Python3读取邮件内容

Python3读取邮件内容

前言

邮件的收取主要有pop(主要用于客户端远程管理服务器上的邮件)和imap(交互式邮件访问协议)，相应的Python中提供了相关的模块poplib和imaplib。POP3尽管得到广泛的支持，但其已经过时，而且POP3服务器的实现差异很大，大多数进行较差，所以如果我们的邮件服务器支持IMAP，那么最好使用imaplib.IMAP4，因为IMAP服务器往往会更好的实现。基本上主流的邮箱都会支持imap协议，如qq、163、gmail、outlook等等。因此我们选择imap协议来实现读取邮件的脚本。

实现过程

- 登录邮箱并读取原始邮件

  使用imaplib库实现邮箱登录，所以需要先导入库import imaplib，然后利用imaplib库中的方法登录邮箱并读取邮件

```
def get_mail(email_address, password):
    # 这里的服务器根据需要选择
    server = imaplib.IMAP4_SSL("imap.gmail.com")
    server.login(email_address, password)
    # 邮箱中的文件夹，默认为'INBOX'
    inbox = server.select("INBOX")
    # 搜索匹配的邮件，第一个参数是字符集，None默认就是ASCII编码，第二个参数是查询条件，这里的ALL就是查找全部
    type, data = server.search(None, "ALL")
    # 邮件列表,使用空格分割得到邮件索引
    msgList = data[0].split()
    # 最新邮件，第0封邮件为最早的一封邮件
    latest = msgList[len(msgList) - 1]
    type, datas = server.fetch(latest, '(RFC822)')
    # 使用utf-8解码
    text = datas[0][1].decode('utf8')
    # 转化为email.message对象
    message = email.message_from_string(text)
    return message
```

  上述程序返回值为email.message，即原始邮件，如果打印出来，我们会发现这些一些代码，无法读懂，因此接下来我们需要将原始邮件转化为可读邮件

  关于email.message

  电子邮件消息由 *headers* 和 *payload* （其也被称为 *content*）组成。标题是 **RFC 5322** 或 **RFC 6532** 样式的字段名称和值。有效载荷可以是简单文本消息，或二进制对象或子消息的结构化序列，每个子消息具有它们自己的一组头部和它们自己的有效载荷。后一类型的有效载荷由具有诸如 *multipart*或 *message/rfc822* 的MIME类型的消息指示。

  由 EmailMessage 对象提供的概念模型是与表示消息的 **RFC 5322** 主体的 *payload* 耦合的标题的有序字典，其可以是子 EmailMessage 对象的列表。除了用于访问头部名称和值的常规字典方法之外，还存在用于从头部（例如MIME内容类型）访问专用信息，用于在有效载荷上操作，用于生成消息的序列化版本的方法，以及用于递归地遍历对象树。

  EmailMessage 类字典接口由标题名称索引，标题名称必须是ASCII值。字典的值是带有一些额外方法的字符串。头以字节保存的形式存储和返回，但字段名匹配大小写不敏感。不像真正的

dict，有一个排序的键，并可以有重复的键。提供了其他方法来处理具有重复键的标头。

- 将原始邮件转化为可读邮件

  1. 邮件的Subject或者Email中包含的名字都是经过编码后的字符串，要正常显示就必须decode，定义 一个decode函数

```
def decode_str(s):
    value, charset = decode_header(s)[0]
    if charset:
        value = value.decode(charset)
    return value
```

  2. 为了防止非UTF-8编码的邮件无法显示，定义一个检测邮件编码函数

```
def guess_charset(msg):
    charset = msg.get_charset()
    if charset is None:
        content_type = msg.get('Content-Type', '').lower()
        pos = content_type.find('charset=')
        if pos >= 0:
            # 去掉尾部不代表编码的字段
            charset = content_type[pos + 8:].strip('; format=flowed; delsp=yes')
    return charset
```

  3. 接下来通过循环遍历来读取邮件内容

```
# 使用全局变量来保存邮件内容
mail_content = '¥n'
# indent用于缩进显示:
def print_info(msg, indent=0):
    global mail_content
    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)
            mail_content += '%s%s: %s' % ('  ' * indent, header, value) + '¥n'
    parts = msg.get_payload()
    for n, part in enumerate(parts):
        content_type = part.get_content_type()
        if content_type == 'text/plain':
            content = part.get_payload(decode=True)
            # charset = guess_charset(msg)
            charset = 'utf-8'
            if charset:
                content = content.decode(charset)
            mail_content += '%sText:¥n %s' % ('  ' * indent, content)
        else:
            # 这里没有读取非text/plain类型的内容，只是读取了其格式，一般为text/html
            mail_content += '%sAttachment: %s' % ('  ' * indent, content_type)
        return mail_content
```

  4. 最后，调用上述函数，输出邮件内容

```
if __name__ == '__main__':
    email_addr = "myEmail@gmail.com"
    password = "mypassword"
    test = print_info(get_mail(email_addr, password))
    print("mail content is: %s" % test)
```

相关问题

- 邮件拒绝访问？

  gmail邮箱的安全性相当高，所以在读取gmail邮件前需要对邮箱进行设置，主要设置两个方面：启用*imap*服务，启用安全性较低的应用的访问权限，设置步骤：

  1. 进入邮箱，选择设置，点击"转发和POP/IMAP"选项，选择启用IMAP，保存设置
  2. 进入网页，选择"启用"选项

- 其他邮件设置的问题？

参考文档：在使用 POP 的其他电子邮件客户端上阅读 Gmail 邮件

Gmail相关设置：监控 Gmail 设置的运行状况