

电工导 C 第一次实验报告

姓名: 宋士祥

学号:521030910013

班级:F2103001

2022 年 9 月 22 日

1 实验概览

本次实验旨在通过对于 HTML 网页的编辑与爬取, 掌握 HTML 网页的基本架构, 同时对于BeautifulSoup4 有了基本的认识, 利用 python 对 html 文件进行解析。

在本实验中, 我们首先了解了 HTML 文件架构和常见的标签, 同时自主修改basic.html 的内容以了解相关的内容架构, 查看对应效果。在此基础上, 本次实验学习了BeautifulSoup 包及其相关的函数, 包括给定标签名查找、给定多个标签名查找、给定标签名属性名值对查找等。最后, 对正则表达式进行了简要的介绍。利用正则表达式, 我们可以很方便地爬取网页并找到对应的内容, 过滤掉一些无效的信息。

本次实验共需完成三个任务:

- 抓取网页返回所有的超链接 URL
- 抓取网页返回所有的图片地址
- 抓取知乎日报 URL 并抓取网页的标题与图片

2 实验环境

本次实验采用所需的实验环境如下:

- Docker 中的sjtunic/ee208 镜像
- Python3 (使用 VSCode 编译)
- BeautifulSoup4 扩展以及lxml 扩展。

3 练习 1

3.1 问题重述与代码说明

练习 1 要求我们在 www.baidu.com 中爬取所有的 `` 形式的超链接, 并将结果打印在res1.txt 中。代码的基本架构助教已经给出, 只需完成praseURL(content) 这一函数即可。

考虑题设中要求, 我们应当首先创建一个空的集合, 然后在网页中找到所有的a 标签, 最后找到所有a 标签中的href 部分即可, 考虑到存在部分链接重叠的可能, 这里应当采用set 来完成这一任务。

这里给出代码的补充部分:

```

1 def parseURL(content):
2     urlset = set()
3     # 此处应指定features 避免warning
4     soup = BeautifulSoup(content, features="lxml")
5     # 超链接格式一般为<a href=...>
6     for k in soup.find_all('a'):
7         urlset.add(k.get('href', ''))
8     return urlset

```

3.2 运行结果

得到的res1.txt 如下图所示（部分）：

```

1 //help.baidu.com
2 https://jiankang.baidu.com/widescreen/home
3 https://www.baidu.com/s?wd=%E5%9D%A%E5%A%E8%8%E4%B8%8%E6%9D%A1%E6%8E%A7%E5%88%B6%E7%B8%A8F
4 +%E6%9C%8D%E5%A1%E9%A8%98%E8%B4%A8%E9%87%8F%E5%8F%91%E5%B1%95%sa=fyb_n_homepage&rsv_dl=fyb_n_homepage&from=super&
5 cl=3&tn=baidutop10&fr=top1000&rsv_idx=2&hisfilter=1
6 javascript;;
7 http://ir.baidu.com
8 http://tieba.baidu.com/f?fr=wwwt
9 https://pan.baidu.com?from=1026962h
10 https://www.baidu.com/s?wd=%E6%B6%B9%E7%96%AB%E5%A4%A7%E5%B7%B4%E4%B8%A7%E7%B8%B8%E8%87%B427%E6%AD%B8
11 +%E8%B4%85%E5%B7%9E%E7%9C%81%E9%95%B3%A%E5%BD%B8%E6%9F%A5%sa=fyb_n_homepage&rsv_dl=fyb_n_homepage&from=super&cl=3&
12 tn=baidutop10&fr=top1000&rsv_idx=2&hisfilter=1
13 https://wenku.baidu.com
14 https://www.baidu.com/s?wd=%E5%88%B8%E6%96%B0%E5%B8%95%E9%A2%B6+%E7%A7%91%E6%8A%80%E8%B5%88%E8%B3%BD%sa=fyb_n_homepage&
15 rsv_dl=fyb_n_homepage&from=super&cl=3&tn=baidutop10&fr=top1000&rsv_idx=2&hisfilter=1
16 http://image.baidu.com/i?tn=baiduimage&ps=i&ct=201326592&lm=-1&cl=2&nc=1&ie=utf-8
17 https://map.baidu.com/?newmap=1&ie=utf-8&s=s
18 https://zhidao.baidu.com
19 http://www.baidu.com/more/
20 http://e.baidu.com/ebaidu/home?refer=887
21 http://image.baidu.com/
22 https://www.baidu.com/s?wd=%E6%97%A5%E6%9C%ACGDP%E6%81%90%E5%B0%86%E8%B7%8C%E5%9B%9E%E8%B7%B3%E5%B9%B4%E5%89%B0&
23 sa=fyb_n_homepage&rsv_dl=fyb_n_homepage&from=super&cl=3&tn=baidutop10&fr=top1000&rsv_idx=2&hisfilter=1
24 https://b2b.baidu.com/s?fr=wwwt
25 http://v.baidu.com/v?ct=301989888&rn=20&pn=0&db=0&s=25&ie=utf-8
26 https://haokan.baidu.com/?sfrom=baidu-top
27 http://fanyi.baidu.com/
28 //home.baidu.com
29 //www.baidu.com/duty
30 /
31 http://music.taihe.com
32 https://www.baidu.com/

