021 爬取北、上、广租房信息

* 核心技术

**在实现爬取北、上、广租房信息时，需要使用fake\_useragent、asyncio、aiohttp、requests、lxml以及pandas模块。各模块关键技术如下：**

1. **fake\_useragent**模块

**fake\_useragent模块是第三方模块，所以在使用前需要先通过以下命令进行模块的安装：**

**pip install fake-useragent**

**模块安装完成以后，首先需要导入fake\_useragent模块中的UserAgent类。代码如下：**

1. **from** fake\_useragent **import** UserAgent # 导入伪造头部信息的模块

**（1）调用fake\_useragent模块中UserAgent类的random属性可以实现随机获取浏览器的头部信息。语法格式如下：**

**UserAgent().random**

**实现随机获取浏览器头部信息的示例代码如下：**

1. **from** fake\_useragent **import** UserAgent # 导入伪造头部信息的模块
2. **for** i **in** range(5):
3. # 打印随机生成浏览器头部信息
4. print({"User-Agent": UserAgent().random})

运行结果如图1所示。



**图1 打印机随机生成的浏览器头部信息**

1. **asyncio模块**

**asyncio模块主要用于执行异步io的处理工作，异步和同步是相对的，在执行一些耗时的I/O操作时，只发出I/O指令，并不会等待I/O的结果，在此期间去执行其它事情，以此提高效率。**

**asyncio模块是python的内置模块，所以在使用前需要使用import导入该模块。代码如下：**

import **asyncio**

**（1）协程是编写异步应用的推荐方式，需要通过async/await语法进行声明。例如，实现并发运行多个协程可以参考以下示例代码：**

1. **import** asyncio # 导入异步IO模块
2. **import** time # 导入时间模块
3. **async def** asyncio\_demo(delay, what): # 创建异步函数
4. **await** asyncio.sleep(delay) # 等待指定时间
5. print(what)
6. **async def** main(): # 异步主函数
7. # 创建协程任务1
8. task1 = asyncio.create\_task(
9. asyncio\_demo(1, 'hello'))
10. # 创建协程任务2
11. task2 = asyncio.create\_task(
12. asyncio\_demo(2, 'world'))
13. print("start\_time",time.strftime('%X'))
14. **await** task1 # 调用协程任务1
15. **await** task2 # 调用协程任务2
16. print("end\_time",time.strftime('%X'))
17. asyncio.run(main()) # 并发运行多个协程任务

运行结果如下：

**start\_time 11:16:52**

**hello**

**world**

**end\_time 11:16:54**

**（2）在asyncio模块中提供了get\_event\_loop()方法，用于获取事件循环的实例对象，然后通过调用实例对象中的run\_until\_complete()方法来运行任务并等待任务完成。示例代码如下：**

1. **import** asyncio # 导入异步IO模块
2. **async def** hello1(a): # 创建异步函数
3. **await** asyncio.sleep(a) # 等待指定时间
4. print('执行任务！')
5. loop = asyncio.get\_event\_loop() # 创建事件循环实例
6. start\_time = loop.time() # 开始时间
7. print('开始时间：',start\_time) # 打印开始时间
8. loop.run\_until\_complete(hello1(1)) # 运行任务函数
9. end\_time = loop.time() # 结束时间
10. print('结束时间：',end\_time) # 打印结束时间
11. print('间隔时间：',end\_time - start\_time) # 打印间隔时间

运行结果如下：

**开始时间： 621910.312**

**执行任务！**

**结束时间： 621911.312**

**间隔时间： 1.0**

1. **aiohttp模块**

**aiohttp模块是异步网络请求模块，该模块需要和asyncio模块配合使用。aiohttp模块是python的内置模块，所以在使用前需要使用import导入该模块。代码如下：**

import **aiohttp**

**在实现异步网络请求时，首先需要定义一个异步请求的函数，在该函数中创建ClientSession()的实例对象，然后通过该实例对象调用get()方法即可实现异步网络请求。示例代码如下：**

1. **import** asyncio # 异步io模块
2. **import** aiohttp # 异步网络请求模块
3. **async def** request(url): # 异步网络请求的方法
4. **async with** aiohttp.ClientSession() **as** session: # 创建异步网络请求对象
5. **try**:
6. # 根据传递的地址发送网络请求
7. **async with** session.get(url,timeout=3) **as** response:
8. **if** response.status == 200: # 如果请求码为200说明请求成功
9. print(response.status) # 打印请求码
10. **except** Exception **as** e:
11. print(e.args) # 打印异常信息
12. **if** \_\_name\_\_ == '\_\_main\_\_':
13. loop = asyncio.get\_event\_loop()
14. loop.run\_until\_complete(request('https://www.baidu.com'))

运行结果如下：

**200**

1. **requests模块**

**requests模块用于实现发送网络请求，requests模块是第三方模块，所以在使用前需要先通过以下命令进行模块的安装：**

**pip install requests**

**模块安装完成以后，首先需要导入requests模块，然后调用对应的get()函数即可实现网络请求的发送功能。示例代码如下：**

1. **import** requests # 导入模块
2. response = requests.get('http://www.baidu.com')
3. print(response.status\_code) # 打印状态码
4. **lxml模块**

**lxml模块是一个解析库，用于解析HTML和XML，支持XPath解析方式，并且解析效率比较高。lxml模块是第三方模块，所以在使用前需要先通过以下命令进行模块的安装：**

**pip3 install lxml**

**在实现解析HTML代码时需要导入lxml模块下的etree子模块，然后创建etree子模块中的HTML实例对象，此时即可调用xpath()方法获取html代码中的指定内容。示例代码如下：**

1. **from** lxml **import** etree # 导入lxml解析html的模块
2. # 模拟html
3. html\_text = '''
4. <html>
5. <head>
6. <title>我的第一个 HTML 页面</title>
7. </head>
8. <body>
9. <p>文件内容</p>
10. <p>详细介绍</p>
11. </body>
12. </html>
13. '''
14. html = etree.HTML(html\_text) # 解析html代码
15. title\_text = html.xpath('//title/text()') # 获取标题内容
16. print(title\_text[0]) # 打印提取的标题内容

运行结果如下：

**我的第一个 HTML 页面**

1. **pandas模块**

**pandas模块用于实现数据的统计与分析工作，该模块为第三方模块所以在使用前需要先通过以下命令进行模块的安装：**

**pip3 install pandas**

**在pandas模块中DataFrame对象式比较常用的，在创建DataFrame对象时，需要通过字典来实现。其中每列的名称为键，而每个键对应的是一个数组，这个数组作为值。示例代码如下：**

1. **import** pandas # 导入数据统计模块
2. data = {'A': [1, 2, 3, 4, 5],
3. 'B': [6, 7, 8, 9, 10],
4. 'C':[11,12,13,14,15]}
5. data\_\_frame = pandas.DataFrame(data) # 创建DataFrame对象
6. print(data\_\_frame) # 打印DataFrame对象内容

运行结果如下：

**A B C**

**0 1 6 11**

**1 2 7 12**

**2 3 8 13**

**3 4 9 14**

**4 5 10 15**