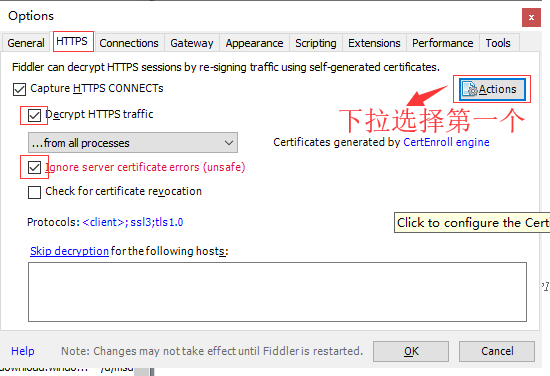
## 反反爬虫——伪装User-Agent

## 安装Fiddler

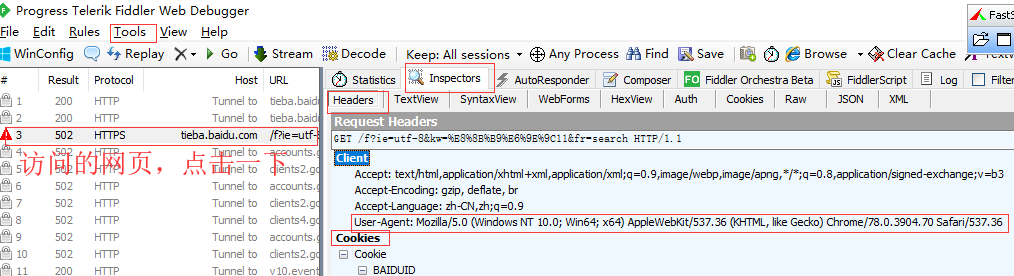
选一英文路径，默认安装。

配置fiddler

Tools 🡪 options



打开浏览器，访问一个任意一个网页，抓包。

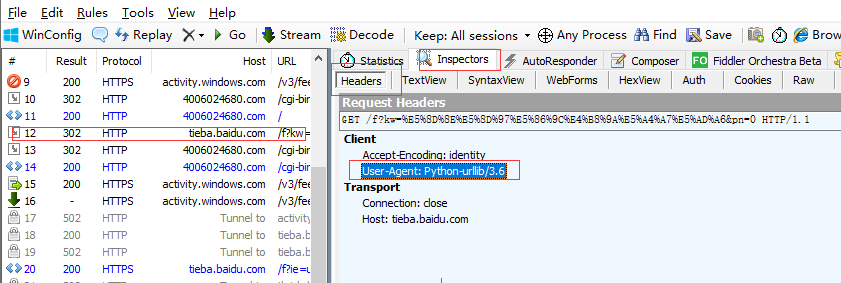


得到请求头！

|  |
| --- |
| User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.70 Safari/537.36 |

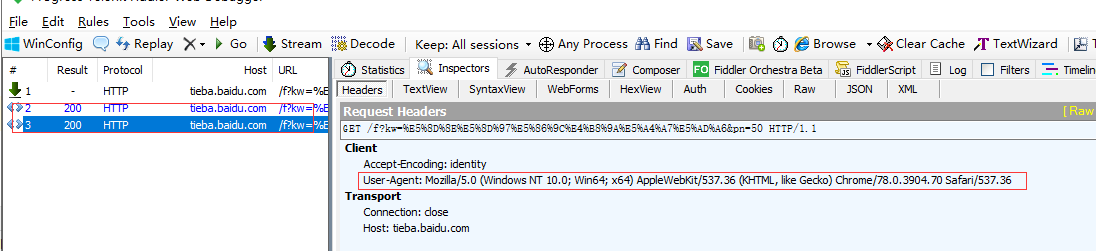
抓取爬虫程序的请求包，发现User-Agent是User-Agent: Python-urllib/3.6。

