如何利用 Selenium 爬取评论数据？

一、前言

我们知道，如今的 web 网页数据很多是动态加载的，普通的爬虫只是抓取静态的网页。实用性很差，因此，我们需要使用 Selenium 来爬取动态数据。评论区的数据，大多数情况下，都需要下拉刷新才能加载出来。而 Selenium 就能帮我们很好的爬取动态数据。

在本场 Chat 中我将介绍如何用基于 Selenium 的爬虫爬取 B 站评论，并介绍如何用 Firefox 浏览器的实用插件 FirePath 协助爬虫。主要内容包括：

1. 对比静态爬虫与动态爬虫
2. 什么是 Selenium？Selenium 工具的安装（基于 Firefox 浏览器）
3. 介绍强大的 Xpath 定位工具——FirePath 协助爬虫
4. 实例操作：爬取 B 站评论
5. 拓展：介绍 Tar 浏览器，实现匿名 IP 爬虫，防止 IP 封禁

二、环境搭建

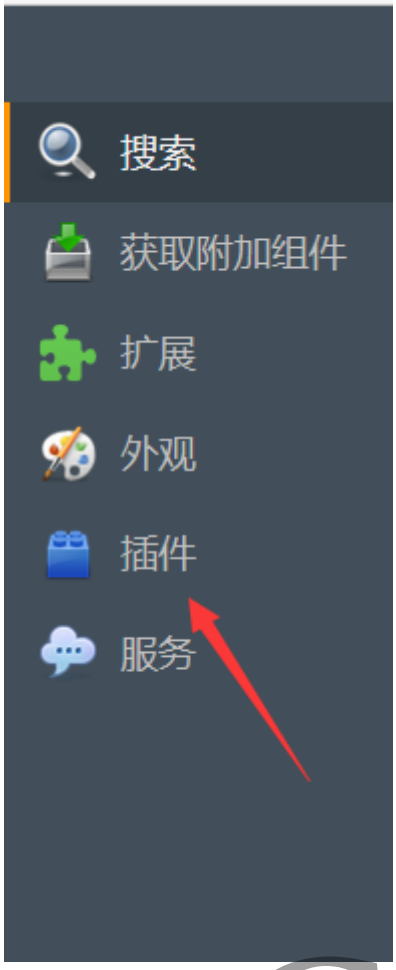
（1）Windows10（有兴趣的小伙伴可以在Linux尝试）

（2）IDE：JetBrains PyCharm Community Edition 2017.1.2 x64（如果是学生、可以申请到免费版）

（3）Python2.7、pip工具

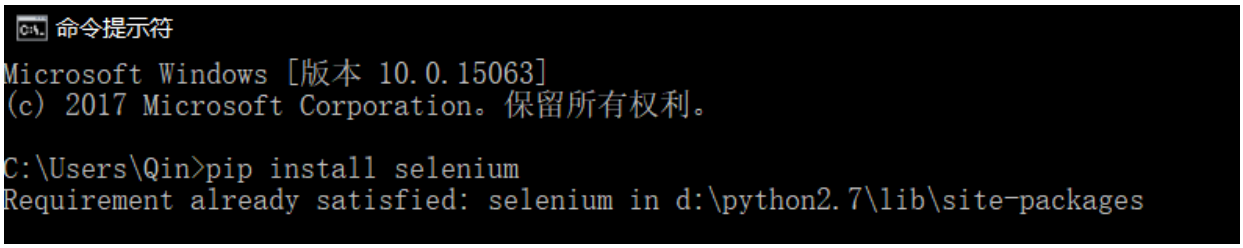
（4）Firefox浏览器（版本55.0）以及Firefox下的插件FirePath


