5.1 构造和解析电子邮件实战

即使是在QQ、微信等各种即时通信软件已经成为主流应用的现在，电子邮件（E-mail）仍是一种重要的互联网应用，尤其是与工作相关的场景。

电子邮件发送后暂存于服务器中，收件人可在任意时间接收和处理。电子邮件由信封（envelope）和内容（content）两部分组成，信封主要为电子邮件传输程序提供地址信息，与现实世界中邮局投递邮件的过程类似。

5.1.1 标准库email常用函数

本节重点演示如何使用Python标准库email创建和解析电子邮件对象，后面几节通过实战案例演示如何使用smtplib、poplib、imaplib等标准库发送、接收和处理电子邮件，相关协议的原理和细节请参考《计算机网络》之类的书籍或官方文档。

Python标准库email提供了构造和解析电子邮件所需要的全部功能，电子邮件对象是一个树状结构，其中每个节点都提供了MIME（Multipurpose Internet Mail Extensions）接口。标准库email顶层函数（可以通过email直接调用的函数）message_from_bytes()、message_from_string()、message_from_file()、message_from_binary_file()可以根据字符串、字节串或文件的内容快速创建EmailMessage电子邮件对象，也可以使用模块email.message中的Message、MIMEPart和EmailMessage类手工创建电子邮件对象（MIMEPart是Message的派生类，EmailMessage是MIMEPart的派生类，重写了set_content()方法用来设置属性'MIME-Version'，MIMEPart作为子部件不需要有自己的'MIME-Version'头部信息）。另外，模块email.parser中提供了解析电子邮件对象序列化结果字节串和字符串并还原为树状结构EmailMessage对象的接口，模块email.generator中提供了把EmailMessage对象序列化为字节串或字符串并直接写入文件的接口，模块email.iterators中提供了用来迭代EmailMessage对象内容的接口，模块email.policy定义了解析和序列化电子邮件对象的几种不同行为，模块email.utils中提供了localtime()、make_msgid()、quote()、parsedate()、parsedate_to_datetime()、formatdate()、format_datetime()等常用函数。

5.1.2 电子邮件对象常用方法和属性

标准库email及其子模块的核心是电子邮件类EmailMessage，其主要方法和属性如表5-1所示。完整列清单可以创建EmailMessage对象之后使用内置函数dir()查看，或者阅读官方文档，也可以阅读模块email.message的源文件，该文件位于Python安

装目录中的 Lib\email\message.py，在同一个文件夹中的很多其他模块源文件也建议阅读一下。EmailMessage 对象的头部类似于 Python 字典，除了表 5-1 中列表的方法，还支持 keys()、values()、items() 方法。

表 5-1　EmailMessage 对象的常用方法和属性

方法或属性	功能简介
__getitem__(name)	用来支持下标运算，返回头部中特定的“键”对应的“值”，功能等价于 get() 方法。如果有多个“键”与参数 name 同名，get() 方法只返回第一个“值”，建议使用 get_all(name) 方法返回所有的“值”
__len__()	支持内置函数 len()，返回头部元素的数量。邮件头部类似于 Python 的字典对象，但是“键”允许重复
__setitem__(name, val)	用来支持下标运算，设置头部中特定的“键”对应的“值”
add_alternative(*args, **kw)	必要时自动创建一个 multipart/alternative 邮件对象，把所有参数传递给该对象的 set_content() 方法，然后使用 attach() 方法添加到当前邮件的 multipart。该方法常用来实现纯文本和超文本共存于邮件体，当客户端无法显示超文本时，显示纯文本内容
add_attachment(*args, **kw)	必要时自动创建一个 multipart/mixed 邮件对象，把所有参数传递给新对象的 set_content() 方法，然后使用 attach() 方法添加到当前邮件的 multipart。常用于给邮件添加附件
add_header(_name, _value, **_params)	类似于 __setitem__() 方法，但支持以关键参数形式提供附加的头部参数。例如，支持 add_header('content-disposition', 'attachment', filename='bud.gif') 或 add_header('content-disposition', 'attachment', filename=('utf-8', '', 董付国教学课件 .ppt')) 这样的用法
add_related(*args, **kw)	必要时自动创建一个 multipart/related 邮件对象，把所有参数传递给新邮件对象的 set_content() 方法，并使用 attach() 添加到当前邮件的 multipart。常用来添加邮件体中的内嵌资源
as_bytes(unixfrom=False, policy=None)	把整个电子邮件对象转换为字节串
as_string(unixfrom=False, maxheaderlen=None, policy=None)	把整个电子邮件对象转换为字符串
del_param(param, header='content-type', requote=True)	从 Content-Type 头部中删除指定的参数
epilogue	实例属性，表示最后一个边界字符串到邮件结束之前的文本

