

一、爬虫简介

1. 模块准备

requests

lxml

selenium

scrapy

命令：

```
python
pip install 模块名
```

2. 爬虫的分类

- 1. 全网爬虫
 - 百度、搜狗、Google、bing
- 2. 聚焦爬虫
 - 针对单一网站的搜索工具

3. 爬虫的注意点

- 1. 变化非常迅速
 - 今天代码没问题——>明天代码不行了
 - 时间问题
 - 身份问题
 - 网站变化
 - 网站更新、升级
 - 网站崩溃、倒闭
- 2. 爬虫爬取数据的来源
 - 网址
 - 网址的构成(数据来源)
 - 协议
 - 域名
 - 资源目录
 - 搜索关键字
- 3. 爬虫的流程
 - 分析网址
 - 构建爬虫
 - 数据清洗
 - 数据保存
- 4. 爬虫代码问题
 - 一个代码只适用于一个网站
- 5. 爬虫只能爬取可以获取的数据
- 6. 学习爬虫的过程中，不要过多的发起请求

4. 爬虫的作用

获取数据

- 文字
- 图片
- 视频
- 音频

为了批量的获取数据、可以进行自动化的操作、快速的获取

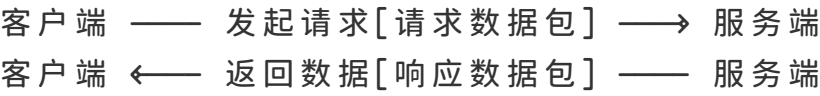
5.反爬虫

学习如何识别对应网站的反爬策略，从而找到可以通过反爬策略的方法

二、抓包分析

1.数据来源

数据包的形式进行数据的传输



2.什么是抓包？

在客户端与服务器交互的时候，对其相互传输的数据包进行解析从而获取需要的内容

❤️ 补充：每个网站都是独立的存在，因为开发人员的不同，不同的网站所实行的方法是不一样的

3.如何抓包(开发者选项、调试控制台)

需要开启抓包后，进行请求操作，才会有数据包被抓到

请求方法

1. GET
2. POST

4.如何找到需要的资源包

方法一：过滤

Doc中的资源包

方法二：搜索(在空白处按 **Ctrl + F**)

寻找数据，构建爬虫的步骤



注：搜索的范围只有：响应、响应头、请求头

5.如何构造请求

状态码

- 200-300：请求正常

- 300-400：重定向
- 400-500：服务器问题
- 大于或等于500：客户端问题

```
python
# 导入模块
import requests      # 发送请求的模块

url = 'https://www.xxsy.net/'
headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://www.baidu.com/link?url=su040z7K_kkQ_LRXty3QNX7Y-iCJFC0abR0R_V_z_T_&wd=&eqid=ca9a0fcb006058fc000000066953763c',
    'cookie': 'newstatisticUUID=1767077440_8196224563',
}

# 构建并发起get请求
res = requests.get(url=url, headers=headers)
# res.status_code    # 查看状态码
print(res.status_code)
# res.text    # 打印响应文本信息
print(res.text)
# res.content    # 响应数据的二进制形式
print(res.content)
```

说明

- res.status_code：查看状态码
- res.text：响应文本信息(字符串类型数据)
- res.content：响应数据的二进制形式(流媒体数据)

代码过期问题

- 身份问题：cookie是会过期

补充

不同的网站，以及同一网站不同资源路径，都可能存在不一样的校验方式

同一、相似的资源路径下通常校验规则一样

所以说一般的爬虫流程：分析每一个页面所需要构建的请求方式

作业

需求：

1. 熟悉浏览器抓包流程，并可以找到目标资源
2. 获取出【<https://www.xxsy.net/>】网站中，无限追书板块的10本书名，将书名打印出来(选做)，提示：使用正则

```
python
# 导入模块
import requests      # 发送请求的模块
# 导入正则模块
import re

url = 'https://www.xxsy.net/'
headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://www.baidu.com/link?url=su040z7K_kkQ_LRXty3QNX7Y-iCJFC0abR0R_V_z_T_&wd=&eqid=ca9a0fcb006058fc000000066953763c',
    'cookie': 'newstatisticUUID=1767077440_8196224563',
}
```

```
# 构建并发起get请求
res = requests.get(url=url, headers=headers)

# 定义正则查找条件
module_re = r'<div class="text-t34 text-l-c-1 font-semibold truncate break-all wrap-word hover:text-l-brand-primary-0 block" data-v-540fce0b>(.*?)</div>'

book_html = re.findall(module_re, res.text)

for i in range(len(book_html)):
    print(f'{i+1}.{book_html[i]}')
```

注：有些网站当中的属性值以及标签，是由于、框架进行设置的，存在不确定性，所以具体的正则表达式的写法，需根据代码当中的响应内容来进行

三、登录流程分析

1. 权证流

权限(VIP)、凭证(是否登录)

2. 登录流程分析

限流问题

单一时间段内，限制同一IP、用户的访问(请求)次数

测试网址

- 笔趣阁登录1
<https://www.bqgam.com/user/login.html>

如何快捷的找到登录目标资源包

- 通过搜索关键字：login、submit
- 通过浏览器手动过滤出一些常见的资源包
- 第三方抓包工具

代码

```
python
import requests

# 请求地址
url = 'https://www.bqgam.com/user/action.html'
# 请求头内容
headers = {
    'Host': 'www.bqgam.com',
    'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8',
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
}

# 负载
data = 'action=login&username=19896223228&password=wzw200508'

# 发起请求
res = requests.post(url=url, headers=headers, data=data)
print(res.text)
```

3.复杂登录流程分析

测试网站

- 天下书盟
- <http://www.fbook.net/>

复杂登录流程分析



图片验证码原理

就是每个验证码都会有一个独自的ID值，这个id值跟当前验证码的结果组成一个键值对，提交请求时，会自动携带图片验证码的ID，从而校验提交的验证码是否与键值对当中的验证码一致

- Cookie:Hm_lvt_60b7389344d5e30b600d3767cdf28d50=1767158046,1767158565,1767159422; Hm_lpvt_60b7389344d5e30b600d3767cdf28d50=1767159422; HMAccount=306E98BBEBC08A58
- Cookie:Hm_lvt_60b7389344d5e30b600d3767cdf28d50=1767158046,1767158565,1767159422; HMAccount=306E98BBEBC08A58; ASP.NET_SessionId=pvcyfaa0uywkwb2qh3gbv5hp; Hm_lpvt_60b7389344d5e30b600d3767cdf28d50=1767159429

实战

```
python
import requests

# http://www.fbook.net/Member/Captcha?t=0.10251672893148756
# 验证码图片地址
code_url = 'http://www.fbook.net/Member/Captcha'
code_res = requests.get(code_url)

# 将验证码图片保存到指定文件夹
with open('验证码图片/code_01.gif', 'wb') as f:
    f.write(code_res.content)

# 登录
# 登录网址
login_url = 'http://www.fbook.net/Member/Login/'
# 登录请求头
login_headers = {
    'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8',
    'Cookie': f'ASP.NET_SessionId={code_res.cookies.get("ASP.NET_SessionId")}; Hm_lvt_60b7389344d5e30b600d3767cdf28d50=1767158046,1767158565,1767159422,1767173244; HMAccount=306E98BBEBC08A58; max=AFA58E2DD36DBFADB7F80B766ADEA6A55691F5404576EF7CA6C692327A33F85DB04FDD8E2067594506B40884CA3D81C28EB30E02A47544B9A6DFEF889CFF8F371A5FA4AA1652CFF41A7FBD4E3FD4626647FB2815E376FF70D111A75692B1292A0F91CB6F6DB41F697FA1A67130D8327C6E4BAB22; Hm_lpvt_60b7389344d5e30b600d3767cdf28d50=1767173476',
}
# 登录负载
login_data = f'loginName=huihuia24&loginPass=wzw200508&captcha={input("请输入验证码：")}'

# 登录并发送请求
login_res = requests.post(url=login_url, headers=login_headers, data=login_data)
# 输出结果
print(login_res.text)
```

四、IP代理

1.代理网址

- 站大爷-免费代理：[点击此处](#)
- 中国-免费在线代理列表(需注册，麻烦)：[点击此处](#)

- 3. 开心代理：[点击此处](#)
- 4. 无忧代理IP：[点击此处](#)

2.代理的原理

什么情况会使用代理

- 1. 自己的主机被网站限流了
- 2. 不想暴露自己的ip的时候

♥♥ 获取一个代理服务器，转发我们的请求以及回传的响应

原理



3.代理的使用

Chrome浏览器使用代理

注：在使用代理前需要关闭所有正在运行的Google浏览器窗口，否则代理无效

打开cmd通过命令：

```
shell
chrome --proxy-server=代理ip:端口号
```

例：

```
shell
Chrome --proxy-server=210.16.160.222:7890
```

4.python使用代理

示例

```
python
import requests

# 使用代理访问https://www.xxsy.net/
url = 'https://www.xxsy.net/'

headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36",
}

# 配置代理
proxy = {
    'http': 'http://210.16.160.222:7890',
    'https': 'http://210.16.160.222:7890',
}

res = requests.get(url=url, headers=headers, proxies=proxy)
print(res.text)
```

IP池的使用：搭建代理池——根据代理IP时长来进行设计使用流程



5.搭建ip代理池

代理池的校验方式

1. 使用前校验
 - 适合在代理ip提供商提供的代理ip质量不是很好的情况下使用，此方法来提高你的代理IP的可靠性
2. 使用时校验
 - 使用在代理IP提供商提供的代理IP质量足够高的时候使用，此方法来节省时间
 - 需要注意的是，此方法会导致子啊数据爬取工程中导致数据丢失，因此需要对出问题的地方进行标识或记录，然后在代码运行后对标识或记录的丢失位置进行重新采集

IP池的使用

搭建代理池→根据代理IP时长来进行设计使用

1. 针对短效IP

使用redis数据库来进行存储此类IP，并且设置对应的有效时长
2. 长效IP

使用本地文件的形式进行存储，

♥ 不管以上两种IP都需要根据它的有效时长来进行合理的代理IP更新

五、批量爬取和翻页规则分析

案例网址：[点击此处](#)