剪切
复制
粘贴
100%
新建窗口
新建隐私窗口
保存页面
打印
历史记录
全屏
查找
选项
附加组件
开发者
受同步标签页
登录同步服务
定制


搜索
获取附加组件
扩展
外观
插件
服务


firepath
名称
上次更新
最佳匹配


搜索：
我的附加组件
其他可用的附加组件

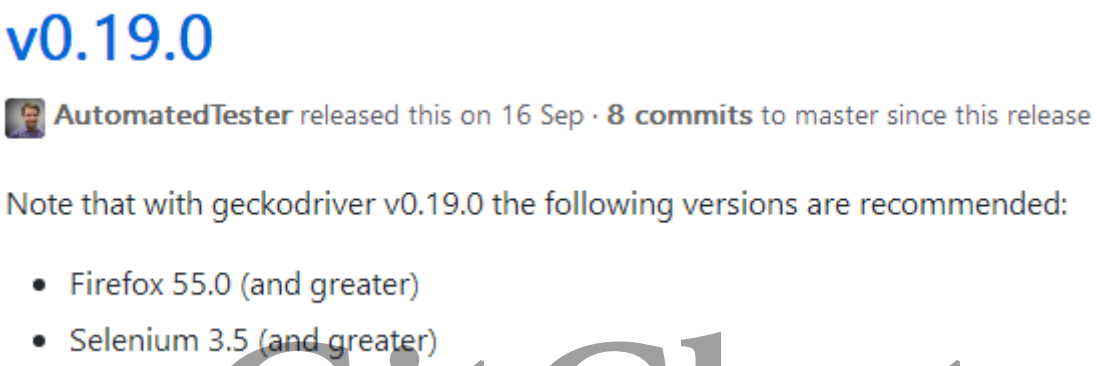


因为我这里已经装好了 所以cmd显示的输出会和你们不一样。

（6）FireFox对应的Selenium驱动程序

下载链接：驱动下载地址

注意FireFox和Selenium版本对应 笔者在安装这个驱动的时候走了不少弯路。



这个是笔者使用的版本。

还有**以下几点一定要注意**，那就是下载好的驱动程序请解压到浏览器所在文件夹目录 并且复制好路径到环境变量path。并且请把Firefox浏览器的.exe文件的路径也复制到环境变量path,把驱动文件拷贝一份放在你Python2.7的目录下。这样才能正常建立浏览器和驱动的联系。

Mac 和 linux 可以参考这篇回答（windows也有介绍）：关于驱动安装失败常见解答

二　正文内容

其实Python也有识别验证码的库，这里给大家推荐pytesseract库，一般的验证码都能解决，有兴趣的朋友可以去了解

可以参考这篇文章：Pytesseract库识别验证码

那么，为什么建设网站的人要检测爬虫呢？你会想，不都是访问网页吗？但是，我们需要知道，使用爬虫会给网站的服务器带来不少负担，影响服务器性能。而且，爬虫并不是真正的人，不是真正的客户，这当然不被建站人喜欢。而且爬虫爬取来的数据最好不要用于商业用途，不然会遇上法律纠纷的。

这里有一篇文章是关于爬虫使用不当的案例：爬虫使用不当法律纠纷文章

因此，我们写爬虫的人，应该站在建站人的角度思考。尽可能在不影响服务器使用的情况下获取需要的数据。

爬虫根据爬取的数据的不同，可以分为静态的爬虫和动态的爬虫。有些网页只是一个简单的web网页，数据不会动态更新，像百度百科、csdn的博文等等，展示一个网页。单纯只有静态数据的web网页已经不多了。所以静态的爬虫实用性很差。

有些数据则不同，他是动态的，像淘宝里的评论区里的数据，b站里的评论区数据，动态加载。那么静态爬虫就不够用了。那么这时我们就需要我们的法宝–Selenium了

2. 什么是 Selenium？Selenium工具安装（基于 Firefox 浏览器）

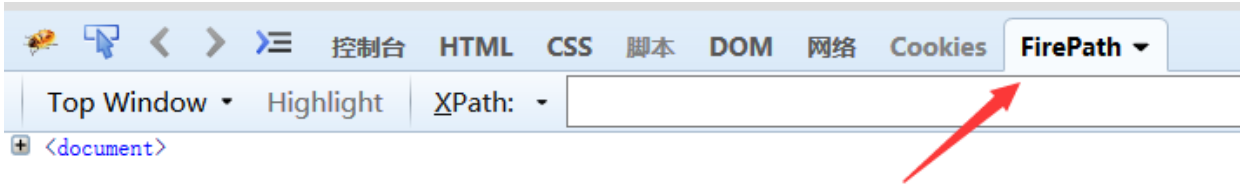
Selenium是一个浏览器自动化测试框架。本来是作为web应用程序测试的工具。它可以直接运行在浏览器里，模仿真正的用户操作。目前支持IE、Firefox、Safari、Chrome大多数主流浏览器。以前是不用驱动的，现在如果要使用Selenium必须要安装对应浏览器的驱动。也就是说，使用Selenium，爬虫能够更像人的行为，去访问网页，从而获取到有用的信息。

