Sending Emails from Python

Reading: Module 3 Introduction
10 min

Discussion Prompt: Discussion Prompt
10 min

Reading: Introduction to Python Email Library
10 min

Reading: Adding Attachments
10 min

Reading: Sending the Email Through an SMTP Server
10 min

Generating PDFs from Python

Module 3 Project

Adding Attachments

Remember, email messages are made up completely of strings. When you add an attachment to an email, whatever type the attachment happens to be, it's encoded as some form of text. The ***Multipurpose Internet Mail Extensions (MIME)*** standard is used to encode all sorts of files as text strings that can be sent via email.

Let's dive in and break down how that works.

In order for the recipient of your message to understand what to do with an attachment, you need to label the attachment with a ***MIME type*** and ***subtype*** to tell them what sort of file you're sending. The ***Internet Assigned Numbers Authority (IANA)*** (iana.org) hosts a registry of valid MIME types. If you know the correct type and subtype of the files you'll be sending, you can use those values directly. If you don't know, you can use the Python **mimetypes** module to make a good guess!

```
>>> import os.path
>>> attachment_path = "/tmp/example.png"
>>> attachment_filename = os.path.basename(attachment_path)
>>> import mimetypes
>>> mime_type, _ = mimetypes.guess_type(attachment_path)
>>> print(mime_type)
image/png
```

Alright, that **mime_type** string contains the MIME type and subtype, separated by a slash. The **EmailMessage** type needs a MIME type and subtypes as separate strings, so let's split this up:

```
>>> mime_type, mime_subtype = mime_type.split('/', 1)
>>> print(mime_type)
image
>>> print(mime_subtype)
png

```

Now, finally! Let's add the attachment to our message and see what it looks like.

```
>>> with open(attachment_path, 'rb') as ap:
...     message.add_attachment(ap.read(),
...                            maintype=mime_type,
...                            subtype=mime_subtype,
...                            filename=os.path.basename(attachment_path))
...
>>> print(message)
Content-Type: multipart/mixed; boundary="===============5350123048127315795=="

--===============5350123048127315795==
Content-Type: text/plain; charset="utf-8"
Content-Transfer-Encoding: 7bit

Hey there!

I'm learning to send email using Python!

--===============5350123048127315795==
Content-Type: image/png
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="example.png"
MIME-Version: 1.0

iVBORw0KGgoAAAANSUhEUgAAASIAAABSCAYAAADw69nDAAAACXBIWXMAAAsTAAALEwEAmpwYAAAg
AElEQVR4nO2dd3wUZf7HP8/M9k2nKIJA4BCUNJKgNJWIBUUgEggCiSgeVhA8jzv05Gc5z4KHiqin
eBZIIBDKIXggkKIeCRCAhjQAqx4UiCARSt83uzDy/PzazTDZbwy4BnHde+9qZydNn97Pf5/uUIZRS
(...We deleted a bunch of lines here...)
wgAAAABJRU5ErkJggg==

--===============5350123048127315795==--

```

The entire message can still be serialized as a text string, including the image that we attached! The email message as a whole has the MIME type "multipart/mixed". Each ***part*** of the message has its own MIME type. The message body is still there as a "text/plain" part, and the image attachment is a "image/png" part. Cool!

Now, how do we *send* this email message? That's coming up!

Mark as completed

Like **Dislike** **Report an issue**