Python使用POP3和SMTP协议收发邮件！

萌新程序员
程序员

+ 关注他

先来了解一下收/发邮件有哪些协议：

- **SMTP协议**
- SMTP（Simple Mail Transfer Protocol），即简单邮件传输协议。相当于中转站，将邮件发送到客户端。
- **POP3协议**
- POP3（Post Office Protocol 3），即邮局协议的第3个版本，是电子邮件的第一个离线协议标准。该协议把邮件下载到本地计算机，不与服务器同步，缺点是更易丢失邮件或多次下载相同的邮件。
- **IMAP协议**
- IMAP（Internet Mail Access Protocol），即交互式邮件存取协议。该协议连接远程邮箱直接操作，与服务器内容同步。
- **Exchange服务**
- Exchange服务是一个设计完备的邮件服务器产品，提供了通常所需要的全部邮件服务功能。除了常规SMTP/POP协议服务之外，它还支持IMAP4 、LDAP和NNTP协议。

Python内置对SMTP/POP3/IMAP的支持。更多详情请移步Python官方教程

SMTP发送邮件

Python对SMTP支持有 smtplib 和 email 两个模块， email 负责构造邮件， smtplib 负责发送邮件。

构造邮件

构造最简单的纯文本邮件，如下：

```
from email.mime.text import MIMEText
msg = MIMEText('hello, send by Python...', 'plain', 'utf-8')
复制代码
```

注意到构造 MIMEText 对象时，第一个参数就是邮件正文，第二个参数是MIME的subtype，传入 'plain' 表示纯文本，最终的MIME就是 'text/plain' ，最后一定要用 utf-8 编码保证多语言兼容性。

发送邮件

```
import smtplib
# 输入Email地址和口令:
from_addr = 'test_from_addr@qq.com'
password = 'Password'
# 输入收件人地址:
to_addr = 'test_to_addr@qq.com'
# 输入SMTP服务器地址:
smtp_server = smtp.qq.com
server = smtplib.SMTP(smtp_server, 25) # SMTP协议默认端口是25
# server.starttls() # 如果是SSL，则用 587 端口，再加上这句代码就行了
server.set_debuglevel(1)    # 打印出和SMTP服务器交互的所有信息
server.login(from_addr, password)   # 登录SMTP服务器
server.sendmail(from_addr, [to_addr], msg.as_string())    # 发邮件
server.quit()
```

sendmail() 方法就是发邮件，由于可以一次发给多个人，所以传入一个 list ， 邮件正文是一个 str ， as_string() 把MIMEText对象变成 str 。

注意：QQ邮件等需要手动开通 SMTP服务， 邮箱设置 => 账号 => POP3/SMTP服务，如下图：

此时，我们就可以收到邮件了，如下：

添加邮件标题、收/发件人

邮件主题、显示发件人、收件人等信息并不是通过SMTP协议发送的，而是包含在 MIMEText 对象中，如下：

```
from email import encoders
from email.header import Header
from email.mime.text import MIMEText
from email.utils import parseaddr, formataddr
import smtplib
def _format_addr(s):
    name, addr = parseaddr(s)
    return formataddr((Header(name, 'utf-8').encode(), addr))
from_addr = 'test_from_addr@qq.com'
password = 'Password'
to_addr = 'test_to_addr@qq.com'
smtp_server = smtp.qq.com
msg = MIMEText('hello, send by Python...', 'plain', 'utf-8')
msg['From'] = _format_addr('发件人昵称 <%s>' % from_addr)
msg['To'] = _format_addr('收件人昵称 <%s>' % to_addr)
msg['Subject'] = Header('这是个有主题的邮件', 'utf-8').encode()
server = smtplib.SMTP(smtp_server, 25)
server.set_debuglevel(1)
server.login(from_addr, password)
server.sendmail(from_addr, [to_addr], msg.as_string())
server.quit()
复制代码
```

收到的邮件，如下：

收件人并不是我们设置的 “收件人昵称”，是因为很多邮件服务商在显示邮件时，会把收件人名字自动替换为用户注册的名字，这无伤大雅。

发送HTML邮件

要发送HTML邮件很简单，在构造 MIMEText 对象时，把HTML字符串传进去，再把第二个参数由 plain 变为 html ，如下：

```
msg = MIMEText('<html><body><h1>Hello</h1>' +
    '<p>send by <a href="http://blog.pangao.vip">PanGao’s blog</a>...</p>' +
    '</body></html>', 'html', 'utf-8')
复制代码
```

