



数据解析

—— Xpath , BeautifulSoup4和正则的
使用

讲师：王老师

1. Xpath语法和lxml库
2. BeautifulSoup4库
3. 正则表达式和re模块

-  1. 了解Xpath
-  2. 熟悉Xpath语法
-  3. 熟练lxml库基本使用



XPath开发工具
Chrome插件XPath Helper。
Firefox插件Try XPath。

什么是XPath ?

xpath (XML Path Language) 是一门在XML和HTML文档中查找信息
的语言，可用在XML和HTML文档中对元素和属性进行遍历。

XPath节点

在 XPath 中，有七种类型的节点：元素、属性、文本、命名空间、处理指令、注释以及文档（根）节点。XML 文档是被作为节点树来对待的。树的根被称为文档节点或者根节点。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<bookstore>
```

```
<book>
```

```
<title lang="en">Harry Potter</title>
```

```
<author>J K. Rowling</author>
```

```
<year>2005</year>
```

```
<price>29.99</price>
```

```
</book>
```

```
</bookstore>
```



<bookstore> （文档节点）

<author>J K. Rowling</author>

（元素节点）

lang="en" （属性节点）

XPath语法：

XPath 使用路径表达式来选取 XML 文档中的节点或者节点集。这些路径表达式和我们在常规的电脑文件系统中看到的表达式非常相似。

| 表达式 | 描述 | 示例 | 结果 |
|----------|--------------------------------|----------------|----------------------|
| nodename | 选取此节点的所有子节点 | bookstore | 选取bookstore下所有的子节点 |
| / | 如果是在最前面，代表从根节点选取。否则选择某节点下的某个节点 | /bookstore | 选取根元素下所有的bookstore节点 |
| // | 从全局节点中选择节点，随便在哪个位置 | //book | 从全局节点中找到所有的book节点 |
| @ | 选取某个节点的属性 | //book[@price] | 选择所有拥有price属性的book节点 |

谓语：

谓语用来查找某个特定的节点或者包含某个指定的值的节点，被嵌在方括号中。
在下面的表格中，我们列出了带有谓语的一些路径表达式，以及表达式的结果：

| 路径表达式 | 结果 |
|------------------------------------|---|
| /bookstore/book[1] | 选取属于 bookstore 子元素的第一个 book 元素。 |
| /bookstore/book[last()] | 选取属于 bookstore 子元素的最后一个 book 元素。 |
| /bookstore/book[last()-1] | 选取属于 bookstore 子元素的倒数第二个 book 元素。 |
| /bookstore/book[position()<3] | 选取最前面的两个属于 bookstore 元素的子元素的 book 元素。 |
| //title[@lang] | 选取所有拥有名为 lang 的属性的 title 元素。 |
| //title[@lang='eng'] | 选取所有 title 元素，且这些元素拥有值为 eng 的 lang 属性。 |
| /bookstore/book[price>35.00] | 选取 bookstore 元素的所有 book 元素，且其中的 price 元素的值须大于 35.00。 |
| /bookstore/book[price>35.00]/title | 选取 bookstore 元素中的 book 元素的所有 title 元素，且其中的 price 元素的值须大于 35.00。 |

通配符：

*表示通配符。

| 通配符 | 描述 | 示例 | 结果 |
|-----|------------|--------------|---------------------|
| * | 匹配任意节点 | /bookstore/* | 选取bookstore下的所有子元素。 |
| @* | 匹配节点中的任何属性 | //book[@*] | 选取所有带有属性的book元素。 |

选取多个路径：

通过在路径表达式中使用 “|” 运算符，可以选取若干个路径。

示例如下：

```
//bookstore/book | //book/title
```

选取所有book元素以及book元素下所有的title元素

运算符：

| 运算符 | 描述 | 实例 | 返回值 |
|-----|---------|---------------|-------------------------|
| | 计算两个节点集 | //book //cd | 返回所有拥有 book 和 cd 元素的节点集 |
| + | 加法 | 6 + 4 | 10 |
| - | 减法 | 6 - 4 | 2 |
| * | 乘法 | 6 * 4 | 24 |
| div | 除法 | 8 div 4 | 2 |

运算符：

| | | | |
|----|-------|-------------|---|
| = | 等于 | price=9.80 | 如果 price 是 9.80，则返回 true。如果 price 是 9.90，则返回 false。 |
| != | 不等于 | price!=9.80 | 如果 price 是 9.90，则返回 true。如果 price 是 9.80，则返回 false。 |
| < | 小于 | price<9.80 | 如果 price 是 9.00，则返回 true。如果 price 是 9.90，则返回 false。 |
| <= | 小于或等于 | price<=9.80 | 如果 price 是 9.00，则返回 true。如果 price 是 9.90，则返回 false。 |
| > | 大于 | price>9.80 | 如果 price 是 9.90，则返回 true。如果 price 是 9.80，则返回 false。 |
| >= | 大于或等于 | price>=9.80 | 如果 price 是 9.90，则返回 true。如果 price 是 9.70，则返回 false。 |

运算符：

| | | | |
|-----|---------|---------------------------|---|
| or | 或 | price=9.80 or price=9.70 | 如果 price 是 9.80，则返回 true。如果 price 是 9.50，则返回 false。 |
| and | 与 | price>9.00 and price<9.90 | 如果 price 是 9.80，则返回 true。如果 price 是 8.50，则返回 false。 |
| mod | 计算除法的余数 | 5 mod 2 | 1 |

lxml库

lxml 是一个HTML/XML的解析器，主要的功能是如何解析和提取 HTML/XML 数据。

lxml和正则一样，也是用 C 实现的，是一款高性能的 Python HTML/XML 解析器，我们可以利用之前学习的XPath语法，来快速的定位特定元素以及节点信息。

lxml python 官方文档：<http://lxml.de/index.html>

需要安装C语言库，可使用 pip 安装：`pip install lxml`

基本使用：

我们可以利用他来解析HTML代码，并且在解析HTML代码的时候，如果HTML代码不规范，他会自动的进行补全。

