day04-课堂笔记

1. lxml的使用

1.1 lxml模块的介绍与安装

前面我们在学习xpath的时候，使用的是 浏览器中的一个插件来学习xpath语法的，但是呢，我们要写的是 一个爬虫的程序，也就是说后去的使用xpath语法还是要在 Python代码中去使用。

如何在Python代码中使用xpath来获取数据？ 就要去借助 lxml模块了，

lxml 就是可以帮助我们在Python代码中去书写xpath语法去提取数据的模块。

lxml是一个第三方模块，在使用之前要先进行安装

```
pip install lxml
```

1.2 lxml的基本使用

使用步骤：

- 导包： from lxml import etree
- 使用 element = etree.HTML(响应的html的字符串类型/响应的html的bytes类型)
- 可以使用 element.xpath("在浏览器中是如何写的xpath就在这里怎么写") 返回的数据是一个列表

to_string() 方法的使用：

- etree.HTML() 可以将 html字符串转换为 element对象
- to_string() 可以将element对象转换为 字符串。

```
from lxml import etree
```

```
2
3  html_str = """
4  <ul>
5    <li class="item-1"><a href="baidu.com">百度一下</a></li>
6    <li class="item-2"><a href="sina.com">新浪</a></li>
7    <li class="item-3"><a href="163.com">网易</a></li>
8    <li class="item-4"><a href="google.com">谷歌</a></li>
9    <li class="item-5"><a href="qq.com">腾讯</a>
10 </ul>
11 """
12
13 # 将 响应的内容 转换为 element对象
14 element = etree.HTML(html_str)
15
16 print(element)
17
18 # 现在已经将 html 字符串 转换为了 element对象，  那么我们如何将这个element 对象 再转换回 str
19 print(etree.tostring(element).decode())
```

基本使用

```
1  from lxml import etree
2
3  # //a/text()
4  html_str = """
5  <ul>
6    <li class="item-1"><a href="baidu.com">百度一下</a></li>
7    <li class="item-2"><a href="sina.com">新浪</a></li>
8    <li class="item-3"><a href="163.com">网易</a></li>
9    <li class="item-4"><a href="google.com">谷歌</a></li>
10   <li class="item-5"><a href="qq.com">腾讯</a>
11 </ul>
12 """
13
14 # 将 响应的内容 转换为 element对象
15 element = etree.HTML(html_str)
16
17 # print(element)
18 #
19 # # 现在已经将 html 字符串 转换为了 element对象，  那么我们如何将这个element 对象 再转换回 str
20 # print(etree.tostring(element).decode())
21
22 # 需求：  获取到所有的 网址的标题   [百度一下, 新浪, xx, x, xx]
23 title_list = element.xpath("//a/text()")
24 print(title_list)
```

```
1  from lxml import etree
2  # pretty
3  from pprint import pprint
4
5  # //a/text()
6  html_str = """
7  <ul>
8    <li class="item-1"><a href="baidu.com">百度一下</a></li>
```

```
  <li class="item-2"><a href="sina.com">新浪</a></li>
  <li class="item-3"><a href="163.com">网易</a></li>
  <li class="item-4"><a href="google.com">谷歌</a></li>
  <li class="item-5"><a href="qq.com">腾讯</a>
</ul>
"""

# 将 响应的内容 转换为 element对象
element = etree.HTML(html_str)

# print(element)
#
# # 现在已经将 html 字符串 转换为了 element对象，  那么我们如何将这个element 对象 再转换回 str
# print(etree.tostring(element).decode())

# 需求： 获取到所有的 网址的标题   [百度一下, 新浪, xx, x, xx]
title_list = element.xpath("//a/text()")
print(title_list)

# 需求： 获取到所有的 网站对应的  具体的url地址
# 获取到所有的a标签中的 href属性的值.
href_list = element.xpath("//a/@href")
print(href_list)

# 需求： 经过上述的操作，我们已经获取到了所以迤网址的名字，网站的具体的url地址
# 数据要合到一起。 数据的格式要按照以下的格式来合并
"""
[
 {"title": "百度一下", "href": "baidu.com"},
 {"title": "新浪", "href": "sina.com"},
 {"title": "网易", "href": "163.com"}
]
"""
# 先去定义一个列表
data_list = []
# 遍历 title_list
for title in title_list:
    # 准备一个空字典
    item = {}
    # 遍历的是 title_list ，可以 获取一下 当前遍历的时候 title在 title_list 中的下标
    index = title_list.index(title)
    print(index)
    item["title"] = title
    item["href"] = href_list[index]
    data_list.append(item)
pprint(data_list)
```

.