续表

方法或属性	功能简介
get(name, failobj=None)	返回头部中 name 字段的“值”，不存在时返回参数 failobj 的值
get_all(name, failobj=None)	返回头部中 name 字段所有的“值”组成的列表
get_body(preferenceli=('related', 'html', 'plain'))	返回邮件体中最佳匹配的 MIMEPart 对象，按参数 preferencelist 指定的顺序进行搜索
get_boundary(failobj=None)	返回电子邮件 Content-Type 头部中参数 boundary 的值
get_content(*args, content_manager=None, **kw)	获取邮件体内容
get_content_charset(failobj=None)	返回 Content-Type 头部中参数 charset 的值
get_content_disposition()	返回电子邮件的 content-disposition 头部，可能的值有 'inline'、'attachment' 或 None
get_content_type()	返回邮件对象的内容类型，返回 'maintype/subtype' 形式的字符串。另外，也可以直接使用 get_content_maintype() 和 get_content_subtype() 分别返回 maintype 和 subtype
get_filename(failobj=None)	返回电子邮件 Content-Disposition 头部中参数 filename 的值，如果不存在就返回 Content-Type 头部中参数 name 的值，如果都不存在就返回参数 failobj 的值（默认为 None）
is_attachment()	如果电子邮存在 Content-Disposition 头部且值为 'attachment' 就返回 True，否则返回 False
is_multipart()	如果当前对象包含其他的 EmailMessage 子部件就返回 True，否则返回 False
iter_attachments()	遍历电子邮件中所有直接子部件中的附件，跳过每个 text/plain、text/html、multipart/related 或 multipart/alternative 的首次出现（除非通过 Content-Disposition:attachment 明确标记为附件），当作用于 multipart/related 部件时返回除根部件之外的其他所有 related 部件，返回迭代器对象
iter_parts()	返回包含邮件所有直接子部件的迭代器对象
preamble	实例属性，表示邮件头部后面的空行与第一个 multipart 边界字符串之间的文本
replace_header(_name, _value)	替换头部中第一个“键”为 _name 的元素对应的“值”为 _value，保持原来的头部元素顺序和大小写
set_boundary(boundary)	设置电子邮件 Content-Type 头部中参数 boundary 的值
set_content(*args, **kw)	设置邮件体内容

续表

方法或属性	功能简介
set_param(param, value, header='Content-Type', requote=True, charset=None, language='', replace=False)	设置或替换 Content-Type 头部的值
walk()	按深度优先的顺序遍历电子邮件中的所有部件和子部件，返回迭代器对象，每次迭代返回下一个子部件

5.1.3　构造与解析电子邮件

例 5-1　编写程序，构造一封简单的电子邮件。

```
from email.policy import SMTP
from email.message import EmailMessage
from email.utils import formatdate, make_msgid

# 创建电子邮件对象
message = EmailMessage(policy=SMTP)
# 设置头部信息
message['From'] = 'dongfuguo2005@126.com'
message['To'] = 'dongfuguo2005@126.com'
message['Subject'] = '测试邮件-主题'
message['Date'] = formatdate()
# 生成并设置唯一的ID字符串
message['Message-ID'] = make_msgid()
# 邮件内容
message.set_content('测试邮件-内容')
# 转化为字符串表达形式，输出显示
print(repr(message.as_string()))
```

运行结果如下，邮件头部和内容之间使用空行分隔，每行以回车换行符 \r\n 结束，如果某个值中有非 ASCII 字符就使用 BASE64 编码格式进行编码。上面代码最后一行使用内置函数 repr() 把邮件对象的字符串转换为适合 Python 解释器阅读的格式，也是为了方便观察每行最后的结束符。读者可以自行删除 repr() 函数的调用，逐行显示邮件对象的字符串便于阅读。

```
'From: dongfuguo2005@126.com\r\nTo: dongfuguo2005@126.com\r\nSubject: =?utf-
8?b?5rWL6K+V6YKu5Lu2LeS4u+mimA==?=\r\nDate: Sun, 03 Jan 2021 03:25:08 -0000\r\
```

```
nMessage-ID: <160964430843.39856.7499018668528534311@DESKTOP-OJ1SMKQ>\r\nContent-
Type: text/plain; charset="utf-8"\r\nContent-Transfer-Encoding: base64\r\nMIME-
Version: 1.0\r\n\r\n5rWL6K+V6YKu5Lu2LeWGheWuuQo=\r\n'
```