发送附件

要想发送附件，需要构造一个 MIMEMultipart 对象代表邮件本身，然后往里面加上一个 MIMEText 作为邮件正文，再继续往里面加上表示附件的 MIMEBase 对象，如下：

```
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
# 邮件对象:
msg = MIMEMultipart()
msg['From'] = _format_addr('发件人昵称 <%s>' % from_addr)
msg['To'] = _format_addr('收件人昵称 <%s>' % to_addr)
msg['Subject'] = Header('这是个有主题的邮件', 'utf-8').encode()
# 邮件正文是MIMEText:
msg.attach(MIMEText('send with file...', 'plain', 'utf-8'))
# 添加附件就是加上一个MIMEBase，从本地读取一个图片:
with open('/Users/pangao/Downloads/test.png', 'rb') as f:
    # 设置附件的MIME和文件名，这里是png类型:
    mime = MIMEBase('image', 'png', filename='test.png')
    # 加上必要的头信息:
    mime.add_header('Content-Disposition', 'attachment', filename='test.png')
    mime.add_header('Content-ID', '<0>')
    mime.add_header('X-Attachment-Id', '0')
    # 把附件的内容读进来:
    mime.set_payload(f.read())
    # 用Base64编码:
    encoders.encode_base64(mime)
    # 添加到MIMEMultipart:
    msg.attach(mime)
复制代码
```

发送图片

由于 mac 自带的邮件会自动把图片附件插入邮件正文中，所以样式很好看。但是普通邮件可能没这么便捷（抱歉，我没见过普通邮件。。。小小得瑟一下）

如果要把一个图片嵌入到邮件正文中怎么做？直接在HTML邮件中链接图片地址行不行？答案是，大部分邮件服务商都会自动屏蔽带有外链的图片，因为不知道这些链接是否指向恶意网站。

要把图片嵌入到邮件正文中，我们只需按照发送附件的方式，先把邮件作为附件添加进去，然后，在HTML中通过引用 src="cid:0" 就可以把附件作为图片嵌入了。如果有多个图片，给它们依次编号，然后引用不同的 cid:x 即可。

把上面代码加入 MIMEMultipart 的 MIMEText 从 plain 改为 html ，然后在适当的位置引用图片，如下：

```
msg.attach(MIMEText('<html><body><h1>Hello</h1>' +
    '<p><img src="cid:0"></p>' +
    '</body></html>', 'html', 'utf-8'))
```
复制代码

同时支持HTML和Plain格式

如果我们发送HTML邮件，收件人通过浏览器或者Outlook之类的软件是可以正常浏览邮件内容的，但是，如果收件人使用的设备太古老，查看不了HTML邮件怎么办?

办法是在发送HTML的同时再附加一个纯文本，如果收件人无法查看HTML格式的邮件，就可以自动降级查看纯文本邮件。

利用 MIMEMultipart 就可以组合一个HTML和Plain，要注意指定subtype是 alternative ，如下：

```
msg = MIMEMultipart('alternative')
msg['From'] = ...
msg['To'] = ...
msg['Subject'] = ...
msg.attach(MIMEText('hello', 'plain', 'utf-8'))
msg.attach(MIMEText('<html><body><h1>Hello</h1></body></html>', 'html', 'utf-8'))
# 正常发送msg对象...
复制代码
```

加密SMTP

使用标准的25端口连接SMTP服务器时，使用的是明文传输，发送邮件的整个过程可能会被窃听。要更安全地发送邮件，可以加密SMTP会话，实际上就是先创建SSL安全连接，然后再使用SMTP协议发送邮件。

某些邮件服务商，例如Gmail，提供的SMTP服务必须要加密传输。我们来看看如何通过Gmail提供的安全SMTP发送邮件。

必须知道，Gmail的SMTP端口是587，因此，修改代码如下：

```
smtp_server = 'smtp.gmail.com'
smtp_port = 587
server = smtplib.SMTP(smtp_server, smtp_port)
server.starttls()
# 剩下的代码和前面的一模一样:
server.set_debuglevel(1)
...
复制代码
```

