实验 3 爬取网络图片数据

实验时长：4 小时

实验难度：一般

实验摘要：在中国，每年会有不少于 4000 万只狗在流浪，通过网络交流平台（如百度贴吧）上发布关于流浪狗领养的信息，爱狗人士把救助站，或是失去家园的狗的图片信息发布出去，来供有余力的人家领养，帮助这些流浪狗获得爱与关怀。但是在交流平台上交换信息内容大部分与主题无关，本次案例就是实现通过数据爬取，快速取得百度贴吧内流浪狗领养图片，以便筛选。

实验建议：了解数据抓取相关知识。

实验目标：能够爬取网络图片数据。

1、爬取网络图片数据

- 1.1、网页分析
- 本任务是爬取百度贴吧流浪狗领养贴中的图片，所爬取网址为 https://tieba.baidu.com/p/6045474546，在进行抓取之前首先获取图片地址信息标签位置。
- 进入网址，右击鼠标，在弹出的快捷菜单中选择“查看页面源代码”选项（或通过快捷键 F12），即可看到网页源代码（以火狐浏览器为例），在弹出页面后单击其左上角的小箭头，选中页面中的元素（或按 Ctrl+Shift+C）即单击所查看的图片。
-

- 即可跳转到图片信息所在的代码行。
-

- 可以看到图片标签<img>，文档内标识属性<class>，图片 URL 属性<src>，图片大小属性<size>、<width>和<height>，截取部分代码如下所示：
- <img class="BDE_Image" src="https://imgsa.baidu.com/forum/w%3D580/sign=bd7e89d7e91190ef01fb92d7fe1b9df7/d3dcc3cec3fdfc03aa0eba82da3f8794a4c226b1.jpg" size="40923" width="520" height="293">
- ---
- 1.2、读取网页内容
- ---
-

接下来开始编写代码，首先创建一个文件名为 fetch_image.py 的.py 文件，代码中首先导入 urllib.request，bs4。参考代码如下，代码前数字含义表示执行顺序和标记：

- `import urllib.request`
- `from bs4 import BeautifulSoup`
-
-

定义目标图像 URL 地址。代码如下：

-

```
domain = 'https://tieba.baidu.com/p/6045474546'
```

-
-

实际情况当中某些网站会采取反爬机制，采取反爬机制之后，百度等搜索引擎无法对网站的内容进行网页爬取，解决方法是修改 User Agent 来模拟浏览器访问。代码如下：

- `# 构造一个请求`
- `req = urllib.request.Request(domain)`
- `# 用虚拟客户端来模拟的浏览器`
- `req.add_header('User-Agent','fake-client')`
-
-

通过 urllib.request 的 urlopen 来打开网站发起请求相应内容，以获取所需数据，并通过 read()方法来读取内容，代码如下所示：

- `html=urllib.request.urlopen(req)`
- `info = html.read()`
-
- 接下来打印输出结果，检查是否成功执行，代码如下：
-

```
print('打印 info','\n',info)
```

-
- 输出结果见图，表示成功爬取。
-

```
打印info
 b'<!DOCTYPE html><!--STATUS OK--><html><head><meta name="keywords" c
288399
```

- 接下来需要解析 info，以及下载图片并重命名，这里自定义一个 get_images 函数。函数功能是取得图片 URL 并下载到本地计算机，同时打印输出"全部抓取完成"提示信息。函数调用如下：
- `get_images(info) # 通过 get_images 取得图片`
- `print(' 全部抓取完成')`

-
-

get_images 函数完整定义见 1.3。

-

-

下一步

-
-

1.3、获取图片数据

接下来编写 get_images 函数，首先创建一个 BeautifulSoup 的对象，获取的数据除了图片还有很多无用的数据，接下做筛选。

beautifulsoup4 库中主要的类是 BeautifulSoup，它的实例化对象相当于一个页面，得到的是一个树形结构，它包含 HTML 页面的每一个标签（Tag），比如<head>,<body>等，可以理解这时候 HTML 中的结构都变成了 BeautifulSoup 的一个属性，可以直接通过 Tag 属性访问。这样就可以通过 Tag 属性获取到图片的路径。

函数内定义 soup 这个 BeautifulSoup 对象，查看是否输出成功。如下：

```
def get_images(info):
    soup = BeautifulSoup(info,'html.parser') # 创建 beautifulsoup 对象 soup
    print("打印 soup",'\n',soup)
```

BeautifulSoup 函数内第二个参数'html.parser'，指明采用 html 解释器。

百度贴吧页面内图片标识为'BDE_Image'，通过 find_all 函数进行筛选，并打印查看是否只有图片数据。参考代码如下：

```
# 找到所有图片标签
    all_img = soup.find_all('img', class_ = 'BDE_Image')
    print("打印 all_img",all_img)
```

可以看到变量 all_img 已经存储了筛选出的图片数据，包含图片基本信息如：height，size，src，width 等。如图所示：

使用 for 循环遍历 all_img 内容把每个图像进行重命名，通过 urllib.request.urlretrieve 下载图片保存到本地，该函数有一个必填参数即网页 <img>标签 src 属性以及可选参数即下载之后的图片存放路径。其中图片存放路径可以只写一个文件名（image_name），这样会默认保存到工作目录，也可以指定路径。

参考代码如下：

```
    i = 0
    for img in all_img:
        image_name = '%s.jpg'% i
        i = i+1
# 在/home/data 工作目录下新建 img 文件夹，下载图片保存到/home/data/img 下
        urllib.request.urlretrieve(img['src'],'./img/'+str(i)+'.jpg')
        print(' 成功抓取到图片 ',img['src'])
    print(' 抓取完成 !')
```

使用 requests 方法参考代码如下：

```
import requests
from bs4 import BeautifulSoup
url = 'https://tieba.baidu.com/p/6045474546'
#设置请求头
header = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:7
0.0) Gecko/20100101 Firefox/70.0"}
r = requests.get(url,headers = header)
info = r.text
```

```
soup = BeautifulSoup(info,'html.parser')
all_img = soup.find_all('img', class_ = 'BDE_Image')
for index,img in enumerate(all_img):
    src = img['src']
url = './img' + str(index+1) + ".jpg"
  r = requests.get(src)
  #下载图片
  with open(url, 'wb') as f:
        f.write(r.content)
        print("下载完%d 张了..."%(index+1))
```