EDU

CSDN学院　IT实战派

动态网页爬虫

讲师：黄老师

本章目标

1. 学会动态网页爬虫。
2. 学会selenium库使用。

什么是动态网页爬虫

1. 动态网页，是网站在不重新加载的情况下，通过ajax技术动态更新网站中的局部数据。比如拉勾网的职位页面，在换页的过程中，url是没有发生改变的，但是职位数据动态的更改了。
2. AJAX（Asynchronouse JavaScript And XML）异步JavaScript和XML。前端与服务器进行少量数据交换，Ajax 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。传统的网页（不使用Ajax）如果需要更新内容，必须重载整个网页页面。因为传统的在传输数据格式方面，使用的是XML语法。因此叫做AJAX，其实现在数据交互基本上都是使用JSON。使用AJAX加载的数据，即使使用了JS，将数据渲染到了浏览器中，在右键->查看网页源代码还是不能看到通过ajax加载的数据，只能看到使用这个url加载的html代码。

动态网页爬虫的解决方案

1. 直接分析ajax调用的接口。然后通过代码请求这个接口。
2. 使用Selenium+chromedriver模拟浏览器行为获取数据。

方式	优点	缺点
分析接口	直接可以请求到数据。不需要做一些解析工作。代码量少，性能高。	分析接口比较复杂，特别是一些通过js混淆的接口，要有一定的js功底。容易被发现是爬虫。
selenium	直接模拟浏览器的行为。浏览器能请求到的，使用selenium也能请求到。爬虫更稳定。	代码量多。性能低。

Selenium+chrome介绍

Selenium相当于是一个机器人。可以模拟人类在浏览器上的一些行为，自动处理浏览器上的一些行为，比如点击，填充数据，删除cookie等。chromedriver是一个驱动Chrome浏览器的驱动程序，使用他才可以驱动浏览器。当然针对不同的浏览器有不同的driver。以下列出了不同浏览器及其对应的driver：

1. Chrome：https://sites.google.com/a/chromium.org/chromedriver/downloads
2. Firefox：https://github.com/mozilla/geckodriver/releases
3. Edge：https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/
4. Safari：https://webkit.org/blog/6900/webdriver-support-in-safari-10/

Selenium的官方网址：https://selenium-python.readthedocs.io/

安装selenium和webdriver

1. 安装Selenium：Selenium有很多语言的版本，有java、ruby、python等。我们下载python版本的就可以了。pip install selenium
2. 安装chromedriver：下载完成后，放到不需要权限的纯英文目录下就可以了。
3. Chromedriver的驱动：
   https://sites.google.com/a/chromium.org/chromedriver/downloads

Selenium的基本使用

现在以一个简单的获取百度首页的例子来讲下Selenium和chromedriver如何快速入门：

```
from selenium import webdriver

# chromedriver的绝对路径
driver_path = r'D:\ProgramApp\chromedriver\chromedriver.exe'

# 初始化一个driver，并且指定chromedriver的路径
driver = webdriver.Chrome(executable_path=driver_path)
# 请求网页
driver.get("https://www.baidu.com/")
# 通过page_source获取网页源代码
print(driver.page_source)
```

Selenium的基本使用-关闭

1. driver.close()：关闭当前页面。
2. driver.quit()：退出整个浏览器。

Selenium的基本使用-定位元素

1. find_element_by_id：根据id来查找某个元素。
2. find_element_by_class_name：根据类名查找元素。
3. find_element_by_name：根据name属性的值来查找元素。
4. find_element_by_tag_name：根据标签名来查找元素。
5. find_element_by_xpath：根据xpath语法来获取元素。
6. find_element_by_css_selector：根据css选择器选择元素。

要注意，find_element是获取第一个满足条件的元素。find_elements是获取所有满足条件的元素。

Selenium的基本使用-操作表单元素

1. 操作输入框：分为两步。第一步：找到这个元素。第二步：使用send_keys(value)，将数据填充进去。

   示例代码如下：

   ```
   inputTag = driver.find_element_by_id('kw')
   inputTag.send_keys('python')
   ```

2. 操作checkbox：因为要选中checkbox标签，在网页中是通过鼠标点击的。因此想要选中checkbox标签，那么先选中这个标签，然后执行click事件。示例代码如下：

   ```
   rememberTag = driver.find_element_by_name("rememberMe")
   rememberTag.click()
   ```

Selenium的基本使用-操作表单元素

3. 选择select：select元素不能直接点击。因为点击后还需要选中元素。这时候selenium就专门为select标签提供了一个类selenium.webdriver.support.ui.Select。将获取到的元素当成参数传到这个类中，创建这个对象。以后就可以使用这个对象进行选择了。示例代码如下：

```
from selenium.webdriver.support.ui import Select
# 选中这个标签，然后使用Select创建对象
selectTag = Select(driver.find_element_by_name("jumpMenu"))
# 根据索引选择
selectTag.select_by_index(1)
# 根据值选择
selectTag.select_by_value("http://www.95yueba.com")
# 根据可视的文本选择
selectTag.select_by_visible_text("95秀客户端" )
```