只需要在创建 SMTP 对象后，立刻调用 starttls() 方法，就创建了安全连接。后面的代码和前面的发送邮件代码完全一样。

POP3收取邮件

Python内置一个 poplib 模块，实现了POP3协议，可以直接用来收邮件。

注意到POP3协议收取的不是一个已经可以阅读的邮件本身，而是邮件的原始文本，这和SMTP协议很像，SMTP发送的也是经过编码后的一大段文本。

要把POP3收取的文本变成可以阅读的邮件，还需要用email模块提供的各种类来解析原始文本，变成可阅读的邮件对象。

所以，收取邮件分两步：

第一步：用 poplib 把邮件的原始文本下载到本地；

第二部：用 email 解析原始文本，还原为邮件对象。

通过POP3下载邮件

POP3协议本身很简单，以下面的代码为例，我们来获取最新的一封邮件内容：

```
from email.parser import Parser
import poplib
# 输入邮件地址, 口令和POP3服务器地址:
email = 'pangao1990@qq.com'
password = 'Password'
pop3_server = 'pop.qq.com'
# 连接到POP3服务器:
server = poplib.POP3_SSL(pop3_server)
# 可以打开或关闭调试信息:
```

```
# 身份认证:
server.user(email)
server.pass_(password)
# list()返回所有邮件的编号:
resp, mails, octets = server.list()
# 获取最新一封邮件, 注意索引号从1开始:
index = len(mails)
resp, lines, octets = server.retr(index)
# lines存储了邮件的原始文本的每一行,
# 可以获得整个邮件的原始文本:
msg_content = b'\r\n'.join(lines).decode('utf-8')
# 稍后解析出邮件:
msg = Parser().parsestr(msg_content)
# 可以根据邮件索引号直接从服务器删除邮件:
# server.dele(index)
# 关闭连接:
server.quit()
复制代码
```

但是这个 Message 对象本身可能是一个 MIMEMultipart 对象，即包含嵌套的其他 MIMEBase 对象，嵌套可能还不止一层。

所以我们要递归地打印出 Message 对象的层次结构：

```
from email.header import decode_header
from email.utils import parseaddr
def print_info(msg, indent=0):
    if indent == 0:
        for header in ['From', 'To', 'Subject']:
            value = msg.get(header, '')
            if value:
                if header == 'Subject':
                    value = decode_str(value)
                else:
                    hdr, addr = parseaddr(value)
                    name = decode_str(hdr)
                    value = u'%s <%s>' % (name, addr)
            print('%s%s: %s' % ('  ' * indent, header, value))
    if (msg.is_multipart()):
        parts = msg.get_payload()
        for n, part in enumerate(parts):
            print('%spart %s' % ('  ' * indent, n))
            print('%s--------------------' % ('  ' * indent))
            print_info(part, indent + 1)
    else:
        content_type = msg.get_content_type()
        if content_type == 'text/plain' or content_type == 'text/html':
            content = msg.get_payload(decode=True)
            charset = guess_charset(msg)
            if charset:
                content = content.decode(charset)
            print('%sText: %s' % ('  ' * indent, content + '...'))
        else:
            print('%sAttachment: %s' % ('  ' * indent, content_type))
def decode_str(s):
    value, charset = decode_header(s)[0]
    if charset:
        value = value.decode(charset)
    return value
def guess_charset(msg):
    charset = msg.get_charset()
    if charset is None:
        content_type = msg.get('Content-Type', '').lower()
```

```
            charset = content_type[pos + 8:].strip()
    return charset
print_info(msg) #解析
# From: 木叶 <pangao1990@qq.com>
# To:  <mail@pangao.vip>
# Subject: 测试主题
# Text: 测试内容
#
# ...
```
复制代码

发布于 2019-04-17 14:52

Python

欢迎参与讨论

1 条评论 默认 最新

非凡-上海 ...

zmail挺好用的

2019-04-17 回复 喜欢

文章被以下专栏收录

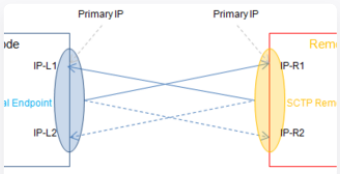
Python学习交流专栏

Python学习交流群：984632579

推荐阅读

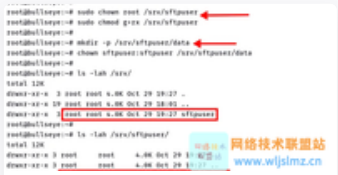
SMTP协议简析

Quinn



SCTP通讯协议

洲洋讲编程 发表于科技日报



如何在 Linux 服务器上安装和使用 SFTP?

网络技术联... 发表于网络技术联...

浅谈SFTP和FTP的区别

一、适用场景 我们平时习惯了使用ftp来上传下载文件，尤其是很多Linux环境下，我们一般都会通过第三方的SSH工具连接到Linux，但是当我们需要传输文件到Linux服务器当中，很多人习惯用ftp来...

yanlei