有下面几点好处：一、爬虫的行为更接近人使用浏览器时的操作，降低了被服务器发现的可能。二、对于有些动态数据，比如需要用户下拉刷新才出来的数据，普通的静态爬虫是无可奈何的。那么使用Selenium模拟用户行为，下拉滚动条就可以把隐藏的数据获取到了

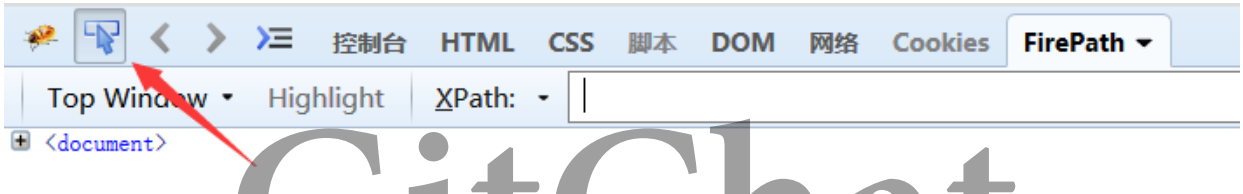
（1）打开Firefox，在插件的界面选择启用FirePath。

这里我已经启用了，所以按钮显示的是禁用。如果你是第一次使用，按钮上显示的应该是启用。

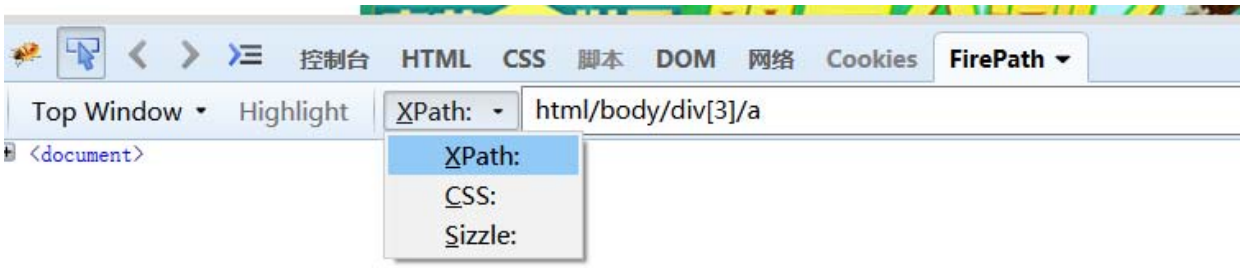
（2）在Firefox打开你需要爬取数据的网页 键盘按F12。



（3）



（4）使用鼠标在网页处点击你需要的获取的数据 比如标题，你会发现FirePath已经自动为你生成对应的xPath定位代码。



Xpath和css定位是比较好的定位方式了。

4. 实例操作：爬取 B 站评论

接下来，通过实战的方式来学习一下，具体如何Selenium去爬取B站的评论。实战以前，介绍一些要用到的方法。

（1）

```
fp = webdriver.Firefox() #获得基于Firefox的对象
fp.set_preference("permissions.default.stylesheet",2)
fp.set_preference("permissions.default.image",2)
```

