Emails with Python

PIERIAN DATA

Complete Python Bootcamp

- In this section we will explore how to send emails with Python and how to check our inbox for received messages.
- Please keep in mind this process is highly reliant on admin privileges on both your local computer, your internet, and your email.

Complete Python Bootcamp

- It is highly likely that on a corporate network, work computer, or work email these methods will be blocked for security reasons.
- If you encounter issues due to this, please contact your IT department, as it is not an issue we can fix on our end.

Complete Python Bootcamp

- Lastly, there is no full exercise for these lecture topics, since there is no real way we can create true self-assessments for a personal email address.
- However, we do provide a notebook with some ideas for you to explore, we encourage you to get creative!

Let’s get started!

Sending Emails

Complete Python Bootcamp

- To send emails with Python, we need to manually go through the steps of connecting to an email server, confirming connection, setting a protocol, logging on, and sending the message.

Complete Python Bootcamp

- Fortunately the built-in smtplib library in Python makes these steps simple function calls.

Complete Python Bootcamp

- Each major email provider has their own SMTP (Simple Mail Transfer Protocol) Server.

Provider	SMTP server domain name
Gmail (will need App Password)	smtp.gmail.com
Yahoo Mail	smtp.mail.yahoo.com
Outlook.com/Hotmail.com	smtp-mail.outlook.com
AT&T	smpt.mail.att.net (Use port 465)
Verizon	smtp.verizon.net (Use port 465)
Comcast	smtp.comcast.net

Complete Python Bootcamp

- We will go over this process with a Gmail account.
- For gmail users, you will need to generate an app password instead of your normal password.
- This let's Gmail know that the Python script attempting to access your account is authorized by you.

Complete Python Bootcamp

- Let’s explore this entire process.

Receiving Emails

Complete Python Bootcamp

- To view received emails with Python we can use the built in imaplib and email libraries in Python.
- The imaplib library has a special syntax for searching your Inbox.

Complete Python Bootcamp

Keyword	Definition
'ALL'	Returns all messages in your email folder. Often there are size limits from imaplib. To change these use imaplib._MAXLINE = 100 , where 100 is whatever you want the limit to be.
'BEFORE date'	Returns all messages before the date. Date must be formatted as 01-Nov-2000.
'ON date'	Returns all messages on the date. Date must be formatted as 01-Nov-2000.
'SINCE date'	Returns all messages after the date. Date must be formatted as 01-Nov-2000.
'FROM some_string '	Returns all from the sender in the string. String can be an email, for example 'FROM user@example.com' or just a string that may appear in the email, "FROM example"
'TO some_string'	Returns all outgoing email to the email in the string. String can be an email, for example 'FROM user@example.com' or just a string that may appear in the email, "FROM example"
'CC some_string' and/or 'BCC some_string'	Returns all messages in your email folder. Often there are size limits from imaplib. To change these use imaplib._MAXLINE = 100 , where 100 is whatever you want the limit to be.
'SUBJECT string','BODY string','TEXT "string with spaces"'	Returns all messages with the subject string or the string in the body of the email. If the string you are searching for has spaces in it, wrap it in double quotes.
'SEEN', 'UNSEEN'	Returns all messages that have been seen or unseen. (Also known as read or unread)
'ANSWERED', 'UNANSWERED'	Returns all messages that have been replied to or unreplied to.
'DELETED', 'UNDELETED'	Returns all messages that have been deleted or that have not been deleted.

Complete Python Bootcamp

- Before beginning this discussion, send yourself a test email with a unique subject line that you will be able to remember and search for using Python.
- Let’s begin exploring checking received emails with Python.