例 5-2　编写程序，构造一封带有普通文本、HTML 代码、图片和多个文件附件的电子邮件，然后解析刚刚创建的邮件对象，提取其中的信息。

例 5-2 讲解

```
from os.path import basename
from email.policy import SMTP
from mimetypes import guess_type
from email.iterators import _structure
from email.message import EmailMessage
from email.utils import formatdate, make_msgid

# 创建电子邮件对象
message = EmailMessage(policy=SMTP)

# 设置头部信息，发件人、收件人、邮件主题
message['From'] = 'dongfuguo2005@126.com'
message['To'] = 'dongfuguo2005@126.com'
message['Subject'] = '复杂测试邮件-主题'
# 日期形式为：'Sun, 03 Jan 2021 08:24:34 -0000'
message['Date'] = formatdate()
# 生成并设置唯一的ID字符串
message['Message-ID'] = make_msgid()

# 邮件内容
html_content = '<p><b><i>一段斜体加粗文本</i></b></p>'
plain_content = '一段加粗文本'
image_content = '<img src="cid:{}" width="360" height="280">'

# 生成唯一的ID字符串，格式为
# '<160965687060.36744.2934190533329057464@DESKTOP-OJ1SMKQ>'
cid = make_msgid()
# 在邮件正文中显示图片，使用ID时要删除两侧的尖括号
message.set_content(html_content+image_content.format(cid[1:-1]),
                    subtype='html')

# 添加要在正文中显示的图片，在邮件中会创建multipart/related段
with open('4Python可以这样学.png', 'rb') as fp:
    content = fp.read()
message.add_related(content, 'image', 'png', cid=cid,
                    filename='4Python可以这样学.png')
```

```
# 创建multipart/alternative段
# 如果客户端无法显示HTML代码，会自动显示plain_content的内容
message.add_alternative(plain_content)

# 添加文件附件，自动创建multipart/mixed段
# 演示用的两个文件都在配套资源里提供了
filenames = ('chromedriver_win32.zip', '5构造电子邮件2.py')
for fn in filenames:
    # guess_type()用来猜测文件类型，返回元组
    # 对于演示用的两个文件，分别返回：
    # ('application/x-zip-compressed', None)和('text/x-python', None)
    mime_type, encoding = guess_type(fn)
    if encoding or (mime_type is None):
        mime_type = 'application/octet-stream'
    main_type, sub_type = mime_type.split('/')
    if main_type == 'text':
        # 文本文件附件
        with open(fn, encoding='utf8') as fp:
            content = fp.read()
        message.add_attachment(content, sub_type,
                               filename=basename(fn))
    else:
        # 二进制文件附件
        with open(fn, 'rb') as fp:
            content = fp.read()
        message.add_attachment(content, main_type, sub_type,
                               filename=basename(fn))

# 查看邮件对象的结构
_structure(message)
print('='*20)

# 解析邮件对象，邮件头部字段名称不区分大小写
for header in ('From', 'to', 'date', 'Subject'):
    print(f'{header}:{message.get(header)}')
# 输出一个空行
print()

try:
    # 获取邮件体，返回邮件中最佳匹配的MIMEPart对象
    # 先搜索纯文本正文，如果没有就继续搜索HTML格式的邮件体
    # preferencelist参数的默认值为('related', 'plain', 'html')
```

```
    # 哪个在前就先搜索哪种类型，可以自行交换'plain'和'html'
    # 然后重新运行程序观察结果
    body = message.get_body(preferencelist=('plain','html'))
except:
    print('没有正文。')
else:
    # 使用内置函数repr()转换，是为了显示最后的换行符
    print(repr(body.get_content()))

# walk()方法深度优先遍历邮件对象树的所有部件，包括子孙部件
for part in message.walk():
    print('='*20)
    content_type = part.get_content_type()
    # Python 3.8语法，低版本可以删除花括号内最后的等号
    print(f'{content_type=}')
    print(f'{part.is_multipart()=}')
    # 下载附件文件
    if part.is_attachment():
        content = part.get_content()
        new_fn = '1new_' + part.get_filename()
        if content_type.startswith('text'):
            # 写入本地文本文件
            with open(new_fn, 'w', encoding='utf8') as fp:
                fp.write(content)
        else:
            # 写入本地二进制文件
            with open(new_fn, 'wb') as fp:
                fp.write(content)
        print('created file:', new_fn)

# 下载邮件直接子部件中的附件，不包括正文中插入的图片
print('直接下载附件'.center(30, '='))
for part in message.iter_attachments():
    content_type = part.get_content_type()
    content = part.get_content()
    new_fn = '2new_' + part.get_filename()
    if content_type.startswith('text'):
        # 写入本地文本文件
        with open(new_fn, 'w', encoding='utf8') as fp:
            fp.write(content)
    else:
        # 写入本地二进制文件
        with open(new_fn, 'wb') as fp:
```

```
                fp.write(content)
        print('created file:', new_fn)

print('使用iter_parts()遍历邮件'.center(30, '='))
def download_related(message):
    for part in message.iter_parts():
        if part.is_multipart():
            # 如果包含子部件，递归进去
            download_related(part)
        else:
            # 不是附件，直接跳过
            if not part.is_attachment():
                continue
            new_fn = '3new_' + part.get_filename()
            content_type = part.get_content_type()
            if content_type.startswith('text'):
                with open(new_fn, 'w', encoding='utf8') as fp:
                    fp.write(content)
            else:
                with open(new_fn, 'wb') as fp:
                    fp.write(part.get_content())
            print('created file:', new_fn)
download_related(message)
```