1.3 lxml的进阶使用

先分组 再提取

```
from lxml import etree
# pretty
from pprint import pprint
```

```
4
5  # //a/text()
6  html_str = """
7  <ul>
8    <li class="item-1"><a href="baidu.com">百度一下</a></li>
9    <li class="item-2"><a href="sina.com">新浪</a></li>
10   <li class="item-3"><a href="163.com">网易</a></li>
11   <li class="item-4"><a href="google.com">谷歌</a></li>
12   <li class="item-5"><a>腾讯</a>
13 </ul>
14 """
15 # [<a href="baidu.com">百度一下</a>, <a href="baidu.com">新浪</a>, <a
   href="baidu.com">网易</a>]
16 """
17 [
18  {"title": "百度一下", "href": "baidu.com"},
19  {"title": "新浪", "href": "sina.com"},
20  {"title": "网易", "href": "163.com"}
21 ]
22 """
23
24 # 将 响应的内容 转换为 element对象
25 element = etree.HTML(html_str)
26
27 # 不去获取两个列表了。 可以将数据进行分组，分组完毕之后 我再去获取每一条数据。
28 # 先分组： 先把所有的a标签获取到
29 a_list = element.xpath("//a")
30 # 准备一个空列表
31 data_list = []
32 # 遍历
33 for a in a_list:
34     # 准备一个空字典，
35     item = {}
36     # 遍历出来的每一个a 是一个 element对象  可以继续去调用xpath方法
37     # . 表示的就是当前 我们遍历出来的每一个 a标签
38     item["title"] = a.xpath("./text()")[0]
39     href = a.xpath("./@href")
40     # 使用三目运算
41     item["href"] = href[0] if len(href)>0 else None
42     # if len(href) > 0:
43     #     item["href"] = href[0]
44     # else:
45     #     item["href"] = None
46     print(item)
```

2. 案例

需求： 要求大家去抓取 https://www.sxpi.edu.cn/xwzx/gyyw.htm 这个url地址下的 所有的新闻的标题以及发布日期

数据的格式：

```
[
    {"title": "陕西工院“三全”举措做好迎新工作", "time": "2021/09/02"},
    {"title": "陕西工院“三全”举措做好迎新工作", "time": "2021/09/02"},
    {"title": "陕西工院“三全”举措做好迎新工作", "time": "2021/09/02"},
]
```

保存数据，将整个数据的列表保存到一个 data.json

```python
import requests, json
from lxml import etree


# 1. 准备url地址
url = "https://www.sxpi.edu.cn/xwzx/gyyw.htm"
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36"
}

# 2. 发送请求 获取响应
response = requests.get(url, headers=headers)

# 3. 提取数据
# 要将响应的内容 转换为 element对象
element = etree.HTML(response.content)
# 先分组 再提取
li_list = element.xpath("//div[@class='list_box']/ul/li")
# 遍历每一个li标签 去获取每一条数据
data_list = []
for li in li_list:
    item = {}
    # 获取新闻的标题
    item["title"] = li.xpath("./a/text()")[0]
    # 获取新闻的发布日期
    item["time"] = li.xpath("./span/text()")[0]
    data_list.append(item)

# 4. 保存数据
with open("data.json", "w", encoding="utf-8") as f:
    json.dump(data_list, f, ensure_ascii=False, indent=2)
```