Selenium的基本使用-操作表单元素

4. 操作按钮：操作按钮有很多种方式。比如单击、右击、双击等。这里讲一个最常用的。就是点击。直接调用click函数就可以了。示例代码如下：

```
inputTag = driver.find_element_by_id('su')
inputTag.click()
```

Selenium的基本使用-行为链

有时候在页面中的操作可能要有很多步，那么这时候可以使用鼠标行为链类selenium.webdriver.common.action_chains.ActionChains来完成。比如现在要将鼠标移动到某个元素上并执行点击事件。那么示例代码如下：

```
inputTag = driver.find_element_by_id('kw')
submitTag = driver.find_element_by_id('su')
actions = ActionChains(driver)
actions.move_to_element(inputTag)
actions.send_keys_to_element(inputTag,'python')
actions.move_to_element(submitTag)
actions.click(submitTag)
actions.perform()
```

还有更多的鼠标相关的操作。

click_and_hold(element)：点击但不松开鼠标。

context_click(element)：右键点击。

double_click(element)：双击。 更多方法请参考：http://selenium-python.readthedocs.io/api.html

Selenium的基本使用-cookie操作

1. 获取所有的cookie：

```
for cookie in driver.get_cookies():
    print(cookie)
```

2. 根据cookie的key获取value：

```
value = driver.get_cookie(key)
```

3. 删除所有的cookie：

```
driver.delete_all_cookies()
```

4. 删除某个cookie：

```
driver.delete_cookie(key)
```

5. 添加cookie：

```
driver.add_cookie({“name”:”username”,”value”:”abc”})
```

Selenium的基本使用-页面等待

现在的网页越来越多采用了 Ajax 技术，这样程序便不能确定何时某个元素完全加载出来了。如果实际页面等待时间过长导致某个dom元素还没出来，但是你的代码直接使用了这个WebElement，那么就会抛出NullPointer的异常。为了解决这个问题。所以 Selenium 提供了两种等待方式：一种是隐式等待、一种是显式等待。

隐式等待：调用driver.implicitly_wait。那么在获取不可用的元素之前，会先等待10秒中的时间。示例代码如下：

```
driver = webdriver.Chrome(executable_path=driver_path)
driver.implicitly_wait(10)
# 请求网页
driver.get("https://www.douban.com/")
```

Selenium的基本使用-页面等待

显示等待：显示等待是表明某个条件成立后才执行获取元素的操作。也可以在等待的时候指定一个最大的时间，如果超过这个时间那么就抛出一个异常。显示等待应该使用
selenium.webdriver.support.excepted_conditions期望的条件和
selenium.webdriver.support.ui.WebDriverWait来配合完成。示例代码如下：

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
driver = webdriver.Firefox()
driver.get("http://somedomain/url_that_delays_loading")
try:
    element = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "myDynamicElement"))
    )
finally:
    driver.quit()
```

Selenium的基本使用-页面切换

有时候窗口中有很多子tab页面。这时候肯定是需要进行切换的。selenium提供了一个叫做switch_to_window来进行切换，具体切换到哪个页面，可以从driver.window_handles中找到。示例代码如下：

```
# 打开一个新的页面
self.driver.execute_script("window.open('"+url+"')")
# 切换到这个新的页面中
self.driver.switch_to.window(self.driver.window_handles[1])
```

Selenium的基本使用-设置代理

有时候频繁爬取一些网页。服务器发现你是爬虫后会封掉你的ip地址。这时候我们可以更改代理ip。更改代理ip，不同的浏览器有不同的实现方式。这里以Chrome浏览器为例来讲解：

```
from selenium import webdriver

options = webdriver.ChromeOptions()
options.add_argument("--proxy-server=http://110.73.2.248:8123")
driver_path = r"D:\ProgramApp\chromedriver\chromedriver.exe"
driver = webdriver.Chrome(executable_path=driver_path,chrome_options=options)

driver.get('http://httpbin.org/ip')
```

Selenium案例-BOSS直聘爬虫

有时候频繁爬取一些网页。服务器发现你是爬虫后会封掉你的ip地址。这时候我们可以更改代理ip。更改代理ip，不同的浏览器有不同的实现方式。这里以Chrome浏览器为例来讲解：

```
from selenium import webdriver

options = webdriver.ChromeOptions()
options.add_argument("--proxy-server=http://110.73.2.248:8123")
driver_path = r"D:\ProgramApp\chromedriver\chromedriver.exe"
driver = webdriver.Chrome(executable_path=driver_path,chrome_options=options)

driver.get('http://httpbin.org/ip')
```

Selenium基本使用-补充

1. webelement.get_property：获取html的官方属性对应的值。
1. webelement.get_attribute：获取这个标签的某个属性（包含自定义的属性）的值。
2. driver.screentshot：获取当前页面的截图。这个方法只能在driver上使用。

更多请阅读相关源代码。

作业

1. 用多线程的方式爬取“百思不得姐”段子作业。
2. 网址：http://www.budejie.com/text/
3. 爬取下来后需要用csv保存下来。

EDU

CSDN学院　IT实战派