The Python installed on your Raspberry Pi includes libraries that can be used to email messages. This could be useful for many applications. For example, suppose you are building an alarm system. You connect a switch to your front door. When someone opens the door, your Raspberry Pi takes his or her picture (with the Pi camera) and then sends a copy of the photo to you in an email attachment.

This lesson provides the Python code for sending an email with attachment. I suggest you create a new email account on Gmail for this lesson. When you are done with the lesson, you can continue to use this email account for IoT projects.

It is necessary for the sending Gmail account to have lower security settings. That is one reason why you should not use your regular email account to send messages by a Python script.

After you create your new email account, you need to open your account settings (click the circle near upper right of page, then click on the **My Account** button). On the My Account page that opens, find the **Sign-in & security** section on the left side. Click on "**Apps with account access**." Under the heading of Apps with account access, find the section labeled "**Allow less secure apps**." This will probably be set to OFF. With the mouse, move the slider button to the right to turn it ON. Now your new email account is configured properly to work with Python email code.

The code for sending an email is provided below. Use the Python version 2 editor to write the code. The code is missing some elements that you must add: sender email address, recipient email address, password of sender. If you are including an attached file, then you need to include the path to the file and the file name in the places specified in the code. In addition, you may wish to edit the subject heading and body of the email message.

```
# Python code to illustrate Sending mail with attachments
# from your Gmail account
#fill in fromaddr and toaddr and password of sender in appropriate places below
#in the line starting with attachment, fill in the path to the file you wish to attach
#comment out the attachment code if you don't want to send an attachment.


# libraries to be imported

import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
```

```
from email.mime.base import MIMEBase
from email import encoders


fromaddr = "EMAIL address of the sender"
toaddr = "EMAIL address of the receiver"

# instance of MIMEMultipart

msg = MIMEMultipart()


# storing the senders email address

msg['From'] = fromaddr


# storing the receivers email address

msg['To'] = toaddr


# storing the subject

msg['Subject'] = "Python generated with attachment"


# string to store the body of the mail

body = "This message was generated by a Python script running on GEAR7 Raspberry Pi"
```

```
# attach the body with the msg instance
msg.attach(MIMEText(body, 'plain'))

# open the file to be sent
filename = "email.png"  #File name with correct extension
attachment = open("/home/pi/Pictures/email.png", "rb")

# instance of MIMEBase and named as p
p = MIMEBase('application', 'octet-stream')

# To change the payload into encoded form
p.set_payload((attachment).read())

# encode into base64
encoders.encode_base64(p)

p.add_header('Content-Disposition', "attachment; filename= %s" % filename)

# attach the instance 'p' to instance 'msg'
msg.attach(p)
```

```
# creates SMTP session
s = smtplib.SMTP('smtp.gmail.com', 587)

# start TLS for security
s.starttls()

# Authentication
s.login(fromaddr, "Password_of_the_sender")

# Converts the Multipart msg into a string
text = msg.as_string()

# sending the mail
s.sendmail(fromaddr, toaddr, text)

# terminating the session
s.quit()
```