```
from lxml import etree
```

```
html = etree.HTML(text)
```

```
result = etree.tostring(html)
```

```
print(result)
```

从文件中读取html代码：

除了直接使用字符串进行解析，lxml还支持从文件中读取内容。

```
from lxml import etree
```

```
# 读取外部文件 hello.html
```

```
html = etree.parse('hello.html')
```

```
result = etree.tostring(html, pretty_print=True)
```

```
print(result)
```

在lxml中使用XPath语法：

1. 获取所有li标签：
2. 获取所有li元素下的所有class属性的值：
3. 获取li标签下href为www.baidu.com的a标签：
4. 获取li标签下所有span标签：
5. 获取li标签下的a标签里的所有class：
6. 获取最后一个li的a的href属性对应的值：
7. 获取倒数第二个li元素的内容：
8. 获取倒数第二个li元素的内容的第二种方式：

演示内容介绍

爬取瓜子二手车网站

注意事项

1. headers
2. 编码








- 必做内容

爬取BOOS直聘职位信息

- 选做内容

爬出前10页

-  1. 了解BeautifulSoup4
-  2. 熟悉BeautifulSoup4语法
-  3. 熟练BeautifulSoup4库使用

BeautifulSoup4库

和 lxml 一样，Beautiful Soup 也是一个HTML/XML的解析器，主要的功能也是如何解析和提取 HTML/XML 数据。

lxml 只会局部遍历，而Beautiful Soup 是基于HTML DOM (Document Object Model) 的，会载入整个文档，解析整个DOM树，因此时间和内存开销都会大很多，所以性能要低于lxml。

Beautiful Soup 3 目前已经停止开发，推荐现在的项目使用Beautiful Soup 4。

安装和文档：

安装：

```
pip install bs4
```

中文文档：

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/index.zh.html>

几大解析工具对比：

| 解析工具 | 解析速度 | 使用难度 |
|---------------|------|------|
| BeautifulSoup | 最慢 | 最简单 |
| lxml | 快 | 简单 |
| 正则 | 最快 | 最难 |

简单使用：