```

3.3 问题讨论

3.3.1 为何不使用 ["href"] 的形式

我们发现部分的链接没有href 模块（也即空连接）。虽然我们可以使用 try 的形式规避这一情况的出现，但是考虑到代码的简洁性，我们直接采用get 的形式更为方便。

3.3.2 features="lxml" 的作用

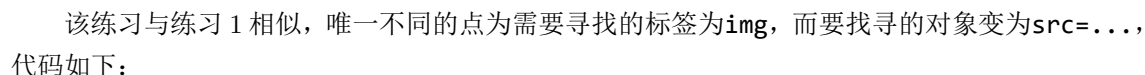
避免出现未指定 HTML 的 praser 而出现 Warning 的情况出现。

3.3.3 网页的顺序

由于代码重新运行了一次，导致报告的结果和提交的结果并不完全已知。其原因为 Python 中 set 的数据结构使得每次打印均以乱序形式打印。

4 练习 2

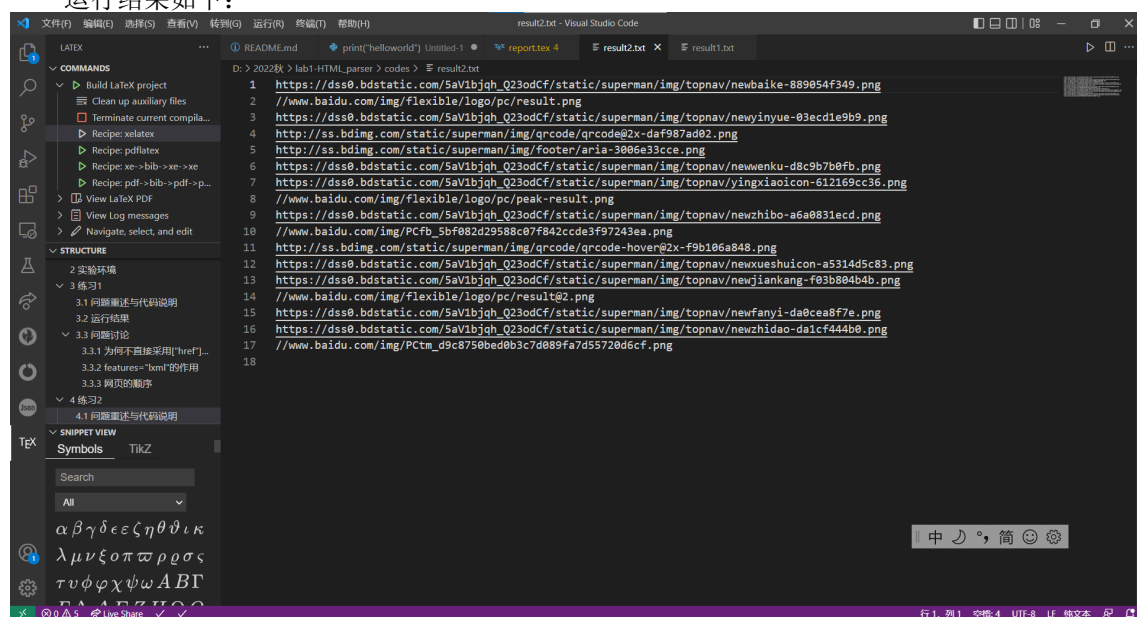
4.1 问题重述与代码说明

该练习与练习 1 相似，唯一不同的点为需要寻找的标签为，而要找寻的对象变为src=...，代码如下：

```
1 def parseURL(content):
2     urlset = set()