1.翻页规则

楼盘网——新房页面翻页规则

主资源路径(<https://cs.loupan.com/xinfang/>)+(pn)，n为数字1-97来进行翻页更新数据

- 第一页网址：<https://cs.loupan.com/xinfang/>
- 第二页网址：<https://cs.loupan.com/xinfang/p2/>
- 第三页网址：<https://cs.loupan.com/xinfang/p3/>

楼盘网——商业地产页面翻页规则

主资源路径(<https://cs.loupan.com/business/>)+(pn)，n为数字1-22来进行翻页更新数据

示例

获取站长素材([链接](#))——图片页面的前五页数据

```
python
# 导入模块
import requests

# 请求头内容
headers = {
    'cookie': '_clck=l8ww91%E2%Eg2h%E0%E2196; _clsk=7urkiw%E1767681834863%E8%E1%Ev.clarity.ms%2Fcollect',
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://sc.chinaz.com/jianli/',
}
```

```
def zhanzhang_tupian():
    # 设置爬取范围1-5
    for i in range(1,6):
        # 根据翻页规则定
        if i == 1:
            tupian_url = 'https://sc.chinaz.com/tupian/'
        else:
            tupian_url = f'https://sc.chinaz.com/tupian/index_{i}.html'
        # 代理IP
        proxy = {
            'http': '120.92.212.16:8890',
            'https': '120.92.212.16:8890',
        }
        # 发起请求
        tupian_res = requests.get(url=tupian_url, headers=headers, proxies=proxy)
        # 设置编码格式防止乱码
        tupian_res.encoding = 'UTF-8'
        # 短点处
        tupian_res

if __name__ == '__main__':
    zhanzhang_tupian()
```

♥ 补充

当我们去批量爬取数据的时候，需要非常注意网站的限流问题

1. 使用代理——减少单一IP的过量访问
2. 添加等待时间——防止单一IP在一个时间段内访问频率过高

2. 无特定规则翻页分析

潇湘书院([链接](#))

♥ 第一本书

第一页：<https://www.xxsy.net/chapter/25866531901067904/69470529228165448>

第二页：<https://www.xxsy.net/chapter/25866531901067904/69474434292962500>

第二本书

第一页：<https://www.xxsy.net/chapter/26169087201414504/70256336176203430>

第二页：<https://www.xxsy.net/chapter/26169087201414504/70256433886714439>

层级爬取

书籍章节目录：<https://www.xxsy.net/chapterlist/> + 书籍编号

书籍章节内容：<https://www.xxsy.net/chapter/> + 书籍编号+书籍章节编号

示例

```
python

# 导入模块
import requests      # 发送请求的模块
# 导入正则模块
import re

url = 'https://www.xxsy.net/'
headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://www.xxsy.net/',
    'cookie': 'newstatisticUUID=1767077440_8196224563',
}

# 构建并发起get请求
res = requests.get(url=url, headers=headers)
```

```
# 获取榜单中书的ID
rank_book_list = re.findall(r'<a href="/book/(\d{14,17})" target="_blank">.*?</a>',res.text)[-30:]

rank_chapter_dict = dict()
for i in rank_book_list[:5]:
    chapter_url = f'https://www.xxsy.net/chapterlist/{i}'
    chapter_res = requests.get(url=chapter_url, headers=headers)
    # 获取每个章节的ID
    chapter_list = re.findall(r'<a href="/chapter(.*?)" target="_blank" title=".*?" style="padding:8px 0;display:flex;">',chapter_res.text)[0:10]
    chapter_list[0] = re.search(r'/chapter(.*?)',chapter_list[0][-50:]).group(1)
    rank_chapter_dict[i] = chapter_list

for key in rank_chapter_dict:
    for chapter_id in rank_chapter_dict[key]:
        chapter_url = f'https://www.xxsy.net/chapter{chapter_id}'
        chapter_res_if = requests.get(url=chapter_url, headers=headers)
        # 获取标题
        title = re.findall(r'<h1 .*>(.*?)</h1>', chapter_res_if.text)[0]
        # 获取内容
        book_text = tuple(map(lambda x:x+'\n',re.findall(r'<p>(.*?)</p>',chapter_res_if.text)))
        # 储存至文件
        with open(f'book/{key}.txt','a',encoding='utf-8') as f:
            f.write(title+'\n')
            f.writelines(book_text)
```

流程：获取书籍编号→获取对应书籍的章节→获取章节中的具体内容

拓展

map() 函数

map函数是用来统一修改可迭代类型的函数

 python

```
test_list = [1,2,3,4,5,6]
test_list_new = list(map(lambda x:x+1,test_list))
```

3. 针对搜索规则批量爬取分析

经过分析，网站数据是根据搜索内容来进行翻页获取规则的

六、正则解析

1. 高阶函数

map

统一操作函数

语法：map(处理函数,可迭代类型)

map返回的是一个可迭代的map对象

注：map对象是不可查看的，需要转化为序列类型才可见（列表、集合、元组）

map函数是用来统一修改可迭代类型的函数

 python

```
test_list = [1,2,3,4,5,6]
test_list_new = list(map(lambda x:x+1,test_list))
```

zip

将两个参数合并为一个item对象(组合)

语法：zip(序列类型1(可以包含集合),序列类型2(可以包含集合))→合并函数

注：两个参数的元素个数必须一致

zip返回一个zip对象，需要转换为可见对象→list/tuple/dict/set

python

```
list1 = [1,2,3,4,5]
list2 = ['a','b','c','d','e']
zipped1 = tuple(zip(list1,list2))
zipped2 = list(zip(list1,list2))
zipped3 = dict(zip(list1,list2))
zipped4 = set(zip(list1,list2))
print(zipped1)
print(zipped2)
print(zipped3)
print(zipped4)
"""
((1, 'a'), (2, 'b'), (3, 'c'), (4, 'd'), (5, 'e'))
[(1, 'a'), (2, 'b'), (3, 'c'), (4, 'd'), (5, 'e')]
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e'}
{(1, 'a'), (3, 'c'), (5, 'e'), (2, 'b'), (4, 'd')}
```

filter

过滤函数

语法：filter(过滤函数,序列类型(包含集合))

filter返回一个filter对象，不可见，需要转换为可见对象

python

```
list_test = [345,454,455,322,456,754]
new_list = filter(lambda x:x>400,list_test)
print(list(new_list))
"""
[454, 455, 456, 754]
```

enumerate

枚举函数

语法：enumerate(序列类型(不包含集合))

将序列类型里面的数据与它所对应的下标进行组合，返回一个对应的item，

返回enumerate对象，不可见，需转换为序列类型

python

```
list_test = [345,454,455,322,456,754]
new_list1 = tuple(enumerate(list_test))
new_list2 = list(enumerate(list_test))
new_list3 = dict(enumerate(list_test))
new_list4 = set(enumerate(list_test))
print(new_list1)
print(new_list2)
print(new_list3)
print(new_list4)
"""
((0, 345), (1, 454), (2, 455), (3, 322), (4, 456), (5, 754))
[(0, 345), (1, 454), (2, 455), (3, 322), (4, 456), (5, 754)]
{0: 345, 1: 454, 2: 455, 3: 322, 4: 456, 5: 754}
{(0, 345), (5, 754), (3, 322), (4, 456), (1, 454), (2, 455)}
```

2. 正则解析

\s：匹配换行符→因为通配符.无法匹配换行

?: 取消贪婪模式→为了防止过多的获取内容

例：

```
python
# 内容
<h1>asjdhfkjash</h1>
# 贪婪模式
<.*> → <h1>asjdhfkjash</h1>
# 取消贪婪模式
<.*?> → <h1>,</h1>
```

()：分组→体现在findall方法与search、match方法之间的区别

3. 正则解析

♥ 案例网址

1. https://www.imdb.com/chart/top/?ref_=fn_nv_menu
2. <https://www.maoyan.com/board/4?timeStamp=1767789872419>

正则解析的优点并不是在于可以很准确的获取数据，而是它可以获取出任何你想要获取的数据

案例一

获取网站中的电影名、电影评分，电影年份、电影时长

```
python
import requests
import re

url = 'https://www.imdb.com/chart/top/?ref_=fn_nv_menu'

headers = {
    'cookie': 'session-id=133-9525601-0942401; session-id-time=2082787201l; ad-oo=0; ci=eyJhZ2VTaWduYWwiOiJBFRFVMVCIsImIzR2RwciI6ZmFsc2V9; ubid-main=130-2984191-2615442; session-token=YoLKgI4H8W+ckpct8UwBW3dolcE+Px4dE12RAKFpsa5B+utgA4wzSF6Ghxw3hhSK9zh6ohd/cb6MDEt+WYJKaP33GzQWKV8EcmnIcHkYnKxgZR/zlmKTUUFhfR3sLUt0u54gpSBL0548XnP/rMJs46CQeS715Mgr9djGMSRQobrTWKdR5ts02pqU8o+FYgtgK8pRFvNrLtRq8WnEGo0ukFNgFzqhvu3YPsjpDeHT1vx+EkCv1Y5E4B4TsUcmL04M59aYMN00WxLL+h0i+boAk0HLHw4ds+l2HbtK8U/kSL15TdCQLTDR08kBX30dMoIUHa2Qhea6jnb+S1w0xa5LFD8wTVa dNLm; csm-hit=tb:3587K4XC6086NRMSH4V8+s-BM1GD1E6FSAYGFZWCA95|1767778920598&t:1767778920598&adb:adblk_yes',
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://www.imdb.com/find/?q=top%2050%20movies&ref_=chttp_nv_srb_sm'
}

# 时间转换
def time_conversion(second):
    hour = second // 3600
    minute = (second % 3600) // 60
    return f'{hour}h{minute}m'

res = requests.get(url, headers=headers)
# 电影名
movie_title = re.findall(r'"url":".*?","name": "(.*?)", "description"', res.text)
# 电影评分
movie_rating = re.findall(r'"worstRating":1, "ratingValue":(.*?)', res.text)
# 电影年份
movie_year = re.findall(r'"releaseYear":\{"year":(\d{4})', res.text)
```

```

# 电影时长
movie_duration = re.findall(r'"seconds":(\d{4,5})', res.text)
# 转换为xh xm 格式
movie_duration_new = [time_conversion(int(i)) for i in movie_duration]

# 拼接输出
max_len = min(len(movie_title), len(movie_rating), len(movie_year), len(movie_duration_new))
for i in range(max_len):
    title = movie_title[i]
    rating = movie_rating[i]
    year = movie_year[i]
    duration = movie_duration_new[i]
    print(f'电影名称: {title}')
    print(f'评分: {rating}')
    print(f'年份: {year}')
    print(f'时长: {duration}')
    print("*"*80)