```
from bs4 import BeautifulSoup

# 创建 Beautiful Soup 对象
# 使用lxml来进行解析
soup = BeautifulSoup(html,"lxml")

print(soup.prettify())
```

四个常用的对象：

Beautiful Soup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:

1. Tag
2. NavigableString
3. BeautifulSoup
4. Comment

1. Tag :

Tag 通俗点讲就是 HTML 中的一个标签。我们可以利用 soup 加标签名轻松地获取这些标签的内容，这些对象的类型是bs4.element.Tag。但是注意，它查找的是在所有内容中的第一个符合要求的标签。

2. NavigableString :

如果拿到标签后，还想获取标签中的内容。那么可以通过tag.string获取标签中的文字。

3. 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,它支持 遍历文档树 和 搜索文档树 中描述的大部分的方法.

4. Comment :

Tag , NavigableString , BeautifulSoup 几乎覆盖了html和xml中的所有内容,但是还有一些特殊对象.容易让人担心的内容是文档的注释部分

Comment 对象是一个特殊类型的 NavigableString 对象

遍历文档树：

1. contents和children：

contents:返回所有子节点的列表

children:返回所有子节点的迭代器

2. strings 和 stripped_strings

strings:如果tag中包含多个字符串,可以使用.strings来循环获取

stripped_strings:输出的字符串中可能包含了很多空格或空行,使用.stripped_strings可以去除多余空白内容

搜索文档树：

1. find和find_all方法：

搜索文档树，一般用得比较多的就是两个方法，一个是find，一个是find_all。find方法是找到第一个满足条件的标签后就立即返回，只返回一个元素。find_all方法是把所有满足条件的标签都选到，然后返回回去。

搜索文档树：

有时候使用css选择器的方式可以更加的方便。使用css选择器的语法，应该使用select方法。以下列出几种常用的css选择器方法：

2. select方法：

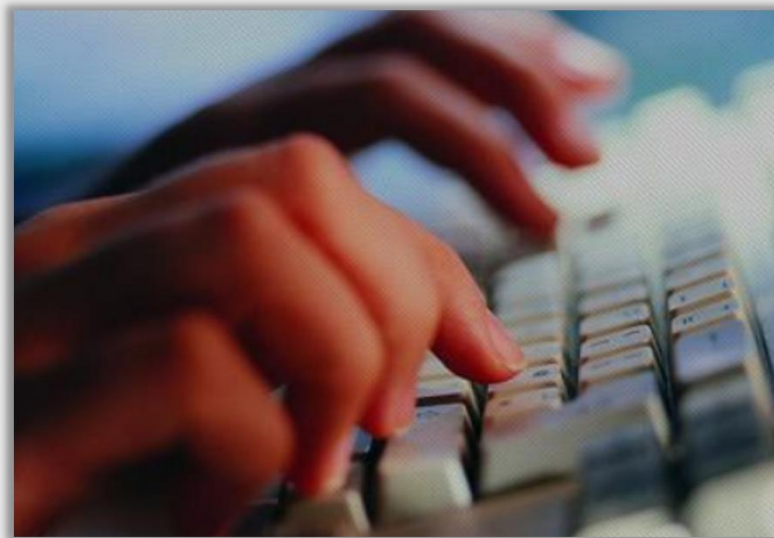
- (1) 通过标签名查找：`print(soup.select('a'))`
- (2) 通过类名查找：`print(soup.select('.sister'))`
- (3) 通过id查找：`print(soup.select("#link1"))`
- (4) 组合查找：`print(soup.select("p #link1"))`
- (5) 通过属性查找：`print(soup.select('a[href="http://example.com/elsie"]'))`
- (6) 获取内容：`print (soup.select('title')[0].get_text())`

演示内容介绍

爬取豆瓣Top250

注意事项

1. headers
2. 编码
3. 使用BeautifulSoup







- 必做内容

爬取快代理ip

- 网址

<https://www.kuaidaili.com/free/inha/1/>

-  1. 熟悉掌握正则表达式语法。
-  2. 熟练使用re模块中的函数。

什么是正则表达式：

通俗理解：按照一定的规则，从某个字符串中匹配出想要的`数据`。这个规则就是正则表达式。

标准答案：<https://baike.baidu.com/item/正则表达式/1700215?fr=aladdin>

正则表达式语法：

- 单字符串匹配规则
- 匹配多个字符串
- 开始结束和或语法
- 转义字符和原生字符串

re模块中常用的函数：

- match
- search
- group分组
- findall
- sub
- split
- compile

| 字符 | 匹配 |
|---------|-------------------------|
| . | 匹配任意字符（除了\n） |
| [] | 匹配中括号中的某一项 |
| \d / \D | 匹配数字/非数字 |
| \s / \S | 匹配空白/非空白字符 |
| \w / \W | 匹配a-z和A-Z以及数字和下划线/与\w相反 |

| 字符 | 匹配 |
|--------------|---------------------|
| * | 匹配前一个字符0次或者无限次 |
| + | 匹配前一个字符1次或者无限次 |
| ? | 匹配前一个字符0次或者1次 |
| {m} / {m, n} | 匹配前一个字符m次或者n次 |
| *? / +? / ?? | 匹配模式变为非贪婪（尽可能少匹配字符） |

| 字符 | 匹配 |
|---------|-------------------|
| ^ | 匹配字符串开头 |
| \$ | 匹配字符串结尾 |
| \A / \Z | 指定的字符串匹必须出现在开头/结尾 |

re模块中常用函数：

`re.match`

尝试从字符串的起始位置匹配一个模式，如果不是起始位置匹配成功的话，`match()`就返回none。

`re.search`

扫描整个字符串并返回第一个成功的匹配。

re模块中常用函数：

分组：

在正则表达式中，可以对过滤到的字符串进行分组。分组使用圆括号的方式。

group：和group(0)是等价的，返回的是整个满足条件的字符串。

groups：返回的是里面的子组。索引从1开始。

group(1)：返回的是第一个子组，可以传入多个。

re模块中常用函数：

`re.compile`

用于编译正则表达式，生成一个正则表达式（ Pattern ）对象

`re.findall`

在字符串中找到正则表达式所匹配的所有子串，并返回一个列表，如果没有找到匹配的，则返回空列表。

re模块中常用函数：

`re.sub`

用来替换字符串。将匹配到的字符串替换为其他字符串。

`re.split`

使用正则表达式来分割字符串。

演示内容介绍

爬取BOOS直聘职位信息





- 必做内容

爬取糗事百科文字区内容

- 选做内容

爬出前10页

EDU

CSDN学院 IT实战派