3. selenium的介绍和环境搭建

selenium 是一个web自动化测试的工具，最开始设计出来的时候，是专门用于软件测试，可以接受一些指令，操作浏览器： 打开某个页面，获取页面中的数据

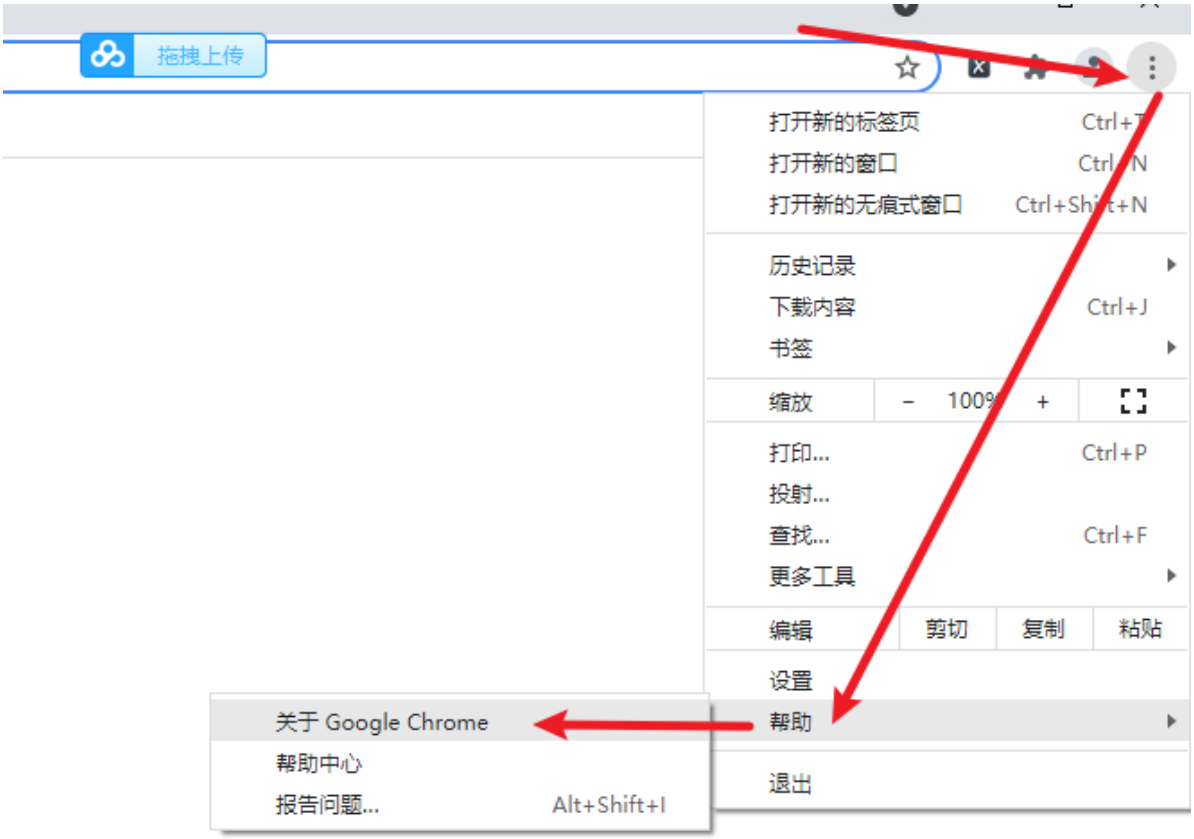
并且可以去操作页面上的标签，截屏。

分类：

- 有头模式：指的是有界面。操作我们是可以看到的到的。
- 无头模式：指的就是没有界面，操作都是在内存中完成，操作步骤我们是看不到的。