运行结果为

```
multipart/mixed
    multipart/alternative
        multipart/related
            text/html
            image/png
        text/plain
    application/x-zip-compressed
    text/x-python
====================
From:dongfuguo2005@126.com
to:dongfuguo2005@126.com
date:Sun, 03 Jan 2021 11:15:05 -0000
Subject:复杂测试邮件-主题

'一段加粗文本\n'
====================
content_type='multipart/mixed'
```

```
part.is_multipart()=True
====================
content_type='multipart/alternative'
part.is_multipart()=True
====================
content_type='multipart/related'
part.is_multipart()=True
====================
content_type='text/html'
part.is_multipart()=False
====================
content_type='image/png'
part.is_multipart()=False
created file: 1new_4Python可以这样学.png
====================
content_type='text/plain'
part.is_multipart()=False
====================
content_type='application/x-zip-compressed'
part.is_multipart()=False
created file: 1new_chromedriver_win32.zip
====================
content_type='text/x-python'
part.is_multipart()=False
created file: 1new_5构造电子邮件2.py
============直接下载附件============
created file: 2new_chromedriver_win32.zip
created file: 2new_5构造电子邮件2.py
======使用iter_parts()遍历邮件======
created file: 3new_4Python可以这样学.png
created file: 3new_chromedriver_win32.zip
created file: 3new_5构造电子邮件2.py
```

5.2 SMTP发送电子邮件实战

发送电子邮件主要使用 SMTP（Simple Mail Transfer Protocol），但 SMTP 存在一些不足，MIME（Multipurpose Internet Mail Extensions）协议是对 SMTP 的重要补充和辅助（并不是替代 SMTP），使得非 ASCII 字符能够通过 SMTP 进行传输。

5.2.1　smtplib.SMTP对象常用方法

Python 标准库 smtplib 提供了发送电子邮件所需要的全部功能，其中核心是 SMTP 类和 SMTP_SSL 类，两者支持同样的接口，后者适用于支持并要求 SSL 连接的服务器，本节重点以 SMTP 类为例进行演示。SMTP 类支持上下文管理关键字 with，创建对象的方法完整语法为

```
SMTP(host='', port=0, local_hostname=None,
     timeout=<object object at 0x000002D8D5254E80>,
     source_address=None)
```

表 5-2 列出了 smtplib.SMTP 对象的常用方法（不包含隐式调用的方法），可以使用 help(smtplib.SMTP) 查看完整方法清单和详细用法。

表 5-2　SMTP 对象的常用方法

方　法	简 要 说 明
close()	关闭到 SMTP 服务器的连接
login(user, password, *, initial_response_ok=True)	登录需要身份认证的 SMTP 服务器，需要提供用户名和授权码（早期支持使用邮箱密码，现在推荐使用更安全的授权码）
starttls(keyfile=None, certfile=None, context=None)	切换 SMTP 连接为 TLS（Transport Layer Security）模式，从此之后所有 SMTP 命令都将被加密。使用该方法时优先推荐使用 context 参数
sendmail(from_addr, to_addrs, msg, mail_options=(), rcpt_options=())	发送电子邮件，其中参数 from_addr 为发件人电子邮箱地址，参数 to_addrs 为收件人电子邮件地址列表（也可以是表示单个收件人地址的字符串），参数 msg 为要发送的电子邮件字符串或字节串。如果邮件被成功发送给至少一个收件人，该方法正常返回，否则会抛出异常。如果该方法没有抛出异常，表示至少有一个收件人收到了电子邮件，此时该方法返回一个字典，其中每个元素表示一个没有成功发送的收件人信息
send_message(msg, from_addr=None, to_addrs=None, mail_options=(), rcpt_options=())	发送电子邮件，参数 msg 为 email.message.Message 对象，其他参数的含义与 sendmail() 方法的参数相同
quit()	退出 SMTP 会话

5.2.2　设置电子邮箱开启SMTP服务

如果要使用程序来登录电子邮箱进行发送和接收邮件，需要首先对电子邮箱账号进行设置。本节以 126 邮箱和 QQ 邮箱为例进行介绍，其他邮箱的设置方式与此类似，可以参