Chapter 2: Sending Email

Summary:
As you've probably guessed after seeing the title of this chapter, we're now going to take a look at how to send emails with Python. If you've ever done any networking before, this process will seem like a walk in the park--which it is, thanks to Python.

To send emails with Python, we'll use the smtplib package that comes installed with Python. SMTP stands for "Simple Mail Transfer Protocol" and is how email is generally sent across the interweb. The way we send email is using a socket connection to the SMTP port for our email service (i.e. Gmail, Yahoo, Hotmail, etc.). The default port to connect to for SMTP is the TCP (Transmission Control Protocol) port, 25. However, mail submission is done through port 587, which we'll be using. If you are more interested in networking, I would suggest looking into the socket package of Python.

To dive right into things, let's take a look at this simple command-line interface (CLI) for sending an email:

Example 2-1 (email_example.py):

```
import smtplib
from getpass import getpass
from email.mime.text import MIMEText

server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
```

```
username = input("Enter your email address: ")
password = getpass(prompt="Enter your password: ")
server.login(username, password)

recipient = input("\nEnter Recipient Email Address: ")
subject = input("Enter Subject: ")
msg = input("\nEnter message: ")

mail = MIMEText(msg)
mail['Subject'] = subject
mail['From'] = username
mail['To'] = recipient

# Send mail
server.send_message(mail)
server.quit()
---
```

First, we import the needed tools such as the obvious, smtplib. We also make use of the getpass module of Python for retrieving the password from the user more appropriately. Lastly, the MIMEText class that we import will be used to construct our mail object with subject, message, etc.

After the imports are done, we can create the SMTP connection and login. The SMTP class takes two parameters here: the host and port to connect to; in this case, we are using Google's gmail service in particular. After creating this, a simple call to starttls() which tells our connection to encrypt the sent messages for safer communication.

Next, we need to login to our Gmail account, so we get the username and password from the user and login. Using getpass() we can get the password without showing the characters entered by the user.

Before constructing our MIMEText object, we need to have the user enter the necessary information. So we get the recipient's email address, the subject line, and the message itself to be sent. Finally, we can build the MIMEText object and set it's headers appropriately. Note: you can actually send mail without using the MIMEText object; however, it is more verbose, and you must use other MIME objects for different media such as images.

Then, all that's left is to hit send! So we call send_message() and quit() which sends the email and logs out, closing our socket connection.

Wrap Up:
There you have it. You can now send emails to your hearts content using Python. Of course, you can delve further into this topic if you like. The best place to start is of course the Python documentation on the smtplib package. From there you can learn how to retrieve mail from your inbox, send images and other attachments, etc. I developed a graphical user interface (gmailer.py) for this program using Tkinter for Python (which we will discuss later), and can be found with the other source code provided with this book.