3.1 环境搭建

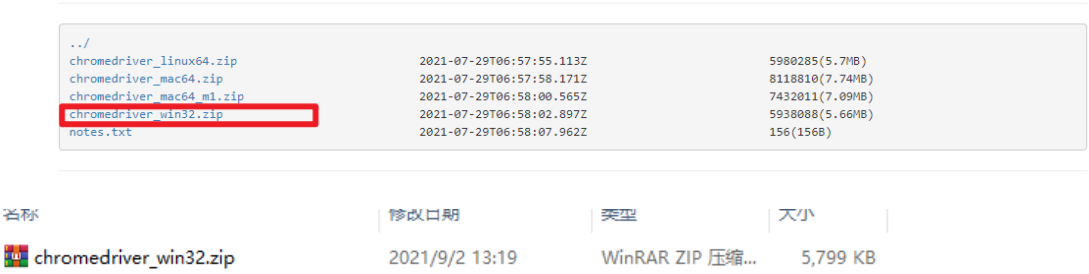
① 打开自己的谷歌浏览器，查看一下自己谷歌浏览器的版本号



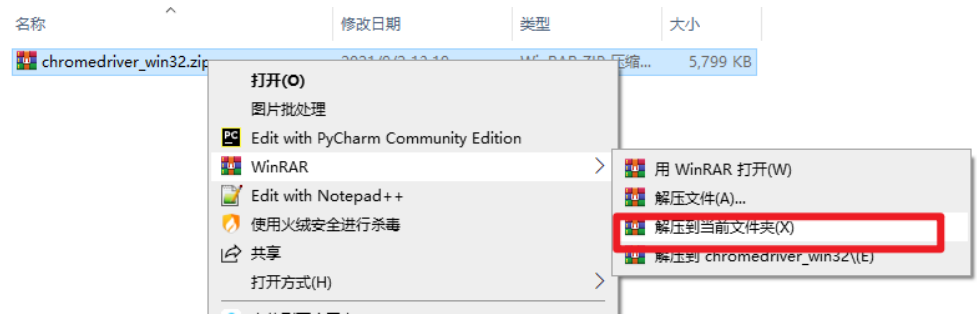
、

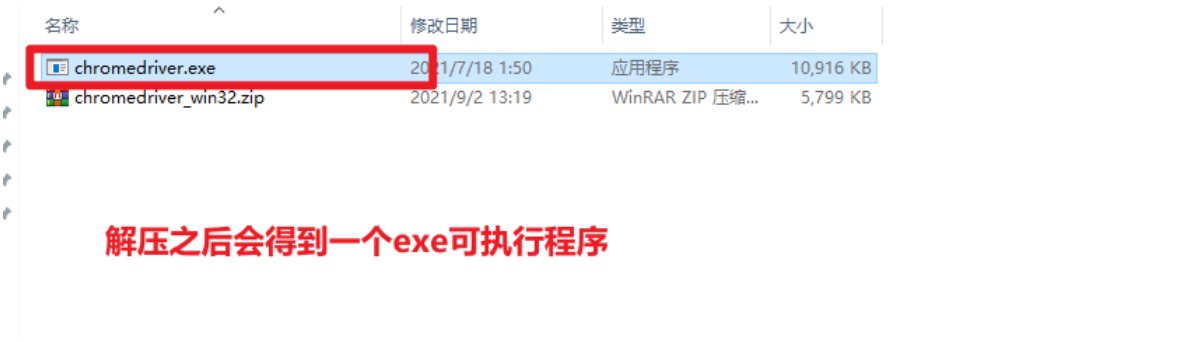
② 打开网址： http://npm.taobao.org/mirrors/chromedriver/ ，下载驱动程序，帮助我们去操作浏览器。

