# 第四阶段 Python数据分析

# Python 爬虫5

## 网校相关视频

Python爬虫开发

第一章网络爬虫基础

## 明确本次课知识点，明确重点难点

### 【知识点目标】

* 利用requests爬取数据
* 利用Beautiful Soup解析数据

### 【重点】

* 利用requests爬取数据
* 利用Beautiful Soup解析数据

### 【难点】

* 利用requests爬取数据
* 利用Beautiful Soup解析数据

## 复习巩固作业讲解

## 本次课程任务讲解

### 【知识点1】使用requests爬取数据

1. requests库介绍
   1. 概念及使用原因

requests是Python的一个HTTP客户端库，跟urllib类似，python的标准库urllib提供了大部分需要的HTTP功能，但是一个简单的功能就需要一大堆代码。而requests则可以很简洁地实现相应的功能。

1. requests使用

*import requests*

(1)发出请求

*#requests.get(url, params=None, \*\*kwargs)*

*response = requests.get(‘http://www.baidu.com’)*

#url：请求网址

#params：字典型请求参数

#kwargs：其他参数，如headers={'User-Agent': 'Mozilla/4.0','content-type': 'text/html; charset=utf-8'}

(2)返回响应结果（response是返回的响应对象)

(2.1)response.url获取请求地址

(2.2)response.text或response.content 获取响应内容。本方法requests会自动将内容转码。大多数unicode字体都会无缝转码。（但中文转码往往会出错）

(2.3)response.encoding来获取或设置网页编码

*print(response.text)*

*print(response.content)*

(3)设置传递参数

#headers设置request-header

#proxies设置代理服务器

*response=requests.get("http://www.baidu.com/", headers={'User-Agent': 'Mozilla/4.0','content-type': 'text/html; charset=utf-8'})*

*#带request headers的requests*

*proxies1 = {"https": "27.40.149.116:61234"}*

*response= requests.get("http://www.baidu.com",headers={'User-agent': 'Mozilla/4.0','content-type': 'text/html; charset=utf-8'},proxies=proxies1)*

*#带proxy和request headers的requests*

### 【知识点2】使用Beautiful Soup解析数据

1. Beautiful Soup库介绍

BeautifulSoup提供一些简单的、python式的函数用来处理导航、搜索、修改分析树等功能。它是一个工具箱，通过解析文档为用户提供需要抓取的数据，因为简单，所以不需要多少代码就可以写出一个完整的应用程序。

BeautifulSoup自动将输入文档转换为Unicode编码，输出文档转换为utf-8编码。

BeautifulSoup已成为和lxml、html6lib一样出色的python解析器，为用户灵活地提供不同的解析策略或强劲的速度。

简单来说，Beautiful Soup就是一个解析HTML结构的对象库，可以帮助用户方便地从HTML文档中提取出需要的内容。

参考：<http://beautifulsoup.readthedocs.io/zh_CN/lat>

1. Beautiful Soup库使用
   1. 基本操作
      1. 导入bs4库

*from bs4 import BeautifulSoup as bs*

* + 1. 创建 beautifulsoup 对象

*soup = BeautifulSoup(html,parser)*

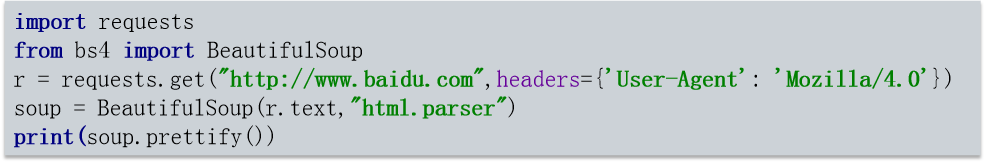
语法形式：soup = BeautifulSoup(input, parser)

input：一个网页的字符流

parser：解析器，可选**html.parser**、lxml、html5lib等，其中html.parser是python自带的解析器，速度快，容错能力强；lxml需要安装；html5lib对新的h5文档解析效果较好

* + 1. 打印 soup 对象的内容

*print (soup.prettify())*



1. Beautiful Soup对象结构

BeautifulSoup将复杂HTML文档转换成一个复杂的树形结构,每个节点都是Python对象,所有对象可以归纳为4种:

**Tag**

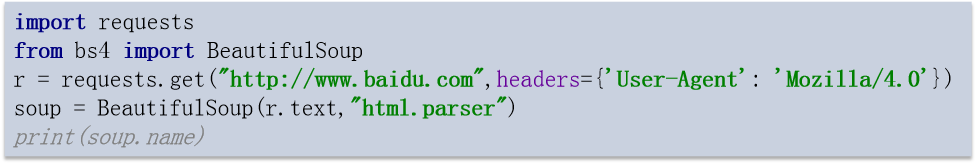
**NavigableString**

BeautifulSoup

Comment

* 1. BeautifulSoup:

BeautifulSoup对象表示的是**一个文档的全部内容。**大部分时候,可以把它当作Tag对象，是一个特殊的Tag，我们可以分别获取它的类型，名称，以及属性。通常，整个页面就是一个BS对象。（整个HTML文档，树状结构的根）



* 1. Tag:

Tag就是 HTML 中的每个标签。

语法形式：

***soup.标签名***——用来查找的是在所有内容中的第一个符合要求的标签

Tag 有两个重要属性，分别是 name 和 attrs。

soup 对象的 name 是输出的值便为标签本身的名称。

soup 对象的 attrs 是把标签的所有属性放在一个字典内

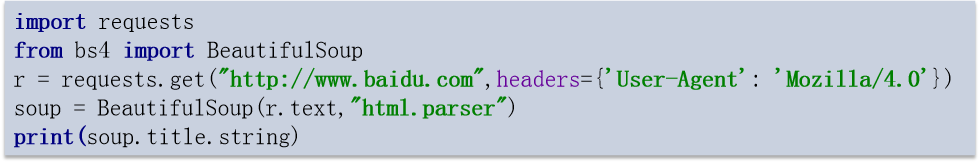


* 1. NavigableString:

NavigableString是可以遍历的字符串，通常用来获取标签内部的文字。

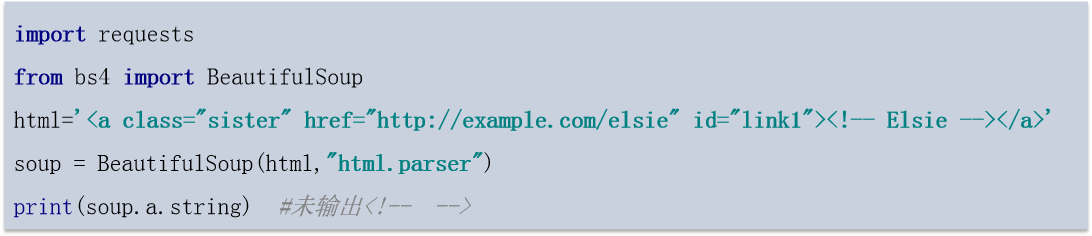
语法形式：

***soup.标签名.string 等价于 tag.string***



* 1. Comment:

Comment 对象是一个特殊类型的 NavigableString 对象，若原始内容是注释，则输出的内容不包括注释符号



1. BeautifulSoup的常用方法和属性
   1. find(name=None, attrs={}, recursive=True, text=None, \*\*kwargs)

搜索当前tag的第一个tag子节点,并判断是否符合过滤器的条件

* 1. find\_all(name=None, attrs={}, recursive=True, text=None, limit=None, \*\*kwargs)

搜索当前tag的所有tag子节点,并判断是否符合过滤器的条件

* 1. find及find\_all各项参数说明如下：
     1. name 参数

name 参数可以查找所有名字为 name 的tag,字符串对象会被自动忽略掉

* + 1. attrs 参数

注意：如果一个指定名字的参数不是搜索内置的参数名,搜索时会把该参数当作指定名字tag的属性来搜索,如果包含一个名字为 id 的参数,BeautifulSoup会搜索每个tag的”id”属性?

* + 1. text 参数

通过 text 参数可以搜索文档中的字符串内容.与 name 参数的可选值一样, text 参数接受 **字符串 , 正则表达式 , 列表, True**

* + 1. limit 参数

find\_all() 方法返回全部的搜索结构,如果文档树很大那么搜索会很慢.如果我们不需要全部结果,可以使用 limit 参数限制返回结果的数量.效果与SQL中的limit关键字类似,当搜索到的结果数量达到limit的限制时,就停止搜索返回结果.