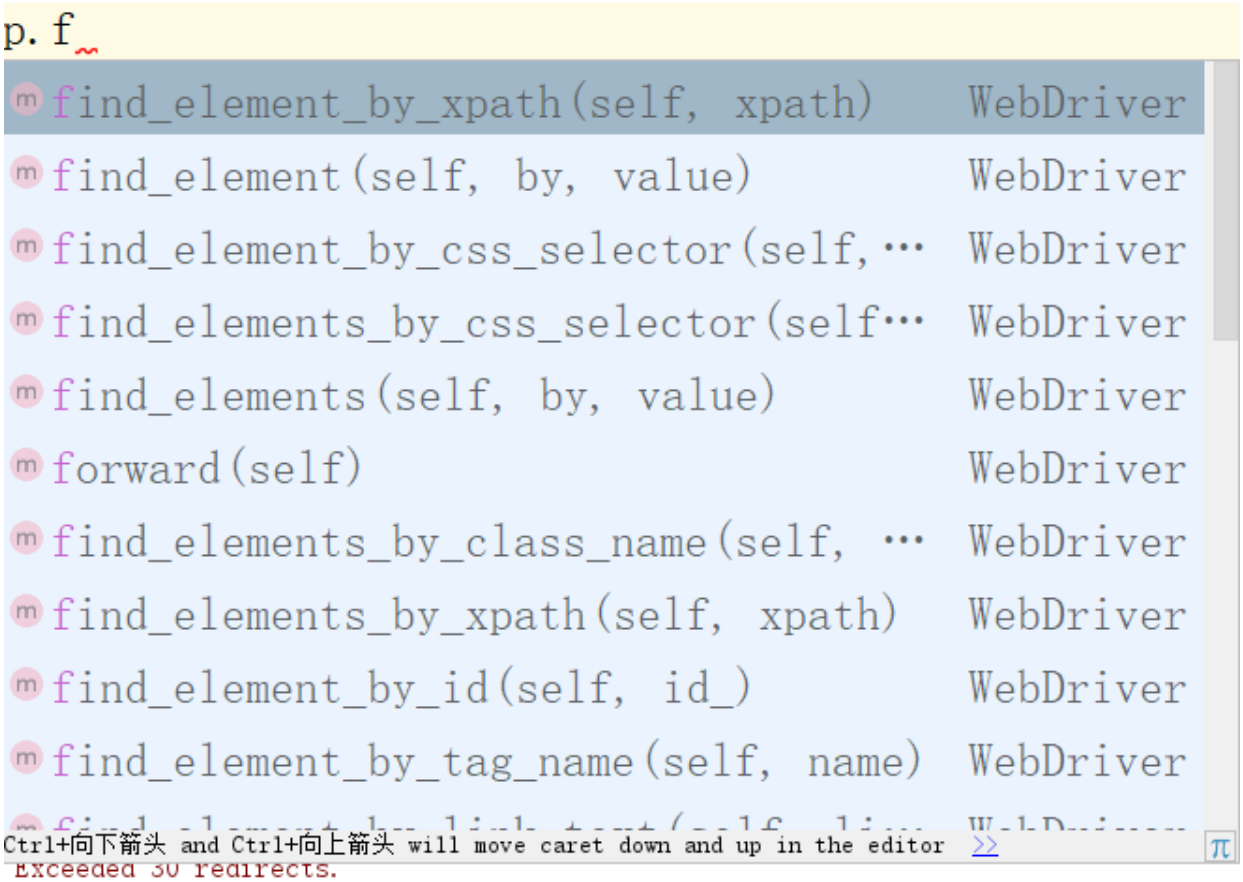
这个是对Selenium自动化测试的配置。可以选择不加载图片、css渲染、禁止使用Javascript目的是为了加快网页的加载。这里根据需要自由选择。第二个参数填2表示禁用。更多用法可以到Selenium官网查看文档

Selenium官网

如果进不去，代表你需要一些特殊的工具。这里不做介绍。

（2）

```
target = app.find_element_by_xpath(".//*
[@id='recommend_report']/div[1]/span")
```



读者可以试试其他方法。

值得注意的是，find_elements_by_xpath和find_element_by_xpath一个有s一个没有。前者返回一个数组，后者返回一个元素。其他方法同理。这里推荐xpath与css两种方法，比较精准。

(3)

```
1.app.execute_script("arguments[0].scrollIntoView();", target)#定位到特定的元素
2.time.sleep(3)
```