3     # 此处应指定features 避免warning
4     soup = BeautifulSoup(content, features="lxml")
5     # 图片格式一般为 <img src=...>
6     for k in soup.find_all('img'):
7         urlset.add(k.get('src', ''))
8     return urlset
```

4.2 运行结果

运行结果如下：



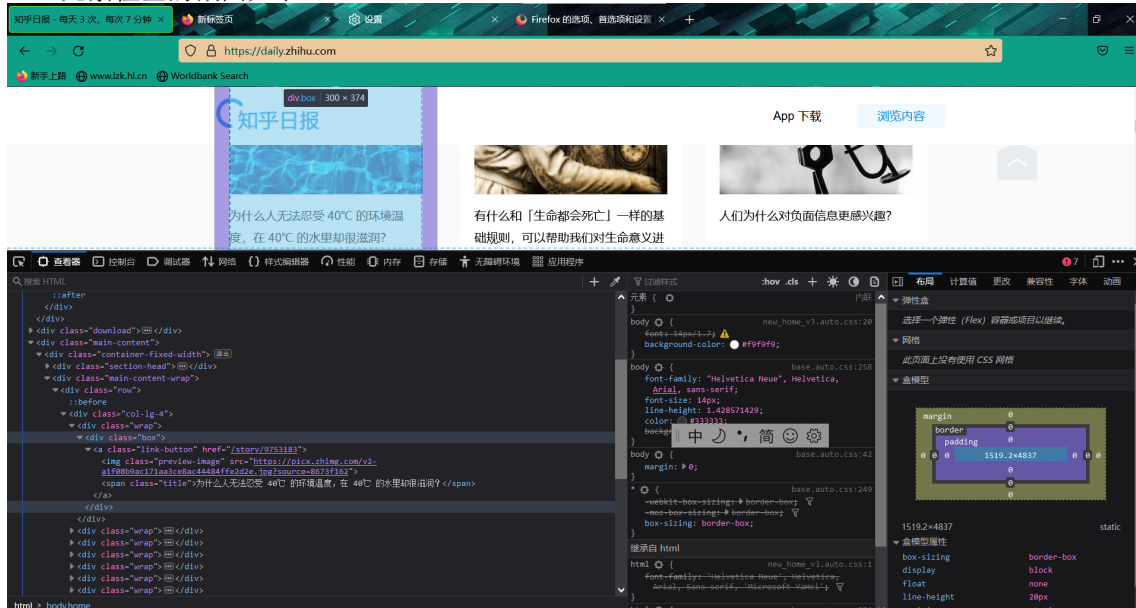
5 练习 3

5.1 问题分析与代码说明

该问题要求给定知乎日报的 url，返回网页中的图片和相应文本，以及每个图片对应的超链接网址。并将图片地址，相应文本，超链接网址以特定格式打印到res3.txt 中。

经过 Firefox 105.0 的元素审查，我们发现知乎日报的链接格式为<div class="box">外行对你熟知的领域有哪些误解？</div> 的形式。于是我们只需要直接找到每一个 box，并提取出 box 中对应的格式即可。