```

案例二

爬取猫眼电影TOP100榜1-10页的数据

python

```

import requests
import re

headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://www.maoyan.com/',
    'cookie': '__mta=47143448.1767771051803.1767790754242.1767791308864.15; _lxsdk_cuid=19b975d6e6fc8-00450a48ddbee6-26061a51-1fa400-19b975d6e6fc8; uuid_n_v=v1; uuid=CA56C320EB9A11F09C6983D1745B13D9E85113618BC240D390361010F21FD821; _csrf=f3bcb09860b1c551112859ec8c32f4a92c56ca73d60be976066f96037dc10791; _lx_utm=utm_source%3DBaidu%26utm_medium%3Dorganic; _lxsdk=CA56C320EB9A11F09C6983D1745B13D9E85113618BC240D390361010F21FD821; _ga=GA1.1.1487119828.1767771051; __mta=47143448.1767771051803.1767771051803.1767791295955.2; _ga_WN80P4PSY7=GS2.1.s1767789851$o2$g1$t1767791308$j46$l0$h0; _lxsdk_s=19b987c7124-7a5-7d3-24e%7C%7C31',
}

result = [i*10 for i in range(0,10)]
for i in result:
    url = f'https://www.maoyan.com/board/4?timeStamp=1767789872419&offset={i}'
    res = requests.get(url, headers=headers)
    # 电影名
    movie_title = re.findall(r'data-val="\{movieId:\d{3,7}\}">(.*?)</a></p>', res.text)
    # 主演
    starring = re.findall(r'主演: (.*?)\s*</p>', res.text)
    # 上映时间
    release_time = re.findall(r'<p class="releasetime">上映时间: (.*?)</p>', res.text)
    # 评分
    rating = re.findall(r'<p class="score"><i class="integer">(\d\.)</i><i class="fraction">(\d)</i></p>', res.text)
    rating_new = [float(i[0]+i[1]) for i in rating]
    # 输出
    max_len = min(len(movie_title), len(starring), len(release_time), len(rating_new))
    for u in range(max_len):
        title_date = movie_title[u]
        starring_date = starring[u]
        release_time_date = release_time[u]
        rating_new_date = rating_new[u]
        print(f'电影名: {title_date}')
        print(f'主演: {starring_date}')
        print(f'上映时间: {release_time_date}')
        print(f'评分: {rating_new_date}')
        print('*'*50)

```

六、XPATH解析

1.XPATH的使用

XPATH的写法

如果没有限定具体标签时，会找到同级的所有当前标签

\：表示分级

- html\body

[n]：下标限制，n为整数，**注**：XPATH中下标从1开始

- html/body/div[7]

元素[@属性="属性值"]：属性限制

- html/body/div[@class="pages"]

//：表示当前节点下任意层级的标签

- //div[@class="pages"]

2.提取数据

text

text()：提取标签当中的内容

- html/body/div[7]/div[3]/div[4]/div[1]/ul/li[1]/div[1]/h2/a/text()

@属性

@属性：获取标签中的属性值

- html/body/div[7]/div[3]/div[4]/div[1]/ul/li[1]/div[1]/h2/a/@href

实战一

使用XPATH获取[猫眼TOP100榜](#)1-10页的数据

```
python
import requests
from lxml import etree # 导入xpath方法

headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'referer': 'https://www.maoyan.com/',
    'cookie': '__mta=47143448.1767771051803.1767790754242.1767791308864.15; _lxsdk_cuid=19b975d6e6fc8-00450a48ddbee6-26061a51-1fa400-19b975d6e6fc8; uuid_n_v=v1; uuid=CA56C320EB9A11F09C6983D1745B13D9E85113618BC240D390361010F21FD821; _csrf=f3bcb09860b1c551112859ec8c32f4a92c56ca73d60be976066f96037dc10791; _lx_utm=utm_source%3DBaidu%26utm_medium%3Dorganic; _lxsdk=CA56C320EB9A11F09C6983D1745B13D9E85113618BC240D390361010F21FD821; _ga=GA1.1.1487119828.1767771051; __mta=47143448.1767771051803.1767771051803.1767791295955.2; _ga_WN80P4PSY7=GS2.1.s1767789851$o2$g1$t1767791308$j46$l0$h0; _lxsdk_s=19b987c7124-7a5-7d3-24e%7C%7C31',
}

# 评分拼接
def splice(a, b):
    rating_list = []
    max_len1 = min(len(a), len(b))
    for i in range(max_len1):
        a1 = a[i]
```

```

        b1 = b[i]
        rating_info = float(a1 + b1)
        rating_list.append(rating_info)
    return rating_list

result = [i*10 for i in range(0,10)]
for i in result:
    url = f'https://www.maoyan.com/board/4?timeStamp=1767789872419&offset={i}'
    res = requests.get(url,headers=headers)
    # res
    # 将text转换为html页面信息，因为获取的数据是字符串信息，字符串数据不具备网页结构，因此需要转换为html页面结构，才能使用XPath
    res_html = etree.HTML(res.text)
    # 电影名
    movie_title = res_html.xpath('//div[4]/div/div/div[1]/dl/dd/div/div/div[1]/p/a/text()')
    # 主演
    starring = tuple(map(lambda x:x.strip(),
res_html.xpath('//div[4]/div/div/div[1]/dl/dd/div/div/div[1]/p[2]/text()')))
    # 上映时间
    release_time = res_html.xpath('//div[4]/div/div/div[1]/dl/dd/div/div/div[1]/p[3]/text()')
    # 评分
    num1 = res_html.xpath('//div[4]/div/div/div[1]/dl/dd/div/div/div[2]/p/i[1]/text()')
    num2 = res_html.xpath('//div[4]/div/div/div[1]/dl/dd/div/div/div[2]/p/i[2]/text()')
    rating = splice(num1,num2)
    # 输出结果
    max_len = min(len(movie_title),len(starring),len(release_time),len(rating))
    for m in range(max_len):
        movie_title_m = movie_title[m]
        starring_m = starring[m]
        release_time_m = release_time[m]
        rating_m = rating[m]
        print(f'电影名: {movie_title_m}')
        print(starring_m)
        print(release_time_m)
        print(rating_m)
        print('*'*80)

```



如果碰到标签数据分散的情况下，需要将数据综合时，应当先把他们所共有的部分获取出来，然后对其单独的获取数据

实战二

获取 [潇湘书院排行榜](#) 所有书名、ID及作者

● ● ● python

```

import requests
from lxml import etree

url = 'https://www.xxsy.net/rank'

headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    'cookie': 'newstatisticUUID=1767077440_8196224563',
    'referer': 'https://www.xxsy.net/'
}

res = requests.get(url,headers=headers)
res_html = etree.HTML(res.text)
# 书名
books_name = res_html.xpath('//div[@id="app"]/div[4]/div[2]/div[2]/div/div[2]/ul/li/p/a/text()')
# 图书ID
books_id = res_html.xpath('//div[@id="app"]/div[4]/div[2]/div[2]/div/div[2]/ul/li/p/a/@href')
# 作者
books_author = res_html.xpath('//div[@id="app"]/div[4]/div[2]/div[2]/div/div[2]/ul/li/span[1]/text()')
# 拼接网址
books_https = tuple(map(lambda x:'https://www.xxsy.net'+x,books_id))

```

```
# 输出
books_list = [f'书名：{name}\n作者：{author}\n链接：{https}\n{' '*80}' for name,author,https in
zip(books_name,books_author,books_https)]

for book in books_list:
    print(book)
```

七、HTML异常处理

1.HTML异常

1. 响应数据与浏览器当中的数据存在差异
2. 1-100页，其中23出现问题，这个页面没有了

实战一

爬取[房天下新房1-10页](#)中的数据

```
python
import requests
from lxml import etree

# 清除函数
def clear_string(z, s_list):
    s = f'{z}'.join(s_list).strip().replace(' ', '')
    return s

# 装饰器，防止网页无法加载时报错结束进程
def requests_error(func):
    def requests_error_except(url):
        try:
            res = requests.get(url=url, headers=headers)
        except Exception as e:
            print(f'{url}——>{e}')
        else:
            func(res)
    return requests_error_except

@requests_error
def res_try(res):
    res_html = etree.HTML(res.text)
    elements_li = res_html.xpath('//div[@id="bx1"]/div/div[1]/div/div/div/ul/li')
    # 新房名
    house_name = tuple(map(lambda x: clear_string('', x.xpath('.//div[@class="nlcd_name"]/a/text()')), elements_li))
    # 户型
    house_type = tuple(map(lambda x: clear_string('', x.xpath('.//div[@class="house_type clearfix"]//text()')),
elements_li))
    # 地址
    address = tuple(map(lambda x: clear_string('', x.xpath('.//div[@class="address"]//@title')), elements_li))
    # 标签
    fangyuan = tuple(map(lambda x: clear_string('/', x.xpath('.//div[@class="fangyuan"]//text()')), elements_li))
    # 价格
    house_price = tuple(map(lambda x: clear_string('/', x.xpath('./div[1]/div[2]/div[5]//text()')), elements_li))
    # 电话
    house_phone = tuple(map(lambda x: clear_string('', x.xpath('.//div[@class="tel"]/p//text()')), elements_li))

    house_list = []

    max_len = min(len(house_name),len(house_type),len(address),len(fangyuan),len(house_price),len(house_phone))

    for u in range(max_len):
        a = house_name[u]
        b = house_type[u]
        c = address[u]
        d = fangyuan[u]
        e = house_price[u]
```

```

        f = house_phone[u]
        house_list.append(f'小区名：{a}，户型：{b}，地址：{c}，标签：{d}，价格：{e}，电话：{f}')

    for li in house_list:
        with open('book/houses.txt', 'a', encoding='utf-8') as f:
            f.write(f'{li}\n')

if __name__ == '__main__':
    for i in range(1, 11):
        url = f'https://newhouse.fang.com/house/s/b9{i}/'
        headers = {
            'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
            'cookie': 'global_cookie=zawiyz5btr1jlt6qzqud4vuzm10mk270vgn; city=www1; otherid=accb6736de6a81dacfbdaf6477400308; csrfToken=exWSPIZB1k-XXrJ7s1mDAWgF; unique_cookie=U_6ixbd5vz73mipuw3li39h244m2omk88nvgd*9',
        }
        if i == 4:
            url = f'https://www.google.com/'
        house = res_try(url)