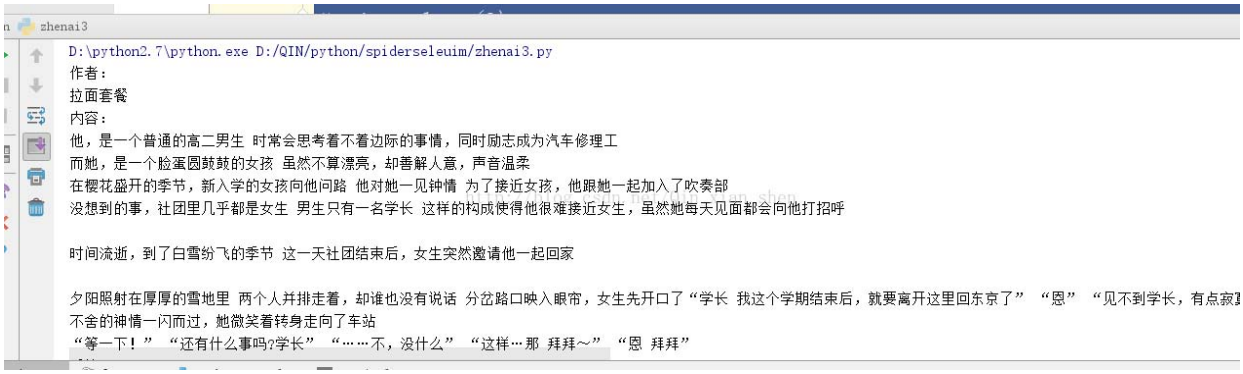
这里执行script语句，去定位到我们要到的位置，模拟滚动条下拉。但是值得一提的是每

```
7.sys.setdefaultencoding("utf-8")
8.
9.
10.app = webdriver.Firefox()
11.app.get("https://www.bilibili.com/video/av3553625/?
from=search&seid=10292605247919873793")
12.
13.target = app.find_element_by_xpath(".//*
[@id='recommend_report']/div[1]/span")
14.app.execute_script("arguments[0].scrollIntoView();", target)#
定位到特定的元素
15.time.sleep(3)
16.
17.target2 = app.find_element_by_xpath(".//*
[@id='bbComment']/div[1]/div[4]/div[4]/span/a");
18.app.execute_script("arguments[0].scrollIntoView();", target2)
19.time.sleep(3)
20.target2.click()
21.
22.for i in range(20):
23.    if(i==7):
24.        continue
25.    name = app.find_element_by_xpath(".//*
[@id='bbComment']/div[1]/div[4]/div["+str(i+1)+"]/div[2]/div[1]/a
[1]")
26.    test = app.find_element_by_xpath(".//*
[@id='bbComment']/div[1]/div[4]/div["+str(i+1)+"]/div[2]/p")
27.    if (i != 13 and i != 17):
28.        pinglun1 = app.find_element_by_xpath(".//*
[@id='bbComment']/div[1]/div[4]/div["+str(i+1)+"]/div[2]/div[3]/d
iv[1]/div/div[1]/span")
29.    if (i != 13 and i != 17):
30.        pinglun2 = app.find_element_by_xpath(".//*
[@id='bbComment']/div[1]/div[4]/div["+str(i+1)+"]/div[2]/div[3]/d
iv[2]/div/div[1]/span")
31.    if(i !=12 and i !=13 and i != 17):
32.        pinglun3 = app.find_element_by_xpath(".//*
[@id='bbComment']/div[1]/div[4]/div["+str(i+1)+"]/div[2]/div[3]/d
```

```
45.     time.sleep(3)
46.app.quit()
```

因为有些评论区的数据是有的，有些是没有的。我们就在for循环里加了一个if判断。如果那一层的评论没有，就Continue跳过就好。

效果图如下：



这些数据是评论区的精彩热评，我爬下了作者的用户名ID 评论内容 和这个评论的跟帖评论。加以整理。

值得一提的是，如果Selenium能做的还远远不止这些，还能模拟点击事件，键盘的输入事件。这个给大家留一个思考题，尝试使用Selenium模拟登陆B站。

提示：使用方法app.click() app.sendkey() app.clear()

