第 129 天：爬取微信公众号文章内容

原创 某某白米饭 Python技术 2月18日

有时候我们遇到一个好的公众号，里面的每篇都是值得反复阅读的，这时就可以使用公众号爬虫将内容抓取保存下来慢慢赏析。

安装 Fiddler

Fiddler 的下载地址为：https://www.telerik.com/download/fiddler ，安装好之后，确保手机和电脑的网络为同一个局域网。

Finddler 的配置

点击 Tools >> Options >> Connections 面板，参考下图配置，Fiddler 的默认端口为 8888，如果 8888 端口被占用了，可修改为其他端口。

点击 Tools >> Options >> HTTPS 面板，参考下图配置

Android 手机配置

进入 WLAN 设置，选择当前所在局域网的 WIFI 设置，代理设置为 手动 ，代理服务器主机名为 Finddler 中 右上角 Online 点击显示，端口号为 8888。

在手机浏览器中访问配置的地址：http://ip:8888，当显示 Fiddler Echo Service，则配置手机成功。

Finddler 为了拦截 HTTPS 请求，手机中必须安装 CA 证书，在 http://ip:8888 也中点击 FiddlerRoot certificate，下载并安装证书。此时配置工作全部完成。

微信历史页面

以 【腾旭大申网】为例，点击【上海新闻】菜单的二级菜单【历史消息】。

观察 Fiddler 的变化，此时在左侧窗口中会陆续出现多个URL连接地址，这个就是 Fiddler 拦截的 Android 请求。

1. Result：服务器的响应结果
2. Protocol：请求协议，微信协议都是 HTTPS 所以需要在手机端和PC端安装证书
3. HOST：主机名
4. URL：URL 地址

其中有一条以https://mp.weixin.qq .com/mp/profile_ext?action=home... 开头的URL就是我们需要的。点击 右侧 Inspectors 面板，再点击下面的 Headers 和 WebView 面板，会出现如下图样

Headers 面板

1. Request Headers：请求行，里面有请求方式、请求地址、请求协议等待
2. Client、Cookies：请求头

WebView 面板

WebView 面板显示的是服务器返回的 HTML 代码渲染后的结果，Textview 面板则显示的为服务器返回的 HTML 源代码

抓取历史页面

在上一节中公众号消息历史页面已经可以显示在 Fiddler 的 WebView 面板了，这一节则使用 Python 抓取历史页面。创建一个名为 wxcrawler.py 的脚本，抓取页面我们需要 URL 地址和 HEADER 请求头，直接从 Finddler 中拷贝

把 header 转换为 Json

```
# coding:utf-8
import requests

class WxCrawler(object):

    # 复制出来的 Headers，注意这个 x-wechat-key，有时间限制，会过期。当返回的内容出现 验证 的情况，就需要换 x-
    headers = """Connection: keep-alive
        x-wechat-uin: MTY4MTI3NDIxNg%3D%3D
        x-wechat-key: 5ab2dd82e79bc5343ac5fb7fd20d72509db0ee1772b1043c894b24d441af288ae942feb4cfb4d2
        Upgrade-Insecure-Requests: 1
        User-Agent: Mozilla/5.0 (Linux; Android 10; GM1900 Build/QKQ1.190716.003; wv) AppleWebKit/53
        Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/wxpic,image/a
        Accept-Encoding: gzip, deflate
        Accept-Language: zh-CN,en-US;q=0.9
        Cookie: wxuin=1681274216; devicetype=android-29; version=27000933; lang=zh_CN; pass_ticket=J
```

```
        X-Requested-With: com.tencent.mm"""

    url = "https://mp.weixin.qq .com/mp/profile_ext?action=home&__biz=MjEwNDI4NTA2MQ==&scene=123&dev


    # 将 Headers 转换为 字典
    def header_to_dict(self):
        headers = self.headers.split("\n")
        headers_dict = dict()
        for h in headers:
            k,v = h.split(":")
            headers_dict[k.strip()] = v.strip()
        return headers_dict;


    def run(self):
        headers = self.header_to_dict()
        response = requests.get(self.url, headers=headers, verify=False)

        print(response.text)


if __name__ == "__main__":

    wx = WxCrawler()
    wx.run()
```

下图就是打印在控制台的内容，其中在 JavaScript 中 变量 msgList 的值就是需要的内容

接下来就是提取 msgList 内容，使用正则表达式提取内容，返回一个文章列表

```
import re
import html
import json

def article_list(self, context):
    rex = "msgList = '({.*?})'"
    pattern = re.compile(pattern=rex, flags=re.S)
    match = pattern.search(context)
    if match:
        data = match.group(1)
        data = html.unescape(data)
        data = json.loads(data)
        articles = data.get("list")
        return articles
```