```

实战二

爬取[一房网-青岛新房](#)的数据

python

```

import requests
from lxml import etree
from tqdm import trange      # 进度条

# 清除函数
def clear_string(z, s_list):
    s = f'{z}'.join(s_list).strip().replace(' ', '')
    return s

# 装饰器，防止网页无法加载时报错结束进程
def requests_error(func):
    def requests_error_except(url):
        try:
            res = requests.get(url=url, headers=headers)
        except Exception as e:
            print(f'{url}——>{e}')
        else:
            func(res)
    return requests_error_except

@requests_error
def res_try(res):
    res_html = etree.HTML(res.text)
    elements_li = res_html.xpath('//div[@class="property-item-box"]')
    # 新房名
    house_name = tuple(map(lambda x: clear_string('', x.xpath('.//p[@class="property-item-title"]/span[1]/text()')),
    elements_li))
    # 户型
    house_type = tuple(map(lambda x: clear_string('', x.xpath('.//p[3]//text()')), elements_li))
    # 地址
    address = tuple(map(lambda x: clear_string('', x.xpath('.//div[@class="property-item-main"]/p[2]//text()')),
    elements_li))
    # 房源
    fangyuan = tuple(map(lambda x: clear_string('/', x.xpath('.//p[@class="property-item-title"]/span[@class="el-tag el-tag--mini el-tag--dark"]/text()')), elements_li))
    # 价格
    house_price = tuple(map(lambda x: clear_string('/', x.xpath('.//div[@class="price-info"]/span/text()')),
    elements_li))

```

```
# 标签
house_label = tuple(map(lambda x: clear_string('/', x.xpath('.//div[@class="property-item-bottom"]//text()')),
elements_li))

house_list = []

max_len = min(len(house_name), len(house_type), len(address), len(fangyuan), len(house_price), len(house_label))

for u in range(max_len):
    a = house_name[u]
    b = house_type[u]
    c = address[u]
    d = fangyuan[u]
    e = house_price[u]
    f = house_label[u]
    house_list.append(f'小区名：{a}，户型：{b}，地址：{c}，房源：{d}，价格：{e}，标签：{f}')

for li in house_list:
    with open('book/qingdao_houses.txt', 'a', encoding='utf-8') as f:
        f.write(f'{li}\n')

if __name__ == '__main__':
    for i in trange(1, 11):
        url = f'http://www.ifang.com/newHouse/list/&/pg{i}/'
        headers = {
            'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/143.0.0.0 Safari/537.36',
        }
        res_try(url)
```

八、SQL数据库操作

1.mysql配置

打开配置文件

```
● ● ● shell
vim /etc/mysql/mysql.conf.d/mysqld.cnf
```

将配置文件中的bind-address设置为0.0.0.0

注：修改配置文件，需要暂停mysql服务

```
● ● ● shell
service mysql stop # 暂停mysql服务
service mysql start # 开启mysql服务
```

2. 测试连接数据库

本地下载pymysql模块

```
● ● ● shell
pip install pymysql
```

测试连接

python

```
from pymysql import Connect

mysql = Connect(
    host = "localhost",
    # host = "127.0.0.1",
    port = 3306,
    user = "root",
    password = "huihuia24",
    charset = "utf8"
)

msysql # 断点
```

实战一

爬取[马蜂窝-去旅行-正在热卖](#)中的数据，并存入数据库

python

```
from pymysql import Connect
import requests
from lxml import etree

url = 'https://www.mafengwo.cn/localdeals/'
headers = {
    'cookie': 'mfw_uid=69674779-4986-8e88-cbe4-c30ff333c37d;
oad_n=a%3A3%A7Bs%3A3%A22oid%22%3Bi%3A1029%3Bs%3A2%A22dm%22%3Bs%3A15%A22www.mafengwo.cn%22%3Bs%3A2%A22ft%22%3Bs%3A19%A222026-01-14+15%3A36%3A25%22%3B%7D; __mfwc=direct;
__mfwa=1768376185854.89689.1.1768376185854.1768376185854; __mfwlv=1768376185; __mfwvn=1;
uva=s%3A92%A22a%3A3%A7Bs%3A2%A22lt%22%3Bi%3A1768376186%3Bs%3A10%A22last_refer%22%3Bs%3A24%A22https%3A%2F%2Fwww.mafengwo.cn%2F%22%3Bs%3A5%A22rhost%22%3BN%3B%7D%22%3B;
__mfwurd=a%3A3%A7Bs%3A6%A22f_time%22%3Bi%3A1768376186%3Bs%3A9%A22f_rdomain%22%3Bs%3A15%A22www.mafengwo.cn%22%3Bs%3A6%A22f_host%22%3Bs%3A3%A22www%22%3B%7D; __mfwuuid=69674779-4986-8e88-cbe4-c30ff333c37d; bottom_ad_status=0;
PHPSESSID=ohha4pifncjblkb81uq9ai9i73; __omc_chl=; __omc_r=; __mfwb=76ff3f541d7e.9.direct; __mfwlt=1768377937;
w_tsfp=ltvuV0MF2utBvS0Q76Lqk0ynETsjdD84h0wpEaR0f5thQLERu5mD2YV9ucLwNHTY5sxnvd7DsZoyJTLYCJI3dwMSRMmQd9wX31uQxoIjjYdAAhM
xEZ7dWlMbJL8kuDRHe3hCNxS00jA8eIUd379yilkMsyN1zap3T014fstJ019E6KDQmI5uDW3HLFWQRzaLbjcMcucPr6g18L5a5T/Z5QiuFW18BLlK1U0bg
XtLW3Er4BLqIuBZMh//I53+SqA=',
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
}
res = requests.get(url, headers=headers)
res_html = etree.HTML(res.text)
# 获取所有的li标签
all_li = res_html.xpath('//div[@class="bd_hotSales"]/ul/li')
# 关闭连接
res.close()
# 类型
mark_tag = tuple(map(lambda x:(x.xpath('.//div[@class="mark-tag"]/text()') or ['暂无'])[0], all_li))
# 信息
res_information = tuple(map(lambda x:(x.xpath('.//div[@class="caption"]/h3/text()') or ['暂无'])[0], all_li))
# 出售情况
sell = tuple(map(lambda x:(x.xpath('.//div[@class="caption"]/span[1]/text()') or ['暂无'])[0].split('|'), all_li))
# 店铺
sold = tuple(map(lambda x:x.pop(),sell))
sell2 = tuple(map(lambda x:x[0].replace(' ', '') if x else '暂无', sell))
# 价格
price = tuple(map(lambda x: '¥0起' if (s:=''.join(x.xpath('.//div[@class="caption"]/span[2]//text()))).replace(' ', '') == '¥起' else s, all_li))

# 将数据保存至数据库
# 连接数据库
mysql = Connect(
    host = "localhost",
    # host = "127.0.0.1",
    port = 3306,
    user = "root",
    password = "huihuia24",
    charset = "utf8mb4"
)
```

```

# 开启游标
cursor = mysql.cursor()
# 创建数据库(如果数据库存在就不会创建)
cursor.execute('create database if not exists mafengwo;')
# 进入数据库
cursor.execute('use mafengwo;')
# 建表
cursor.execute('create table if not exists localdeals('
               'id int primary key auto_increment,'
               'mark_tag varchar(255) not null,'
               'res_information varchar(255) not null,'
               'sell char(30) not null,'
               'sold char(30) not null,'
               'price char(20) not null);')

# 插入数据
insert_sql = 'insert into localdeals(mark_tag,res_information,sell,sold,price) values (%s,%s,%s,%s,%s);'
"""单条数据插入"""
# cursor.execute(insert_sql, (数据))
"""

多条插入
cursor.executemany(insert_sql, data)
多条插入时date的数据格式
data = (
(mark_tag1,information1,sell1,sold1,price1),
(mark_tag2,information2,sell2,sold2,price2),
(mark_tag3,information3,sell3,sold3,price3)
)
"""

data = tuple(map(lambda x:(x[1],res_information[x[0]],sell2[x[0]],sold[x[0]],price[x[0]]),enumerate(mark_tag)))
cursor.executemany(insert_sql, data)
# 提交事务
mysql.commit()
# 关闭游标
cursor.close()
# 关闭数据库连接
mysql.close()

```

实战二

获取[gratisography网](https://gratisography.com/)中的图片链接，并保存至数据库

python

```

import requests
from pymysql import Connect
from lxml import etree
from tqdm import trange

headers = {
    'cookie': '_ga=GA1.1.828024986.1768446989;
_hjSession_2556496=eyJpZCI6IjJiYTdkNDU4LTQwNTEtNGRhOC1hNDE0LWUyMWRLZjQ1MGNiMCI6ImMiOjE3Njg0NDY5OTA5NjEsInMiOjAsInIiOjAsInNiIjowLCJzciI6MCwic2UiOjAsImZzIjoxLCJzciI6MX0=;
_hjSessionUser_2556496=eyJpZCI6ImQ1N2VlYjVmLWRhM2QtNTZmZC04MWIxLTA3MTM3OGVjODUyNyIsImNyZWZ0ZWQiOjE3Njg0NDY5OTA5NjAsImV4aXN0aW5nIjp0cnVlQ=; _ga_TNFKZTM4P0=GS2.1.s1768446989$o1$g1$t1768447380$j52$l0$h0',
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
}

mysql = Connect(
    host='localhost',
    port=3306,
    user='root',
    password='huihuia24',
    charset='utf8'
)

# 开启游标
cou = mysql.cursor()
# 创建数据库
cou.execute('create database if not exists gratisography;')
# 进入数据库