元素检查的截图如下：

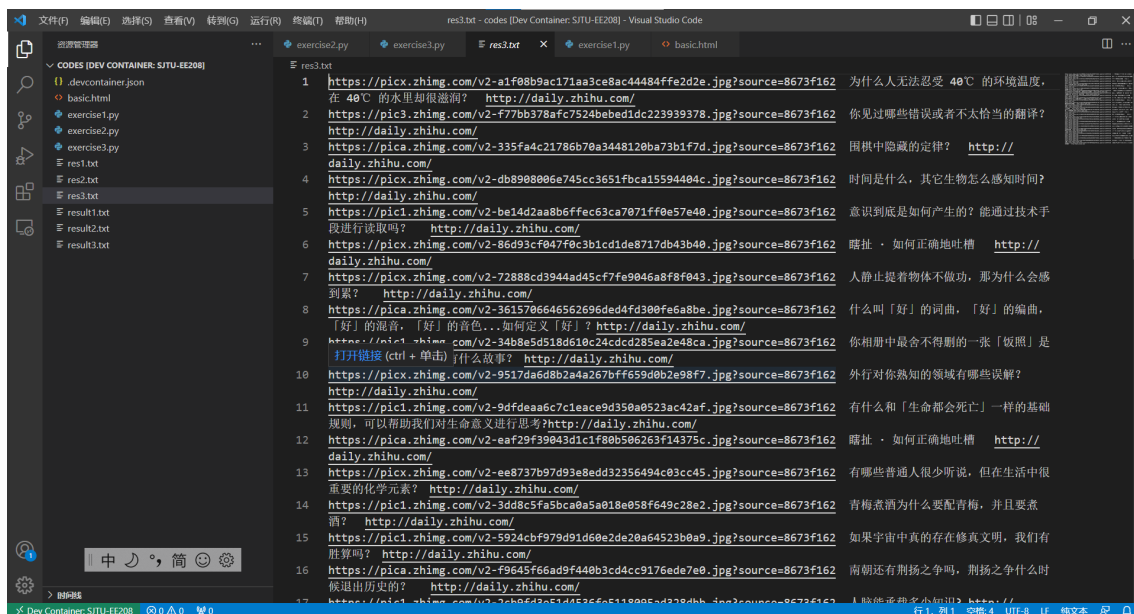


因此本题的核心代码如下：

```
1 def parseZhihuDaily(content, url):
2     # 创建目标列表
3     zhihulist = list()
4     # 此处应指定避免featureswarning
5     soup = BeautifulSoup(content, features="lxml")
6     # 找到每一个链接（即boxes）通用格式为<div class = "box"><a href="..."><img><span>...</
7     for boxes in soup.find_all("div", {"class": "box"}):
8         # 直接调用内部的boxesimg span 和链接
9         image = boxes.find("img").get("src", "")
10        title = boxes.find("span").contents[0]
11        link = boxes.find("a").contents[0].get("href", "")
12        # 将链接和daily.zhihu.com 连起来
13        fulllink = urllib.parse.urljoin("http://daily.zhihu.com/", link)
14        zhihu = [image, title, fulllink]
15        zhihulist.append(zhihu)
16    return zhihulist
```

5.2 运行结果

运行结果如下：



5.3 添加报头

将爬虫掩盖为正常用户访问，我们采取添加报头的形式，具体函数如下（为了化简代码直接采用了本机 Firefox 的报头信息）。

```
1 def main():
2     url = "http://daily.zhihu.com/"
3     req = urllib.request.Request(url)
4     req.add_header('User-Agent', "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:105.0)
5                                     Gecko/20100101 Firefox/105.0")# 本
6                                     机Firefox 的报头
7
8     content = urllib.request.urlopen(req).read()
9     zhihus = parseZhihuDaily(content, url)
10    write_outputs(zhihus, "res3.txt")
```

5.4 问题讨论

添加报头的意义

很多时候，站点为了信息安全，都有对应的反爬虫机制，拒绝爬虫请求。采用add_header 的方式可以有效地规避这一问题，让服务器认为是正常的用户，从而爬取到更多信息。

6 拓展思考

href 的形式根据我们的爬取结果，有以下几种形式：

- 未加密的超链接http://形式
- 加密的超链接https://形式
- 调用 JavaScript 的javascript;; 格式
- 与当前页面采用同一种协议的//形式
- /.../链接为原网站的相对路径