下面就是解析 msgList 的结果

1. title：文章标题
2. content_url：文章链接
3. source_url：原文链接，有可能为空
4. digest：摘要
5. cover：封面图
6. datetime：推送时间

其他的内容保存在 multi_app_msg_item_list 中


```
{'comm_msg_info':
    {
        'id': 1000033457,
        'type': 49,
        'datetime': 1575101627,
        'fakeid': '2104285061',
        'status': 2,
        'content': ''
    },
    'app_msg_ext_info':
    {
        'title': '快查手机！5000多张人脸照正被贱卖，数据曝光令人触目惊心！',
        'digest': '谁有权收集人脸信息？',
        'content': '',
        'fileid': 0,
      'content_url': 'http:\\/\\/mp.weixin.qq.com\\/s?__biz=MjEwNDI4NTA2MQ==&amp;mid=2651824634&amp;
        'source_url': '',
        'cover': 'http:\\/\\/mmbiz.qpic.cn\\/mmbiz_jpg\\/G8vkERUJibkstwkIvXB960sMOyQdYF2x2qibTxAIq2e
        'subtype': 9,
        'is_multi': 1,
        'multi_app_msg_item_list':
            [{
                'title': '先有鸡还是先有蛋？6.1亿年前的胚胎化石揭晓了',
                'digest': '解决了困扰大申君20多年的问题',
                'content': '',
                'fileid': 0,
                'content_url': 'http:\\/\\/mp.weixin.qq.com\\/s?__biz=MjEwNDI4NTA2MQ==&amp;mid=26518
                'source_url': '',
                'cover': 'http:\\/\\/mmbiz.qpic.cn\\/mmbiz_jpg\\/yl6JkZAE3S92BESibpZgTPE1BcBhSLiaGOg
                'author': '',
                'copyright_stat': 100,
                'del_flag': 1,
                'item_show_type': 0,
                'audio_fileid': 0,
                'duration': 0,
                'play_url': '',
                'malicious_title_reason_id': 0
```

```
                'malicious_content_type': 0
            },
            {
                'title': '外交部惊现“李佳琦”！网友直呼：“OMG被种草了！”',
                'digest': '种草了！',
                'content': '', ...}
            ...]
```

抓取单个页面

在上节中我们可以得到 app_msg_ext_info 中的 content_url 地址了，这是需要从 comm_msg_info 这个不规则的 Json 中取出。这是需要使用 demjson 模块补全不规则的 comm_msg_info。

安装 demjson 模块

```
pip3 install demjson
```

```
import demjson

# 获取单个文章的URL
content_url_array = []

def content_url(self, articles):
    content_url = []
    for a in articles:
        a = str(a).replace("\/", "/")
        a = demjson.decode(a)
        content_url_array.append(a['app_msg_ext_info']["content_url"])
        # 取更多的
        for multi in a['app_msg_ext_info']["multi_app_msg_item_list"]:
            self.content_url_array.append(multi['content_url'])
    return content_url
```

获取到单个文章的地址之后，使用 requests.get() 函数取得 HTML 页面并解析

```

# 解析单个文章
def parse_article(self, headers, content_url):
    for i in content_url:
        content_response = requests.get(i, headers=headers, verify=False)
        with open("wx.html", "wb") as f:
            f.write(content_response.content)
        html = open("wx.html", encoding="utf-8").read()
        soup_body = BeautifulSoup(html, "html.parser")
```

```
        context = soup_body.find('div', id = 'js_content').text.strip()
        print(context)
```

所有历史文章

把历史消息往下滑动时出现了正在加载中...，这是公众号的历史消息正在翻页，在 Fiddler 中查看得知，公众号请求的地址为 https://mp.weixin.qq .com/mp/profile_ext?action=getmsg&__biz...

翻页请求地址返回结果，一般可以分析出

1. ret：是否成功，0为成功
2. msg_count：每页的条数
3. can_msg_continue：是否继续翻页，1为继续翻页
4. general_msg_list：数据，包含了标题、文章地址等信息

```
def page(self, headers):
    response = requests.get(self.page_url, headers=headers, verify=False)
    result = response.json()
    if result.get("ret") == 0:
        msg_list = result.get("general_msg_list")
        msg_list = demjson.decode(msg_list)
        self.content_url(msg_list["list"])
        #递归
        self.page(headers)
    else:
        print("无法获取内容")
```

总结

到这里已经爬取到了公众号的内容，但是单个文章的阅读数和在看数还未爬取。思考一下，这些内容改如何爬取?

> 示例代码：Python-100-days

系列文章

第123天：Web 开发 Django 管理工具
第122天：Flask 单元测试
第121天：机器学习之决策树
从 0 学习 Python 0 - 120 大合集总结

PS： 公号内回复 ：Python，即可进入Python 新手学习交流群，一起**100天计划！**

-END-

Python 技术

关于 **Python** 都在这里