参考文章：selenium自动化登陆操作

5. 拓展：介绍 Tor 浏览器，实现匿名 IP 爬虫，防止 IP 封禁

在爬虫过程中，如果操作不当，被服务器监测到，就有可能导致自己的IP被网站封禁。在一定时间内，拒绝访问。那么有什么办法可以解决呢？

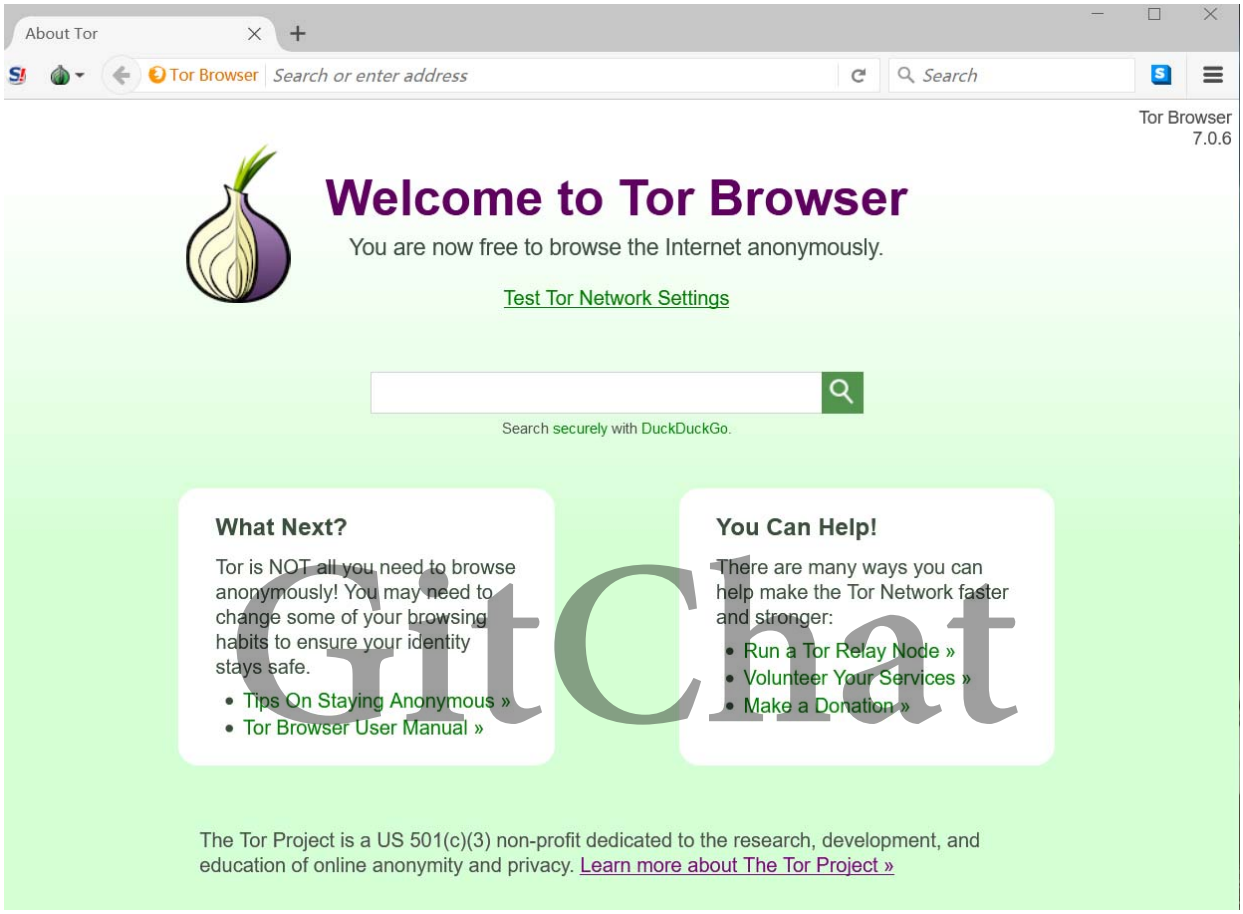
再向大家推荐一款神器，Tor浏览器。这个浏览器，据说这个浏览器本来是
美国军方用来获取信息的工具，能够匿名IP，把自己的的真实IP给隐藏，使用别人的ip

（3）Python安装相应的库

```
Pip install pysocks
Pip install stem
```

（4）使用方法

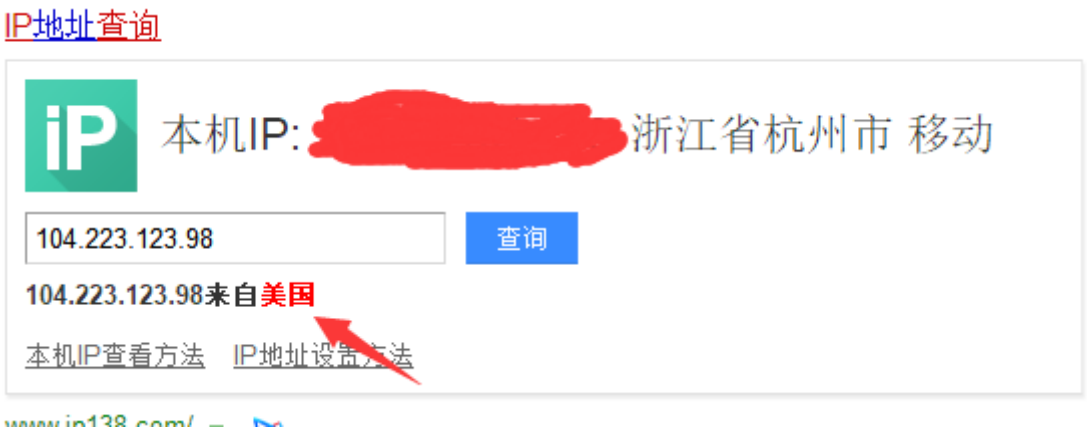
先打开Tor浏览器 再运行程序：




```
import socks
import socket
import requests
```

```
:Pachong
D:\python2.7\python.exe D:/QIN/python/spiderseleuim/TorPachong.py
104.223.123.98


Process finished with exit code 0
```