```

```

cou.execute('use gratisography')
# 创建表
cou.execute('create table if not exists img_url('
            'id int primary key auto_increment,'
            'name varchar(500) not null,'
            'data longblob not null);')
for i in trange(1, 11):
    url = f'https://gratisography.com/page/{i}/?s=girl'
    res = requests.get(url, headers=headers)
    res_html = etree.HTML(res.text)
    # 图片地址
    img_type = res_html.xpath('//div[@class="off-canvas-content"]/section[2]/div/div/article/div/a[1]/img/@src')
    img_info = tuple(map(lambda x: x.replace('-800x525', ''), img_type))
    res.close()
    # 将图片地址进行请求
    for a in range(len(img_info)):
        img_url_b = requests.get(url= img_info[a], headers=headers)
        img_s = img_url_b.content
        img_url_b.close()
        # with open(f'girl/{img_info[a]}', 'wb', ) as f:
        #     f.write(img_s)
        # 插入数据
        sql = 'insert into img_url(name,data) values(%s,%s);'
        cou.execute(sql, (img_info[a], img_s))
        # 提交
        mysql.commit()

# 关闭
cou.close()
mysql.close()

```

实战三

获取[飞猪旅行-旅游度假-丽江](#)中1-3页数据,并储存至数据库中

```

python

import requests
from lxml import etree
from pymysql import Connect

# 图片URL处理
def img_url_info(u):
    img_data_list = []
    for l in u:
        res_img = requests.get(l,headers=headers)
        res_b = res_img.content
        img_data_list.append(res_b)
        res_img.close()
    img_data_tuple = tuple(img_data_list)
    return img_data_tuple

# 链接数据库
mysql = Connect(
    host = 'localhost',
    port = 3306,
    user = 'root',
    passwd = 'huihuia24',
    charset = 'utf8'
)
# 开启游标
cur = mysql.cursor()
# 创建数据库
cur.execute('create database if not exists pachong;')
# 进入数据库
cur.execute('use pachong;')
# 建表
cur.execute('create table if not exists fliggy('

```

```
'id int primary key auto_increment,'
'tag varchar(50) not null,'
'img_info longblob not null,'
'title varchar(500) not null,'
'address varchar(500) not null,'
'hotel_star varchar(50) not null,'
'rating decimal(3,1) not null,'
'price varchar(50) not null;')
```

爬取数据

```
headers = {
    'cookie': 'lid=tb764317057; wk_cookie2=19e5a18fda9e5ad850c05da184c6a41e; wk_unb=UUpGrsNstd9bfuqdKg%3D%3D; cna=10HjIeP5NmUCAdyoN0Tl3rxr; dnk=tb764317057; tracknick=tb764317057; havana_lgc_exp=1799134684908; lgc=; cookie2=1c38e67d82ce54a5b965cd32b31e8eb9; sgcookie=E100d%2BBrIrsZ0UIv433Te5daGExY56P0chz1s%2FktWzPR9ovYrr%2BVeaP%2FkTv6%2FHHsMt2tb2kBy1Qx0vApJIO%2BTaoC8orxzKpy3fKwfQaarj6NtrZNs6hLto83RMTAI9wzlYoE; cancelledSubSites=empty; t=67bfa247584aa29997f00573aeb0c26c; csg=a83283aa; sn=; _tb_token_=77633e36e6bd6; mtop_partitioned_detect=1; _m_h5_tk=12dba58ee68d6291653b5a96272cd2b2_1768740632889; _m_h5_tk_enc=4eaa6e69520b33d1dbd2ab5fdd64c4c9; xlly_s=1; isg=BAIC-EEM37FXjcNtw0Ec6_2AUwhk0wbTvQ0kFEwbLnUgn6IZNGNW_YjMSZvjz36F; _mw_us_time_=1768734760944; x5sec=7b22733b32223a2263323331323134363939333636323639222c22617365727665723b33223a22307c434b654473387347454c61322f706345476738794d6a45774e7a51784f544d324d7a67794f7a45777a4e4c7132762f2f2f2f2f2f41513d3d227d; tfstk=gA4ZU9T8mNQZMj6ua70V8YLdA2utlqW7srMjiSVm1ADihmg2m8wH5lstcxRqL8gb1Pa1oky0pcMgfSG2u5HQhEN_uKL4wSU16q1tW53xoT65PrPT6qF1HwPNdWfmMqlgyPzlp53xo90Bog_z6RwhZfnio6ungb-ioxcM-6loNqYimFxHKvhmoqmimeAnZjJMoEccTWDKiqD0nqfE-vhmox2mobPusSYENCfdICT2ngHjbYViLUPL8fmMX5DeoE4UYckkIv8Doylai5b9qUSjEuHsVYehRFu4tjyEVlWy7-P0womUudf_ES2T8V3Pc1nzxkZSbkWMuvassg00mYIY0TqqimrulrplLx5Z0WRR2ofU_Pmk-YsYxc24Squ2wM_FnSj2x2r6JWAV0wzESzNJ-_kVr8g-6HX0gLrEwnnoi9Xk5T6-SezG1aeiJ5nKxx4hEF1G6Dnni9Xk5T6-vDD0-TY1s1',
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',

    'accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7',
    'referer': 'https://www.fliggy.com/'
}

for i in range(1,4):
    if i == 1:
        url = 'https://travelsearch.fliggy.com/index.htm?spm=181.15077045.1398723350.13.7ab1620d0HgDhv&searchType=product&keyword=%E4%B8%BD%E6%B1%9F&category=MULTI_SEARCH'
    else:
        url = f'https://travelsearch.fliggy.com/index.htm?spm=181.15077045.1398723350.13.7ab1620d0HgDhv&searchType=product&keyword=%E4%B8%BD%E6%B1%9F&category=MULTI_SEARCH&pageNum={i}'

    res_z = requests.get(url, headers=headers)
    res_z_html = etree.HTML(res_z.text)
    # 全部div
    div_all = res_z_html.xpath('//div[@class="page-products-block-left clear-fix"]/div[1]/div/div')
    # 标签
    tag = tuple(map(lambda x: x.xpath('./div[1]/a/span[1]/text()')[0], div_all))
    # 图片地址
    img_url = tuple(map(lambda x: ('https:' + x.xpath('./div[1]/a/div/img/@data-src')[0].replace('_200x200xz', '')).replace('https:https://', 'https://'), div_all))
    # 图片二进制元组
    img_info = img_url_info(img_url)
    # 标题
    title = tuple(map(lambda x: x.xpath('./div[2]/div[1]/a/h3/div/text()')[0], div_all))
    # 地址
    address = tuple(map(lambda x: (x.xpath('./div[2]/div[1]/h4/text()') or ['未知'])[0].replace(' | ', ''), div_all))
    # 酒店星级
    hotel_star = tuple(map(lambda x: (x.xpath('./div[2]/p[1]//text()') or ['未知'])[0], div_all))
    # 评分
    rating = tuple(map(lambda x: (x.xpath('./div[2]/p[3]//text()') or ['0.0'])[0], div_all))
    # 价格
    price = tuple(map(lambda x: ''.join(x.xpath('./div[3]/div/div/span//text()') or ['未知']), div_all))
    # 关闭
    res_z.close()
    # 数据拼接
    data = tuple(map(lambda x: (x[1], img_info[x[0]], title[x[0]], address[x[0]], hotel_star[x[0]], rating[x[0]], price[x[0]]), enumerate(tag)))
    # 将数据传入数据库中
    sql = 'insert into fliggy(tag, img_info, title, address, hotel_star, rating, price) values (%s,%s,%s,%s,%s,%s,%s,%s)'
    cur.executemany(sql, data)
```

```
# 提交
mysql.commit()
# 关闭
cur.close()
mysql.close()
```

3. 拓展

网络编码转换

```
python
from urllib import parse
# 导入urllib
s = '香港'
print(parse.quote(s))
# %E9%A6%99%E6%B8%AF
```

九、多线程-异步爬虫

💕 实现批量爬虫，多爬虫统一运行

1. 多线程

多线程的本质就是多个“人”一起爬取内容

多线程多用于IO(读写)的操作(文件打开，数据存储，爬虫等待)

多线程的使用

不使用多线程

```
python
import time

start = time.time()

def sleep_time(x):
    print(x)
    time.sleep(2)

for i in range(1,11):
    sleep_time(i)

print(time.time() - start)
# 运行时间20秒
```

使用多线程

```
python
import time
from threading import Thread    # 线程模块

start = time.time()

def sleep_time(x):
    print(x)
    time.sleep(2)

tasks = [Thread(target=sleep_time,args=(i,))for i in range(1,11)]
for t in tasks:
    # 启动线程任务
```

```
t.start()
for x in tasks:
    # 守护线程
    x.join()

print(time.time() - start)
# 运行时间2秒
```