③ 下载对应操作系统的驱动， win

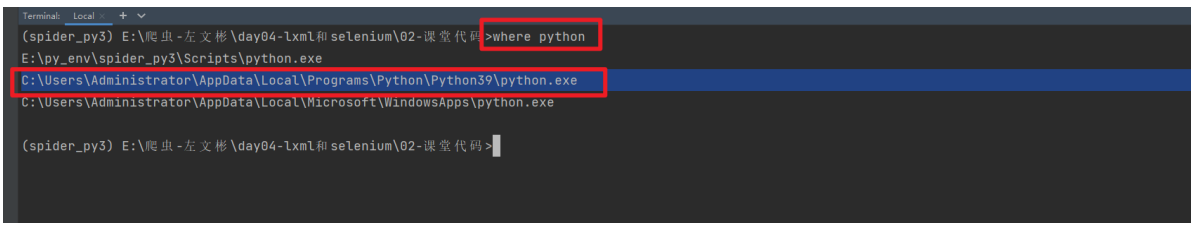


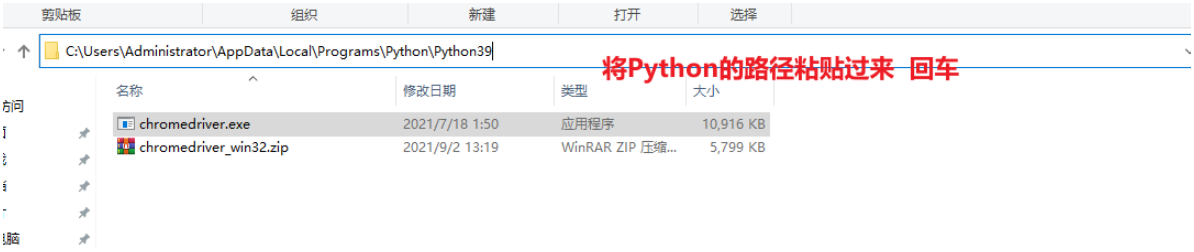
④ 解压这个压缩包

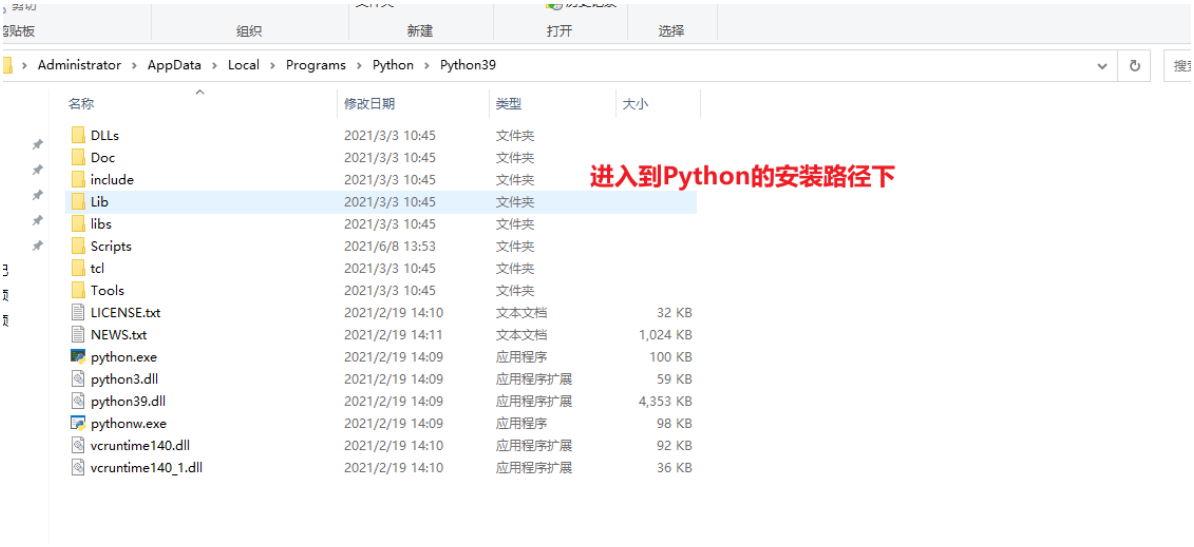




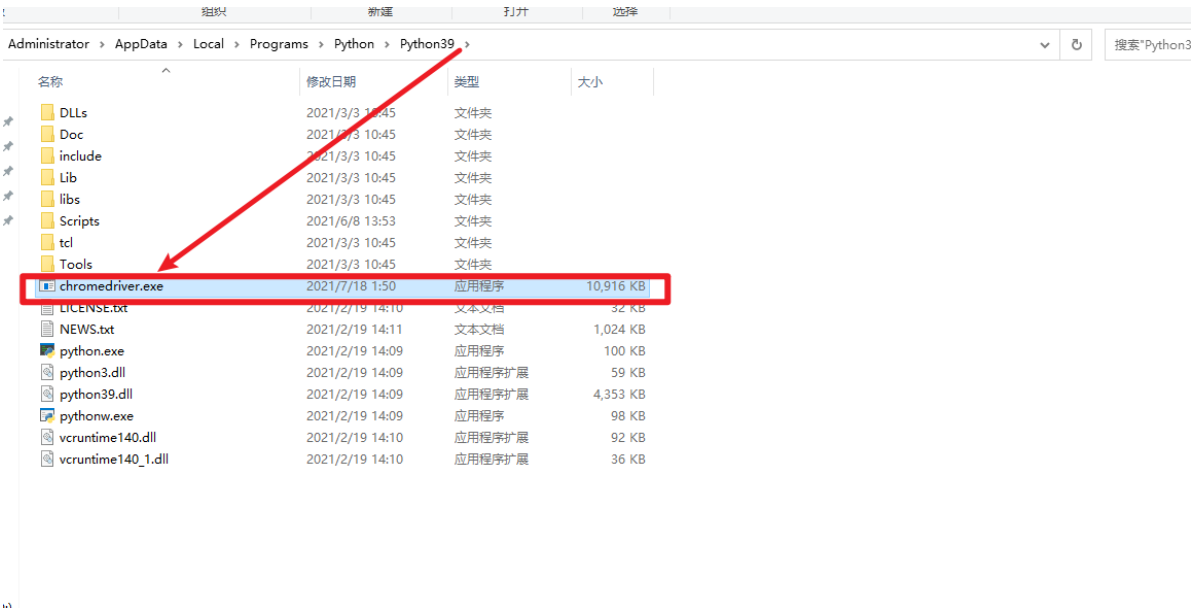
⑤ 找到自己的电脑上Python的安装路径





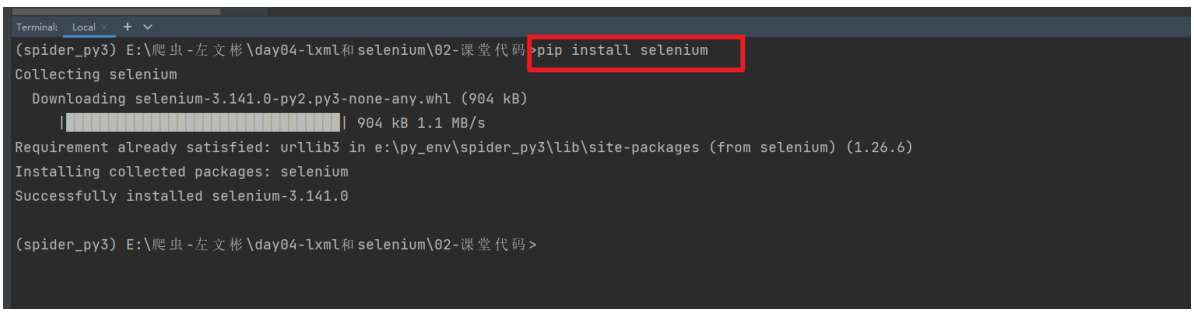


⑥ 将刚才解压出来的 chromedriver.exe 文件 复制 粘贴到 Python的安装路径下



⑦ 在Python环境中 安装一个模块， selenium

```
pip install selenium
```



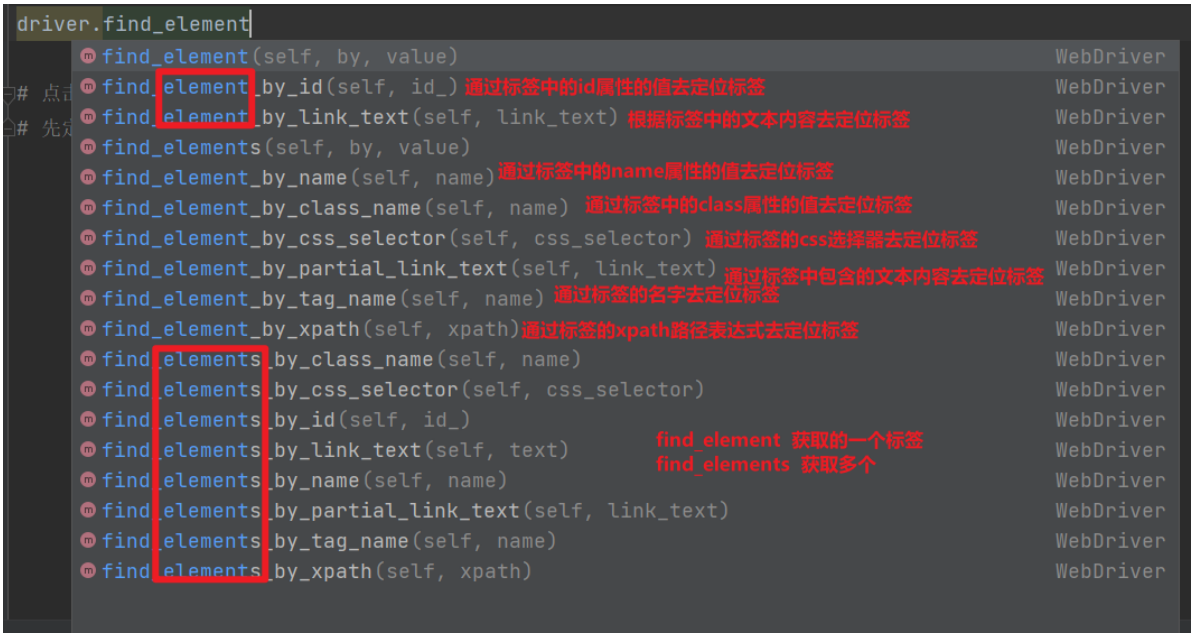
4. selenium的基本使用

操作步骤：

- 导包： from selenium import webdriver
- 创建driver对象： driver = webdriver.Chrome()
- 调用get方法传入一个url地址，相当于使用浏览器打开了一个网站： driver.get("http://www.baidu.com")
- 调用 dirver对象中的方法，进行对页面的操作。
- 所有的操作操作完毕之后，记得退出浏览器： driver.quit()

```python
# 1.导包
import time

from selenium import webdriver

# 2. 创建driver对象
driver = webdriver.Chrome()

# 窗口最大化
driver.maximize_window()

# 3. 调用get方法传入url地址 就可以打开这个网站
driver.get("https://www.sxpi.edu.cn/")

# 截图
driver.save_screenshot("sxpi.png")

time.sleep(5)

# 4. 操作都完毕之后，退出浏览器
driver.quit()
```