这种情况下，如果不伪装请求头，很容易被网站管理员封IP



伪装请求头：

|  |
| --- |
| **from** urllib **import** request **import** ssl  **def** loadPage(url):  **context = ssl.\_create\_unverified\_context()** *# 创建未经验证的上下文  #伪装一个User-agent* **headers = {'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.70 Safari/537.36'}**   req = request.Request(url,**headers=headers**) *#构建请求对象* response = request.urlopen(req,**context=context**) *#发送请求,得到响应对象* html = response.read() *#获取响应对象的内容* **return** html |



随机数，选取任意一个User-Agent进行伪装

|  |
| --- |
| **def** loadPage(url):  context = ssl.\_create\_unverified\_context() *# 创建未经验证的上下文* ua\_list = [  **'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/5.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; .NET4.0C; .NET4.0E; SE 2.X MetaSr 1.0)'**,  **'Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US) AppleWebKit/534.3 (KHTML, like Gecko) Chrome/6.0.472.33 Safari/534.3 SE 2.X MetaSr 1.0'**,  **'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; .NET4.0C; .NET4.0E)'**,  **'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/13.0.782.41 Safari/535.1 QQBrowser/6.9.11079.201'**,  **'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/5.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; .NET4.0C; .NET4.0E) QQBrowser/6.9.11079.201'**,  **'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)'**,  **'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.70 Safari/537.36'** ]  user\_agent = random.choice(ua\_list)  *#伪装一个User-agent* headers = {**'User-Agent'**:user\_agent}  print(headers)  req = request.Request(url,headers=headers) *#构建请求对象* response = request.urlopen(req,context  =context) *#发送请求,得到响应对象* html = response.read() *#获取响应对象的内容* **return** html |

## 安装lxml

管理员进入cmd Dos命令行， 输入pip install lxml

## Xpath

筛选匹配需要的数据，

解压xpath\_helper\_2\_0\_2.rar, 存放在一个英文路径上（D:\Program Files）

在谷歌浏览器上，打开“扩展程序”的按钮， 点击打开开发者模式，

然后点击“加载已解压的扩展程序” 

### Xpath规则

全文查找

|  |
| --- |
| //div |
| //a |

子路径查找

|  |
| --- |
| //div/div/span |
| //div/div/a |

在当前路径下查找

|  |
| --- |
| ./div/a |
| ./div/span |

查找属性

|  |
| --- |
| //div[@class='t\_con cleafix'] |
| //span[@class='threadlist\_rep\_num center\_text'] |

取文本 text()

|  |
| --- |
| //span[@class='threadlist\_rep\_num center\_text']/text() |
| //a/text() |

取属性

|  |
| --- |
| //a/@href |
| //img/@src |

找出贴吧标题、回复数，链接，简介的xpath规则

标题： //div[@class='t\_con cleafix']/div/div/div/a/text()

帖子链接： //div[@class='t\_con cleafix']/div/div/div/a/@href

回复数：//div[@class='t\_con cleafix']/div/span/text()

简介： ？

楼主： ？

最后回复人： ？

## 爬取帖子标题、链接、回复数

|  |
| --- |
| **def** tiebaInfo(html):  content = etree.HTML(html) *#解析HTML文档* title\_list = content.xpath(**"//div[@class='t\_con cleafix']/div/div/div/a/text()"**)  link\_list = content.xpath(**"//div[@class='t\_con cleafix']/div/div/div/a/@href"**)  ans\_list = content.xpath(**"//div[@class='t\_con cleafix']/div/span/text()"**)  **for** title,link,ans **in** zip(title\_list,link\_list,ans\_list):  print(title)  print(link)  print(ans) |
| **……**  **for** page **in** range(beginPage,endPage+1):  pn = (page-1)\*50  fullurl = url + **'&pn='** + str(pn)  print(fullurl)   html = loadPage(fullurl)   tiebaInfo(html) |

## 验收作业

1、找出简介、楼主、最后回复人的xpath规则，

2、爬取标题、帖子链接（需拼接成完整的链接）、回复数、简介、楼主、最后回复人等信息

3、将标题、帖子链接、回复数、简介、楼主、最后回复人等信息写入到本地文本文档（xx贴吧.txt）中