实战

使用多线程对[飞卢小说网](#)中的推理灵异内的书籍内容进行爬取

python

```
import requests
import re
from lxml import etree
from threading import Thread

def book_id():
    book_id_url = 'https://b.faloo.com/'
    headers = {
        'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
    }
    book_id_res = requests.get(url=book_id_url, headers=headers)
    book_id_html = etree.HTML(book_id_res.text)
    books_id = tuple(map(lambda x: re.findall(r'//b\.faloo\.com/(\d+)\.html\?1', x)
[0],book_id_html.xpath('//div[@class="TenLeft"]/div[2]/div[1]/div[2]/ul/li/div/a/@href')))
    return books_id

def book_count(id_info):
    f = open(f'book/{id_info}.txt', 'a', encoding='utf-8')
    for i in range(1,41):
        book_count_url = f'https://b.faloo.com/{id_info}_{i}.html'
        headers = {
            'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
            'cookie': 'host4chongzhi=b.faloo.com; readline=1; fontFamily=1; fontsize=16; vip_img_width=3; font_Color=666666; nc_rela=2; novelrelative=1458703; favorates28=1458703%2C2%7C1436865%2C4; autobuychapters28=1458703%2C2%7C1436865%2C4; bgcolor=%23FFFFFFE; curr_url=https%3A//b.faloo.com/1436865.html%3F1',
        }
        book_count_res = requests.get(url=book_count_url, headers=headers)
        book_count_html = etree.HTML(book_count_res.text)
        text = tuple(map(lambda x:x+'\n',book_count_html.xpath('//div[@class="noveContent"]/p/text()')))
        f.writelines(text)

if __name__ == '__main__':
    book_id_info = book_id()
    tasks = [Thread(target=book_count,args=(i,)) for i in book_id_info]
    for i in tasks:
        i.start()
    for i in tasks:
        i.join()
```

2. 异步-协程

异步的使用

💕 只用于一个场景→等待IO，遇到IO就切换到下一个任务

python

```
import asyncio # 异步模块
```

```
# 声明异步函数
async def count(x):
    # await 标记需要等待的IO操作
    await asyncio.sleep(1)
    print(x)

# 异步容器 任务表
loop = asyncio.new_event_loop()
# 启动任务(单次)
# loop.run_until_complete(count(1))
# 启动任务(多次)
ap = [loop.create_task(count(i)) for i in range(10)]
loop.run_until_complete(asyncio.wait(ap))
```

实战

使用异步对[赶集招聘](#)中的数据进行爬取

需要导入异步请求模块

● ● ● shell

```
pip install aiohttp
```

● ● ● python

```
import asyncio
import aiohttp
from lxml import etree

async def download(page:int):
    url = f'https://cs.ganji.com/tech/pn{page}/'
    headers = {
        'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36',
        'cookie': 'id58=CkwAG2lvK0U6FAFwEvZYAg==; 58tj_uuid=dac3aab2-2276-469e-a743-f8b8d18f8420; ngj_city_id=2; ngj_city_name=%E4%B8%8A%E6%B5%B7; ngj_city_listname=sh; new_uv=2',
    }
    async with aiohttp.ClientSession() as session:
        # 使用代理的写法
        # async with session.get(url, headers=headers, proxy='http://代理ip:端口号') as res:
        async with session.get(url, headers=headers) as res:
            # text()字符串
            # read()二进制数据
            # json()的json类型的数据
            text = await res.text()
            return text

# 无法直接解析数据
# loop = asyncio.new_event_loop()
# tasks = [loop.create_task(download(i)) for i in range(1,11)]
# loop.run_until_complete(asyncio.wait(tasks))

# 解析函数
def func(fut):
    result = fut.result()
    html = etree.HTML(result)
    a_all = html.xpath('//div[@class="position"]/div[4]/div/a[@class="ibox"]')
    # 标题
    title = tuple(map(lambda x:x.xpath('./ul[1]/li[1]/text()')[0],a_all))
    # 薪水
    salary = tuple(map(lambda x: ''.join(x.xpath('./ul[1]/li[2]//text()')).strip('\n '),a_all))
    # 标签
    label = tuple(map(lambda x: '/'.join(x.xpath('./ul[1]/div/span/span/text()')),a_all))
    # 公司
    company = tuple(map(lambda x: x.xpath('./ul[2]/li[1]/object/a/text()')[0].strip('\n '), a_all))
    # 地址
    address = tuple(map(lambda x: ''.join(x.xpath('./ul[2]/li[2]//text()')).strip('\n ').replace(' | ', '| '),a_all))
```

```
# 拼接数据
data = tuple(map(lambda x:(x[1],salary[x[0]],label[x[0]],company[x[0]],address[x[0]]), enumerate(title)))
pass

# 分配任务的异步函数
async def main():
    tasks = list()
    for page in range(1,11):
        c = download(page) # 创建协程对象
        task = asyncio.ensure_future(c) # 将任务对象进行异步封装
        task.add_done_callback(func) # 指定回调函数
        tasks.append(task)
    await asyncio.gather(*tasks)

if __name__ == '__main__':
    asyncio.run(main())
```

十、selenium

1.selenium工具准备

- Chrome浏览器
 - Chrome驱动
- 下载地址：<https://googlechromelabs.github.io/chrome-for-testing/>
- 驱动版本查看
- Chrome浏览器地址栏输入:chrome://version

2.selenium的使用

需要安装第三方模块

```
● ● ● shell
pip install selenium
```

当使用selenium这种自动化程序时，会有对应的“指纹”信息

在网页控制台终端输入：

```
● ● ● shell
navigator.webdriver
```

反检测模块

```
● ● ● shell
pip install undetected_chromedriver
```

示例

```
python
from selenium import webdriver
import undetected_chromedriver as webdriver_1

# 使用代理
options = webdriver.ChromeOptions()
options.add_argument('--proxy-server=socks4://39.104.16.201:3128')

chrome = webdriver_1.Chrome()
# 注：在运行前尽量关闭所有Google浏览器
chrome.get('https://www.youku.com/ku/webhome')

input()
```

十一、Scrapy框架基础

1.Scrapy框架

Scrapy爬虫框架是基于异步的爬虫框架

主要封装的的功能：多线程+异步操作(twisted网络异步通信架构)
多机协同工作(redis服务)

Scrapy框架是一个完整的爬虫

完整的爬虫包括：爬虫构造，数据获取，数据库初始化，数据保存

2.模块准备

```
shell
pip install scrapy
```

3. 框架的使用

创建项目

```
shell
scrapy startproject 项目名
```

项目设置

构建爬虫

数据存储

十二、js逆向-代码反编译

案例网址：[oklink](#)

1.js逆向

有些网站不仅仅通过一些常规字段来进行身份检验，如：cookie、sessionid，因为这些常见字段具有一定缺陷，那么此时就会由前端js代码来生成一个对应的校验字段，来进行身份校验，这种校验是算法的校验。

💖 x-apikey比较：
LWIZMWUtNDU0Ny05Mjk5LWI2ZDA3Yjc2MzFhYmEyYzkwM2NjfDI4ODAxODAzNDA1MzAzMTA=

当看到某个加密字段每次都会变化，但会有一段始终不变，可以推断它是一个签名算法

签名算法：由一个或者多个变化的值，加上固定值[盐]，通过加密算法得来

2. 寻找目标位置

Ctrl + shift + f 进行查找

这些数据是通过加密算法来的：看搜索出来的上下文当中是否包含[parse、encrypt、md5、rsa、aes]这类型的字符

得到的js代码

```

● ● ● js

function getApiKey() {
    var e = (new Date).getTime(), t = encryptApiKey();
    return e = encryptTime(e),
    comb(t, e)
}
API_KEY = "a2c903cc-b31e-4547-9299-b6d07b7631ab"
function encryptApiKey() {
    var e = API_KEY, t = e.split(""), n = t.splice(0, 8);
    return e = t.concat(n).join("")
}

function encryptTime(e) {
    var t = (1 * e + s).toString().split("")
    , n = parseInt(10 * o.o.mathRandom(), 10)
    , r = parseInt(10 * o.o.mathRandom(), 10)
    , i = parseInt(10 * o.o.mathRandom(), 10);
    return t.concat([n, r, i]).join("")
}

function comb(e, t) {
    var n = "".concat(e, "|").concat(t);
    return a.A.btoa(n)
}

```

转换后

```

● ● ● python

from time import time
from random import randint
from base64 import b4encode
import requests

def getApiKey():
    e = int(time()*1000)
    t = encryptApiKey()
    e = encryptTime(e)
    apikey = comb(t, e)
    return apikey

API_KEY = "a2c903cc-b31e-4547-9299-b6d07b7631ab"
def encryptApiKey():
    e = API_KEY
    t = list(e)
    n,t = t[:8],t[8:]
    t.extend(n)
    e = ''.join(t)
    return e

def encryptTime(e):
    t = list(str((1 * e + 111111111111)))
    n = str(randint(0,9))
    r = str(randint(0,9))
    i = str(randint(0,9))

```

```

t.extend([n,r,i])
return ''.join(t)

def comb(e, t):
    n = f'{e}|{t}'
    apikey = b64encode(n.encode()).decode()
    return apikey

key = getApiKey()

url = f'https://www.oklink.com/api/explorer/v1/btc/blocks?offset=0&limit=100&t={int(time()*1000)}'
headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36',
    'cookie': 'devId=db003635-abc4-45aa-94d4-3698e67bde1e; ok_site_info=9Fj0ikHdpRnblJCLiskTJx0SPJi0iUGZvNmIsIyUVJi0i42bpdWZyJye; locale=zh_CN; ok-exp-time=1769069055106; first_ref=https%3A%2F%2Fwww.qklw.com%2F; fingerprint_id=db003635-abc4-45aa-94d4-3698e67bde1e; fp_s=-1; okg.currentMedia=md; traceId=2020190762744410001; __cf_bm=fQb5tR0qasRo6vByWpjb4Jl0_2Xqq3HrpX0hnayvhkY-1769076274-1.0.1.1-H0RhLlq0B6gzsJAlkGkDRb5KM4cQ1k.MzM8Ik0vN10Hdpgt3UEN7i44WjbxbpA78Jf0q6gI0U2lZwiXkLJHRn2seNjbwDLeYzS3uVD9Z8MQ; ok-ses-id=iVwVAiHXnes+wD/HMLU0A0QIZ2QKUX9KZ18WRiIiM/TsdL6ALbbVAT/ky0yxWCINLay9JBLxKLbEE0KuiCMi2C2Gk/vccjYy1+SQojTseDiF2IH2Yhdn4HfS5fVN/zbP; _monitor_extras={"deviceId":"K7C9deCj86-hyeCzzPKH5t","eventId":98,"sequenceNumber":98}',
    'x-apikey': key,
    'devId': 'db003635-abc4-45aa-94d4-3698e67bde1e'
}
res = requests.get(url, headers=headers)
res