5. selenium定位标签和获取数据的方法




```python
import time

from selenium import webdriver

# 1. 创建driver对象
driver = webdriver.Chrome()

# 2. 打开页面
driver.maximize_window()
driver.get("http://www.baidu.com")

# 向百度的输入框中输入内容： 学校的名字
# 想要向输入框中输入内容，先定位到输入框  定位到之后 再去输入内容
# send_keys("字符串") 表示向 输入框中输入文本内容
# driver.find_element_by_id("kw").send_keys("abcdefg")
# driver.find_element_by_name("wd").send_keys("abc")
time.sleep(5)
driver.find_element_by_class_name('s_ipt').send_keys('abc')
time.sleep(5)

# 点击 搜索的按钮
# 先定位到 搜索的按钮， 再去点击
driver.find_element_by_id('su').click()

time.sleep(3)

# 退出浏览器
driver.quit()
```

```python
import time

from selenium import webdriver

# 1. 创建driver对象
driver = webdriver.Chrome()


# 2. 打开页面
```

```
driver.get("https://www.jd.com/")

# 3. 操作： 向 输入框中输入  笔记本电脑  点击搜索
# 定位到输入框
driver.find_element_by_xpath('//input[@id="key"]').send_keys("笔记本电脑")
# driver.find_element_by_id('key')
time.sleep(5)

# 点击搜索按钮
driver.find_element_by_xpath("//button[@class='button']").click()
time.sleep(3)

# 截图
driver.save_screenshot('jd.png')

# 退出
driver.quit()
```

6. selenium的案例

爬虫：模拟客户端发送请求 获取响应。

selenium：打开浏览器 打开页面（发送请求 获取响应）

需求：使用selenium去完成爬虫的编写，

- 目标网址： https://wz.sun0769.com/political/index/politicsNewest
- 数据内容：
  - 编号
  - 状态
  - 标题
  - 发布时间
  - 响应时间
- 数据的格式：`[{},{}.{}}]`

```
from selenium import webdriver
import json

# 1. 准备url地址
url = "https://wz.sun0769.com/political/index/politicsNewest"

# 2. 发送请求  获取响应(使用selenium打开页面)  要求的是使用selenium
# 创建一个driver对象
driver = webdriver.Chrome()
driver.maximize_window()
driver.get(url)

# 3. 提取数据
# 先使用driver定位到所有的 包含数据的标签 li
# 使用 find_elemnt 返回的是 WebElement 对象
# 先分组
li_list = driver.find_elements_by_xpath("//ul[@class='title-state-ul']/li")
# 准备空列表 用于保存每一条数据（字典）
data_list = []
# 遍历获取每一个li里面的数据的内容，获取每一条数据
for li in li_list:
```

```
    item = {}
    # 在遍历的时候 获取数据的时候因为是一条数据， 所以这个地方我们要使用 find_element
    # 注意点： 在selenium中使用xpath的时候 只能写到定位标签 不能在xpath中书写 text() 以及 @属性名
    # 在selenium中 要获取标签中的文本内容要使用 text来获取，
    item["id"] = li.find_element_by_xpath("./span[1]").text
    item["status"] = li.find_element_by_xpath("./span[2]").text
    item["title"] = li.find_element_by_xpath("./span[3]/a").text
    # get_attribute("属性名")  表示根据属性的名字去获取标签中的属性的值。
    item["detail_href"] = li.find_element_by_xpath("./span[3]/a").get_attribute("href")
    item["response_time"] = li.find_element_by_xpath("./span[4]").text
    item["release_time"] = li.find_element_by_xpath("./span[5]").text
    data_list.append(item)
# 4. 保存数据 保存到 sun.json
with open('sun.json', "w", encoding="utf-8") as f:
    json.dump(data_list, f, ensure_ascii=False, indent=2)


# 退出浏览器
driver.quit()
```

网站： https://www.qiushibaike.com/text/