Introduction to Python Email Library

Email messages look simple in an email client. But behind the scenes the client is doing a lot of work to make that happen! Email messages -- even messages with images and attachments -- are actually complicated text structures made entirely of readable strings!

The ***Simple Mail Transfer Protocol (SMTP)*** and ***Multipurpose Internet Mail Extensions (MIME)*** standards define how email messages are constructed. You *could* read the standards documentation and create email messages all on your own, but you don't need to go to all that trouble. The email built-in Python module lets us easily construct email messages.

We'll start by using the email.message.EmailMessage class to create an empty email message.

```
>>> from email.message import EmailMessage
>>> message = EmailMessage()
>>> print(message)
```

As usual, printing the message object gives us the string representation of that object. The email library has a function that converts the complex EmailMessage object into something that is fairly human-readable. Since this is an empty message, there isn't anything to see yet. Let's try adding the sender and recipient to the message and see how that looks.

We'll define a couple of variables so that we can reuse them later.

```
>>> sender = "me@example.com"
>>> recipient = "you@example.com"
```

And now, add them to the From and To fields of the message.

```
>>> message['From'] = sender
>>> message['To'] = recipient
>>> print(message)
From: me@example.com
To: you@example.com
```

Cool! That's starting to look a bit more like an email message now. How about a subject?

```
>>> message['Subject'] = 'Greetings from {} to {}!'.format(sender, recipient)
>>> print(message)
From: me@example.com
To: you@example.com
Subject: Greetings from me@example.com to you@example.com!
```

From, **To**, and **Subject** are examples of ***email header fields***. They’re ***key-value pairs*** of labels and instructions used by email clients and servers to route and display the email. They’re separate from the email's ***message body***, which is the main content of the message.

Let's go ahead and add a message body!

```
>>> body = """Hey there!
...
... I'm learning to send emails using Python!"""
>>> message.set_content(body)
```

Alright, now what does that look like?

```
>>> print(message)
From: me@example.com
To: you@example.com
Subject: Greetings from me@example.com to you@example.com!
MIME-Version: 1.0
Content-Type: text/plain; charset="utf-8"
Content-Transfer-Encoding: 7bit

Hey there!

I'm learning to send email using Python!
```

The message has a body! The **set_content()** method also automatically added a couple of headers that the email infrastructure will use when sending this message to another machine. Remember in an earlier course, when we talked about ***character encodings***? The ***Content-Type*** and ***Content-Transfer-Encoding*** headers tell email clients and servers how to interpret the bytes in this email message into a string. Now, what about this other header? What is MIME? We'll learn about that next!