```

3. 练习

vivo社区

使用js逆向对[vivo社区](#)进行爬取

md5的加密特征

- 不可逆的加密算法
- 明文与密文一对一
- 进行加密后的结果为32位16进制的字符组成 $\implies 0-9, a-f$

python

```

import requests
import json
from hashlib import md5
from time import time
from random import randint

for page in range(1,11):
    # js
    # e. params.nonce = Object(je.md5)(n + "" + parseInt(1e7 * Math.random(), 10) + 1, 32)
    times = int(time() * 1000)
    nonce = md5(f'{times}{randint(1000000, 9999999) + 1}'.encode()).hexdigest() # hexdigest()获取32位加密结果

    url = 'https://bbs.vivo.com.cn/api/community/index'
    headers = {
        'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36',
        'cookie': 'cookieId=2af71ac1-d2eb-1171-be0a-a4da24190f7d1767335490049; sessionId=9ad7e8ac-3f6b-abc2-5611-caf9e978768f',
        'referer': 'https://bbs.vivo.com.cn/newbbs/',
        'content-type': 'application/json; charset=UTF-8',
    }
    # 因为前后端数据传输需要的是json格式的数据，可能需要对data进行序列化操作
    data = json.dumps({"lastId": "", "pageNum": page, "pageSize": 10, "imgSpecs": ["t577x324", "t577x4096"], "timestamp": times, "nonce": nonce})
    res = requests.post(url, headers=headers, data=data)

```

4.HOOK

因为有些网站会干扰调试，那么此时就需要进行HOOK注入，替换对应的索引文件

HOOK注入

十三、js逆向-webpack

1.常见的加密算法

- 签名算法

不可逆的加密算法，通常明文和密文是一对一关系，在进行加密的时候，在明文前后加入一些固定的字符串(盐)进行加密
⇒md5加密，sha系列

- MD5的密文⇒16进制的字符组成0-9，a-f

- 对称加密算法

可逆的算法，可以加密也可以解密，明文和密文之间是对称的关系，对称加密往往要传入3个参数

- 明文(data、messge)
 - 钥匙(key)
 - 偏移量(iv)

aes、des,加密后的内容⇒a-z,A-Z,0-9,+,/,=

- 非对称加密算法

明文和密文存在一对多的关系，可以解密，会存在私钥、公钥，公钥只能用来加密，私钥用来解密

2.python调用js代码

模块准备：pyexecjs

```
● ● ● shell
pip install pyexecjs
```

execjs模块使用示例

```
● ● ● js
function a(x,y){
    return x+y;
}
```

```
● ● ● python
# 导入模块
import execjs

# 加载js代码
with open('text.js', 'r', encoding='utf-8') as f:
    nodejs = f.read()

# 转换为可执行的代码
ctx = execjs.compile(nodejs)
# call方法：调用js中的函数，参数1：函数名，参数2：传入值
a = ctx.call('a',3,5)
print(a)
# 8
```

案例展示

1.oklink

```
js
API_KEY = "a2c903cc-b31e-4547-9299-b6d07b7631ab"
function encryptApiKey() {
    var e = API_KEY
        , t = e.split("")
        , r = t.splice(0, 8);
    return e = t.concat(r).join("")
}
p = 11111111111111
function encryptTime(e) {
    var t = (1 * e + p).toString().split("")
        , r = parseInt(10 * Math.random(), 10)
        , n = parseInt(10 * Math.random(), 10)
        , o = parseInt(10 * Math.random(), 10);
    return t.concat([r, n, o]).join("")
}

function comb(e, t) {
    var r = "".concat(e, "|").concat(t);
    return Buffer(r).toString('base64')
}

function getApiKey() {
    var e = (new Date).getTime()
        , t = encryptApiKey();
    return e = encryptTime(e),
        comb(t, e)
}
```

```
python
import execjs

with open('oklink.js','r',encoding='utf-8') as f:
    nodejs = f.read()

ctx = execjs.compile(nodejs)
a = ctx.call('getApiKey')
print(a)
```

2.vivo社区

```
js
var n = Date.now();

function md5(e, t) {
    function r(e, t) {
        return e << t | e >>> 32 - t
    }
    function n(e, t) {
        var r, n, u, o, l;
        return u = 2147483648 & e,
            o = 2147483648 & t,
            l = (1073741823 & e) + (1073741823 & t),
            (r = 1073741824 & e) & (n = 1073741824 & t) ? 2147483648 ^ l ^ u ^ o : r | n ? 1073741824 & l ? 3221225472 ^ l ^ u ^ o : 1073741824 ^ l ^ u ^ o : l ^ u ^ o
    }
    function u(e, t, u, o, l, i, c) {
        return e = n(e, n(n(function(e, t, r) {
            return e & t | ~e & r
        })(t, u, o), l), c)),
            n(r(e, i), t)
    }
}
```

```
function o(e, t, u, o, l, i, c) {
    return e = n(e, n(n(function(e, t, r) {
        return e & r | t & ~r
    })(t, u, o), l), c)),
    n(r(e, i), t)
}
function l(e, t, u, o, l, i, c) {
    return e = n(e, n(n(function(e, t, r) {
        return e ^ t ^ r
    })(t, u, o), l), c)),
    n(r(e, i), t)
}
function i(e, t, u, o, l, i, c) {
    return e = n(e, n(n(function(e, t, r) {
        return t ^ (e | ~r)
    })(t, u, o), l), c)),
    n(r(e, i), t)
}
function c(e) {
    var t, r = "", n = "";
    for (t = 0; t ≤ 3; t++)
        r += (n = "0" + (e >>> 8 * t & 255).toString(16)).substr(n.length - 2, 2);
    return r
}
var d, a, s, p, f, h, m, A, g, v = e, b = Array();
for (b = function(e) {
    for (var t, r = e.length, n = r + 8, u = 16 * ((n - n % 64) / 64 + 1), o = Array(u - 1), l = 0, i = 0; i < r;
    )
        l = i % 4 * 8,
        o[t = (i - i % 4) / 4] = o[t] | e.charCodeAt(i) << l,
        i++;
    return l = i % 4 * 8,
    o[t = (i - i % 4) / 4] = o[t] | 128 << l,
    o[u - 2] = r << 3,
    o[u - 1] = r >>> 29,
    o
}(v),
h = 1732584193,
m = 4023233417,
A = 2562383102,
g = 271733878,
d = 0; d < b.length; d += 16)
    a = h,
    s = m,
    p = A,
    f = g,
    m = i(m = i(m = i(m = i(m = l(m = l(m = l(m = l(m = o(m = o(m = o(m = o(m = u(m = u(m = u(m = u(m, A = u(A, g
= u(g, h = u(h, m, A, g, b[d + 0], 7, 3614090360), m, A, b[d + 1], 12, 3905402710), h, m, b[d + 2], 17, 606105819), g,
h, b[d + 3], 22, 3250441966), A = u(A, g = u(g, h = u(h, m, A, g, b[d + 4], 7, 4118548399), m, A, b[d + 5], 12,
1200080426), h, m, b[d + 6], 17, 2821735955), g, h, b[d + 7], 22, 4249261313), A = u(A, g = u(g, h = u(h, m, A, g, b[d
+ 8], 7, 1770035416), m, A, b[d + 9], 12, 2336552879), h, m, b[d + 10], 17, 4294925233), g, h, b[d + 11], 22,
2304563134), A = u(A, g = u(g, h = u(h, m, A, g, b[d + 12], 7, 1804603682), m, A, b[d + 13], 12, 4254626195), h, m,
b[d + 14], 17, 2792965006), g, h, b[d + 15], 22, 1236535329), A = o(A, g = o(g, h = o(h, m, A, g, b[d + 1], 5,
4129170786), m, A, b[d + 6], 9, 3225465664), h, m, b[d + 11], 14, 643717713), g, h, b[d + 0], 20, 3921069994), A =
o(A, g = o(g, h = o(h, m, A, g, b[d + 5], 5, 3593408605), m, A, b[d + 10], 9, 38016083), h, m, b[d + 15], 14,
3634488961), g, h, b[d + 4], 20, 3889429448), A = o(A, g = o(g, h = o(h, m, A, g, b[d + 9], 5, 568446438), m, A, b[d +
14], 9, 3275163606), h, m, b[d + 3], 14, 4107603335), g, h, b[d + 8], 20, 1163531501), A = o(A, g = o(g, h = o(h, m,
A, g, b[d + 13], 5, 2850285829), m, A, b[d + 2], 9, 4243563512), h, m, b[d + 7], 14, 1735328473), g, h, b[d + 12], 20,
2368359562), A = l(A, g = l(g, h = l(h, m, A, g, b[d + 5], 4, 4294588738), m, A, b[d + 8], 11, 2272392833), h, m, b[d
+ 11], 16, 1839030562), g, h, b[d + 14], 23, 4259657740), A = l(A, g = l(g, h = l(h, m, A, g, b[d + 1], 4,
2763975236), m, A, b[d + 4], 11, 1272893353), h, m, b[d + 7], 16, 4139469664), g, h, b[d + 10], 23, 3200236656), A =
l(A, g = l(g, h = l(h, m, A, g, b[d + 13], 4, 681279174), m, A, b[d + 0], 11, 3936430074), h, m, b[d + 3], 16,
3572445317), g, h, b[d + 6], 23, 76029189), A = l(A, g = l(g, h = l(h, m, A, g, b[d + 9], 4, 3654602809), m, A, b[d +
12], 11, 3873151461), h, m, b[d + 15], 16, 530742520), g, h, b[d + 2], 23, 3299628645), A = i(A, g = i(g, h = i(h, m,
A, g, b[d + 0], 6, 4096336452), m, A, b[d + 7], 10, 1126891415), h, m, b[d + 14], 15, 2878612391), g, h, b[d + 5], 21,
4237533241), A = i(A, g = i(g, h = i(h, m, A, g, b[d + 12], 6, 1700485571), m, A, b[d + 3], 10, 2399980690), h, m, b[d
+ 10], 15, 4293915773), g, h, b[d + 1], 21, 2240044497), A = i(A, g = i(g, h = i(h, m, A, g, b[d + 8], 6, 1873313359),
m, A, b[d + 15], 10, 4264355552), h, m, b[d + 6], 15, 2734768916), g, h, b[d + 13], 21, 1309151649), A = i(A, g = i(g,
h = i(h, m, A, g, b[d + 4], 6, 4149444226), m, A, b[d + 11], 10, 3174756917), h, m, b[d + 2], 15, 718787259), g, h,
b[d + 9], 21, 3951481745),
```

```

    h = n(h, a),
    m = n(m, s),
    A = n(A, p),
    g = n(g, f);
    return 32 == t ? c(h) + c(m) + c(A) + c(g) : c(m) + c(A)
}

function res(){
    data = md5(n + "" + parseInt(1e7 * Math.random(), 10) + 1, 32)
    return data;
}

console.log(res())
```

python

```
import execjs

with open('vivo.js','r',encoding='utf-8') as f:
    nodejs = f.read()

ctx = execjs.compile(nodejs)
a = ctx.call('res')
print(a)
```