* + 1. recursive 参数

调用tag的 find\_all() 方法时,BeautifulSoup会检索当前tag的所有子孙节点,如果只想搜索tag的直接子节点,可以使用参数 recursive=False .

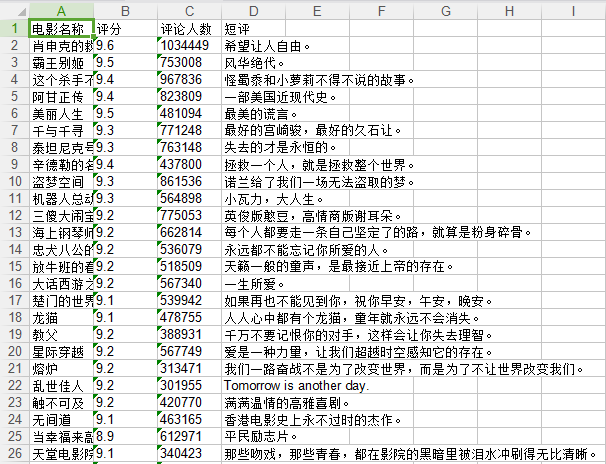
* 1. 直接子节点：.contents .children属性
  2. 所有子孙节点：.descendants属性
  3. 获取文本：.string属性
  4. 父节点：.parent .parents(迭代器)
  5. 兄弟节点：.next\_sibling .previous\_sibling 加s同上
  6. 下一个与上一个要解析的元素：.next\_elements .previous\_elements

### 【课程案例】使用requests和Beautiful Soup抓取数据

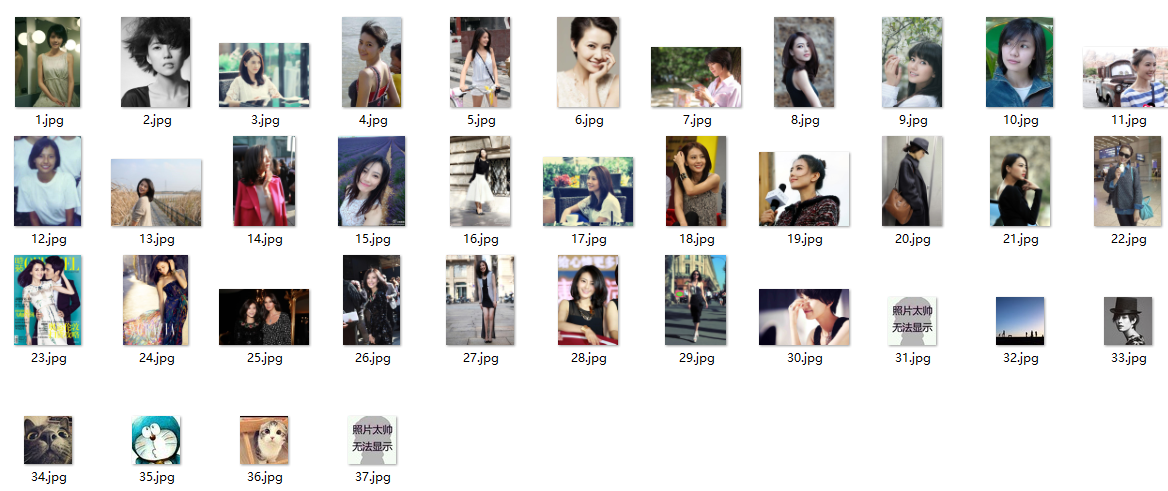
1. 爬取豆瓣电影top250的信息，并写入EXCEL

<https://movie.douban.com/top250>





1. 爬取豆瓣艺人信息中的高圆圆照片，并存入指定文件夹



## 自主学习作业讲解

无

## 课程总结

#### 使用requests爬取网页数据

* 使用BeautifulSoup处理爬取的网页数据

## 下次自主学习任务布置

作业：爬取51job上的数据分析师职位信息，将爬取到的内容写入EXCEL