用百度查找IP ：



（5）切换IP

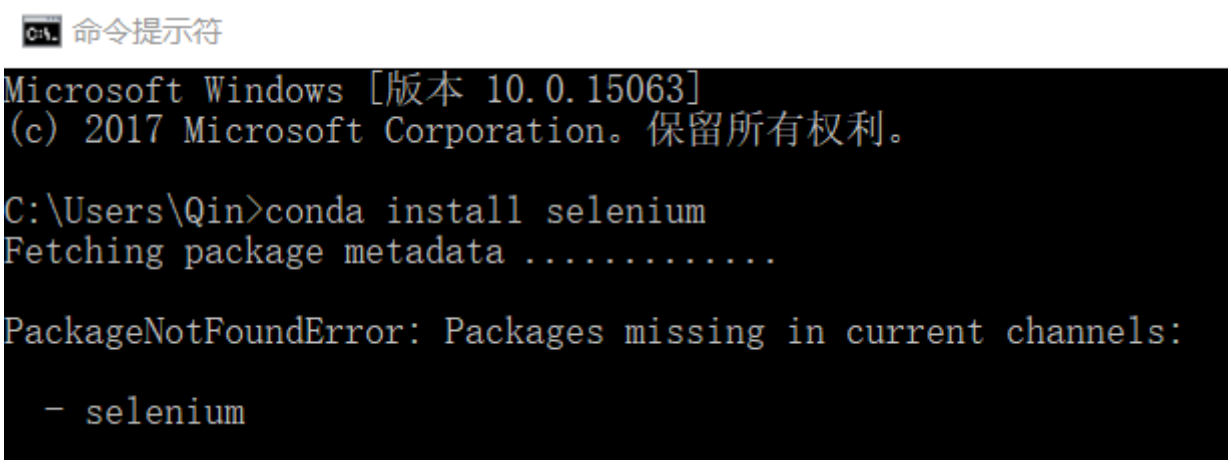

```
1.#coding=utf-8
2.from stem import Signal
3.from stem.control import Controller
4.import socket
5.import socks
6.import requests
7.import time
8.import sys
9.reload(sys)
10.sys.setdefaultencoding('utf-8')
11.
12.controller = Controller.from_port(port=9151)
```

```
28.     print ("第"+str(x+1)+"次抓取花费时间：  "+str(time2-time1))
29.
30.     time3 = time.time()
31.     controller.signal(Signal.NEWNYM)
32.     time.sleep(5)
33.     time4 = time.time()
34.     total_changeIP_time = total_changeIP_time + time4-time3-5
35.     print ("第"+str(x+1)+"次更换IP花费时间：  "+str(time4-time3-
5))
36.
37.print ("平均抓取花费时间：  "+str(total_scrappy_time/10))
38.print ("平均更换IP时间：  "+str(total_changeIP_time/10))
```

```
D:\python2.7\python.exe D:/QIN/python/spiderseleuim/torpachong2.py
第1次IP：185.170.42.18

第1次抓取花费时间：5.45399999619
第1次更换IP花费时间：0.00499987602234
第2次IP：199.249.223.41

第2次抓取花费时间：6.59599995613
第2次更换IP花费时间：0.00300002098083
第3次IP：51.15.134.120

第3次抓取花费时间：4.69000005722
第3次更换IP花费时间：0.0019998550415
第4次IP：18.248.2.85

第4次抓取花费时间：5.42200016975
```


6. 可能提到到的问题

为什么使用Python2.7而不是Python3?为什么不用更好的Anaconda而是使用pip下载工



7. 推荐资料

- 《Python网络爬虫从入门到实践》-唐松（非常赞的一本书 17年刚刚出版）
- 《Selenium2 自动化测试实战》-虫师（推荐虫师的博文，非常不错）

有兴趣的朋友可以了解Scrapy框架，爬虫非常好用，实用。爬虫效率会得到大大提升。

8. 写在最后的话

真的真的非常感谢各位能够来参加这场chat,这是我第一次做chat,感谢各位的支持。感激不尽。如果本文能给你带来些许帮助，这真是我的荣幸。感谢。

松爱家的小秦