3.webpack

代码容器——因为一个实际的模块当中存在非常多的方法，并且有可能存在相互反复的调用，所以我们不能保证一个一个去找到，所有的配置函数，因此应该找到调用这个加密函数的代码容器来进行执行当中的代码

💖 Promise(函数)-then(参数)

js

```
Promise(function).then(x)
```

Promise里面的function函数的结果会传入到then方法的参数当中

十四、 滑块验证码

验证码是通过什么来校验的

ID	data
abc01	170px-175px

1. 案例

案例网址：[慈善中国](#)

清除网站cookie



2.ddddocr

下载ddddocr模块

```
● ● ● shell
```

```
pip install dddocr
```

实例

```
● ● ● python
```

```
import requests
from dddocr import DdddOcr
from base64 import b64decode,b64encode

# 创建ocr实例
# ocr:检测文字
# det:用作目标检测，如扫码
ocr = DdddOcr(ocr=False,det=False)

get_img_url = 'https://cszg.mca.gov.cn/biz/ma/csmh/filter/getSlideCaptcha.html'
img_headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36'
}
img_res = requests.get(get_img_url, headers=img_headers)
oriImage,cutImage = img_res.json().get('c').get('oriImage'),img_res.json().get('c').get('cutImage')
ori_img,cut_img = b64decode(oriImage),b64decode(cutImage)
with open(f'img/ori_img.png', 'wb') as f:
    f.write(ori_img)

with open(f'img/cut_img.png', 'wb') as f:
    f.write(cut_img)

position = ocr.slide_match(cut_img,ori_img,simple_target=True)['target']
print(position)
x1,y1,x2,y2 = position

slide_cap_url = f'https://cszg.mca.gov.cn/biz/ma/csmh/filter/slideCaptchaCheck.html?slidevalue={b64encode(str(x1).encode()).decode()}'
slide_cap_headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36',
    'cookie': f'https_waf_cookie=92fbb60c-23e4-4884d27f5352a8a01883cce7d8f068eed942; JSESSIONID={img_res.cookies.get('JSESSIONID')}',
}
slide_cap_res = requests.get(slide_cap_url, headers=slide_cap_headers)
print(slide_cap_res.json())
```

3.opencv-python

下载模块

```
● ● ● shell
```

```
pip install opencv-python
```

```
● ● ● python
```

```
import cv2

# 读取背景图片和缺口图片
cut_img = cv2.imread('img/cut_img.png')
ori_img = cv2.imread('img/ori_img.png')
# 识别图片边缘
cut_edge = cv2.Canny(cut_img, 100, 200)
ori_edge = cv2.Canny(ori_img, 100, 200)
# 转换图片格式
cut_pic = cv2.cvtColor(cut_edge, cv2.COLOR_GRAY2BGRA)
ori_pic = cv2.cvtColor(ori_edge, cv2.COLOR_GRAY2BGRA)
# 缺口匹配
```

```
res = cv2.matchTemplate(ori_pic, cut_pic, cv2.TM_CCORR_NORMED)
# 寻找最佳匹配
position = cv2.minMaxLoc(res)
print(position)
```

十四、字体反爬

案例网址：[懂车帝](#)

1. 字体反爬原理

实际数据是由一个网络编码来进行占位，后期通过js脚本去将对应的编码替换成对应的“字符”

字体查看工具:[BEJSON](#)

python

```
import requests
```

[illegible]

```
sh_price = tuple(map(lambda x:x.get('sh_price'),res.json().get('data').get('search_sh_sku_info_list')))
sh_price = tuple(map(lambda x:x.split('.'),sh_price))
sh_Price = []
for i,f in sh_price:
    I = ''
    F = ''
    for i_test in i:
        I += font_dict[i_test]
    for f_test in f:
        F += font_dict[f_test]
    sh_Price.append(I+'.'+F)
```

```
official_price = tuple(map(lambda x:x.get('official_price'),res.json().get('data').get('search_sh_sku_info_list'))))
official_price = tuple(map(lambda x:x.split('.'),official_price))
official_Price = []
for i,f in official_price:
    I = ''
    F = ''
    for i_test in i:
        I += font_dict[i_test]
```

```
for f_test in f:
    F += font_dict[f_test]
official_Price.append(I+'.'+F)
```

2.fontTools

模块准备

● ● ● shell

```
pip install fontTools
```

案例网址：[猫眼电影-国内票房榜](#)

● ● ● python

```
import requests
from fontTools.ttLib import TTFont # 加载、读取原始字体文件的方法
from lxml import etree

# 读取字体文件
Font_Dict = dict()
font1 = TTFont('猫眼font/20a70494.woff')
# 保存为XML文件
font1.saveXML('猫眼font/20a70494.xml')
index1 = tuple(map(lambda x:x.lower().replace('uni',r'\u'),font1.getGlyphOrder()[2:]))
font_dict1 = dict(zip(index1,[7,3,6,1,2,8,0,4,9,5]))

font2 = TTFont('猫眼font/e3dfe524.woff')
# 保存为XML文件
font2.saveXML('猫眼font/e3dfe524.xml')
index2 = tuple(map(lambda x:x.lower().replace('uni',r'\u'),font2.getGlyphOrder()[2:]))
font_dict2 = dict(zip(index2,[9,2,4,1,5,3,6,8,0,7]))

font3 = TTFont('猫眼font/75e5b39d.woff')
# 保存为XML文件
font3.saveXML('猫眼font/75e5b39d.xml')
index3 = tuple(map(lambda x:x.lower().replace('uni',r'\u'),font3.getGlyphOrder()[2:]))
font_dict3 = dict(zip(index3,[0,3,8,2,6,7,9,4,1,5]))

font4 = TTFont('猫眼font/2a70c44b.woff')
# 保存为XML文件
font4.saveXML('猫眼font/2a70c44b.xml')
index4 = tuple(map(lambda x:x.lower().replace('uni',r'\u'),font4.getGlyphOrder()[2:]))
font_dict4 = dict(zip(index4,[7,5,3,9,0,2,6,4,1,8]))
Font_Dict.update(font_dict1)
Font_Dict.update(font_dict2)
Font_Dict.update(font_dict3)
Font_Dict.update(font_dict4)

url = 'https://www.maoyan.com/board/1'
headers = {
    'user-agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36',
    'cookie': '__mta=47143448.1767771051803.1769171627927.1769171630945.34; _lxsdk_cuid=19b975d6e6fc8-00450a48ddbbee6-26061a51-1fa400-19b975d6e6fc8; uuid_n_v=v1; uuid=CA56C320EB9A11F09C6983D1745B13D9E85113618BC240D390361010F21FD821; _lxsdk=CA56C320EB9A11F09C6983D1745B13D9E85113618BC240D390361010F21FD821; _ga=GA1.1.1487119828.1767771051; _csrf=41b68d1e6a40d78c0860c78aa6776dce2b5ef4c99bf4e27f8ca934ccd6bafc18; global-guide-isclose=true; hotMovieIds=1478868,1322627,1498191,1142033,78463,1525137,1504573,338414,1487265,1422798,1355532,356895,1528369,1529812,1399234,1405509,1504556,1425266,1603579,1432807,1618267,1323,1573941,1535383,1299948,1572282,1614889,1548343,1502290,1428854,1547279,346650,1559720,1393510,1526767,1501173,1485017,376806,4430,247161,583,1502849,1302195,7284,285540,1295,1340,233631,32124,1487834,1528899; old-moviepage-ci=70; __mta=47143448.1767771051803.1769171570301.1769171591436.8; _lx_utm=utm_source%3DBaidu%26utm_medium%3Dorganic; _ga_WN80P4PSY7=GS2.1.s1769171394$o6$g1$t1769171630$j57$l0$h0; _lxsdk_s=19bead51e7d-5c-fbd-cda%7C%7C15',
    'referer':'https://www.maoyan.com/board/4?timeStamp=1767789872419&offset=0',
}
```

```
res = requests.get(url,headers=headers)
res_html = etree.HTML(res.text)
res
realtime = res_html.xpath('//dl[@class="board-wrapper"]/dd/div/div/div[2]/p[1]/span/span/text()')
total_boxoffice = res_html.xpath('//dl[@class="board-wrapper"]/dd/div/div/div[2]/p[2]/span/span/text()')
realtime = tuple(map(lambda x:x.split('.'), realtime))
total_boxoffice = tuple(map(lambda x:x.split('.'), total_boxoffice))

# 替换
def replace(l,s:str):
    L = []
    for i,f in l:
        I = ''
        F = ''
        for i_test in i:
            I += str(Font_Dict[i_test.encode('unicode-escape').decode()])
        for f_test in f:
            F += str(Font_Dict[f_test.encode('unicode-escape').decode()])
        L.append(I+s+F)
    return L

realtime = replace(realtime, '.')
total_boxoffice = replace(total_boxoffice, '.')

print(realtime)
print(total_boxoffice)
```