):
        try:
            # Use with context manager to create an smtp_server object
            with smtplib.SMTP(
                gmail_credentials.SMTP_SERVER,
                gmail_credentials.PORT
            ) as smtp_server:
                # Show all communication with the server
                # This line can be commented out in production
                smtp_server.set_debuglevel(True)
                # Say enhanced hello to the smtp server
                smtp_server.ehlo()
                # Request a TLS connection with the SMTP server
                smtp_server.starttls()
                # Login to the SMTP server
                smtp_server.login(
                    gmail_credentials.LOGIN,
                    gmail_credentials.PASSWORD
                )
                # Ask smtp_server to send message
                smtp_server.send_message(
                    self.message
                )
            print("\n******************************************")
            print("*    Email message successfully sent.    *")
            print("******************************************\n")
        except Exception as e:
            print(25*"***")
            print(f"Message not sent.")
            print(e)
            print(25*"***")


# Create program object, start program
gmail = Gmail()
gmail.create_message()
gmail.send_email()
```

The example run contains all the chatter back and forth between your Python program and the mail server. The debug level is set to True so we can see how much background communication occurs sending a simple email message. The debug level line can be commented out.

Example run:



```
reply: retcode (250); Msg: b'2.1.0 OK ep25-20020a056870a99900b0011f390fdb0asm361
9396oab.12 - gsmtp'
send: 'rcpt TO:<loringw@wncc.edu>\r\n'
reply: b'250 2.1.5 OK ep25-20020a056870a99900b0011f390fdb0asm3619396oab.12 - gsm
tp\r\n'
reply: retcode (250); Msg: b'2.1.5 OK ep25-20020a056870a99900b0011f390fdb0asm361
9396oab.12 - gsmtp'
send: 'rcpt TO:<williamloring@hotmail.com>\r\n'
reply: b'250 2.1.5 OK ep25-20020a056870a99900b0011f390fdb0asm3619396oab.12 - gsm
tp\r\n'
reply: retcode (250); Msg: b'2.1.5 OK ep25-20020a056870a99900b0011f390fdb0asm361
9396oab.12 - gsmtp'
send: 'data\r\n'
reply: b'354  Go ahead ep25-20020a056870a99900b0011f390fdb0asm3619396oab.12 - gs
mtp\r\n'
reply: retcode (354); Msg: b'Go ahead ep25-20020a056870a99900b0011f390fdb0asm361
9396oab.12 - gsmtp'
data: (354, b'Go ahead ep25-20020a056870a99900b0011f390fdb0asm3619396oab.12 - gs
mtp')
send: b'From: William Loring <williamaloring@gmail.com>\r\nContent-Type: text/pl
ain; charset="utf-8"\r\nContent-Transfer-Encoding: 7bit\r\nMIME-Version: 1.0\r\n
Subject: Email from Python Email Program\r\nTo: loringw@wncc.edu, williamloring@
hotmail.com\r\n\r\n\r\nThis test message is sent from Python Bill.\r\n.\r\n'
reply: b'250 2.0.0 OK  1662293307 ep25-20020a056870a99900b0011f390fdb0asm3619396
oab.12 - gsmtp\r\n'
reply: retcode (250); Msg: b'2.0.0 OK  1662293307 ep25-20020a056870a99900b0011f3
90fdb0asm3619396oab.12 - gsmtp'
data: (250, b'2.0.0 OK  1662293307 ep25-20020a056870a99900b0011f390fdb0asm361939
6oab.12 - gsmtp')
send: 'QUIT\r\n'
reply: b'221 2.0.0 closing connection ep25-20020a056870a99900b0011f390fdb0asm361
9396oab.12 - gsmtp\r\n'
reply: retcode (221); Msg: b'2.0.0 closing connection ep25-20020a056870a99900b00
11f390fdb0asm3619396oab.12 - gsmtp'

**************************************************
    Email message successfully sent.
**************************************************
```

Test your program.

Assignment 1 - Python CLI Email Program

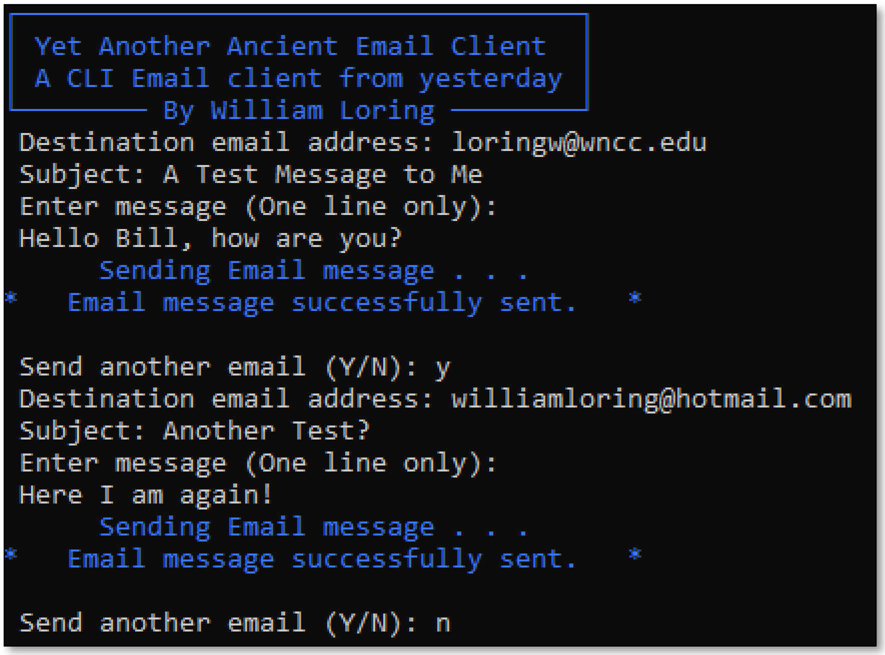
Way back in prehistoric times, we used CLI (command line interface) email programs like pine. Everything was on Unix systems, not a GUI in sight.

Let’s build our own blast from the past email program.

Requirements

1. Add a creative program title
2. Add input for
   a. Destination email
   b. Subject
   c. Message
3. Allow user to send another message

Example run:



```
  Yet Another Ancient Email Client
  A CLI Email client from yesterday
                By William Loring
Destination email address: loringw@wncc.edu
Subject: A Test Message to Me
Enter message (One line only):
Hello Bill, how are you?
     Sending Email message . . .
*   Email message successfully sent.   *

Send another email (Y/N): y
Destination email address: williamloring@hotmail.com
Subject: Another Test?
Enter message (One line only):
Here I am again!
     Sending Email message . . .
*   Email message successfully sent.   *

Send another email (Y/N): n
```

Yes, you have successfully created a money making app, a CLI email program in Python!

Assignment Submission

- Send the instructor an email at loringw@wncc.edu

- Insert a screenshot of the program run
- Attach the completed program files
- Submit the assignment in Blackboard.