山东大学
SHANDONG UNIVERSITY

数据搜索：scrapy爬虫架构

时间：周二下午5-6节

腾讯会议：762-7402-4410

授课人：徐晶晶

联系方式：xujj@sdu.edu.cn，教研楼323

课程内容

一、概述

Scrapy 是用Python实现的一个为了爬取网站数据、提取结构性数据而编写的应用框架。

Scrapy常应用在包括数据挖掘，信息处理或存储历史数据等一系列的程序中。

通常我们可以很简单的通过Scrapy框架实现一个爬虫，抓取指定网站的内容或图片。

二、Scrapy五大基本构成:

- ✓ Scrapy引擎(Scrapy Engine)
- ✓调度器(Scheduler):
- ✓下载器(Downloader)
- ✓爬虫（Spider）
- ✓实体管道(Item Pipeline)

二、Scrapy五大基本构成:引擎

(1)、Scrapy引擎(Scrapy Engine):

Scrapy引擎是整个框架的核心.它用来控制调试器、下载器、爬虫。实际上，引擎相当于计算机的**CPU**，它控制着整个流程。也就是说，负责**Spider**、**ItemPipeline**、**Downloader**、**Scheduler**中间的通讯，信号、数据传递等。

二、Scrapy五大基本构成:调度器

(2)、调度器(Scheduler):

调度器，说白了把它假设成为一个URL（抓取网页的网址或者说是链接）的优先队列，由它来决定下一个要抓取的网址是什么，同时去除重复的网址（不做无用功）。用户可以自己的需求定制调度器。它负责接受引擎发送过来的Request请求，并按照一定的方式进行整理排列，入队，当引擎需要时，交还给引擎。

二、Scrapy五大基本构成:下载器

(3)、下载器(Downloader):

下载器，是所有组件中负担最大的，它用于高速地下载网络上的资源。Scrapy的下载器代码不会太复杂，但效率高，主要的原因是Scrapy下载器是建立在twisted这个高效的异步模型上的(其实整个框架都在建立在这个模型上的)。负责下载Scrapy Engine(引擎)发送的所有Requests请求，并将其获取到的Responses交还给Scrapy Engine(引擎)，由引擎交给Spider来处理。

二、Scrapy五大基本构成:爬虫

(4)、 爬虫（Spider）：

爬虫，是用户最关心的部分。用户定制自己的爬虫(通过定制正则表达式等语法)，用于从特定的网页中提取自己需要的信息，即所谓的实体(Item)。 用户也可以从中提取出链接,让Scrapy继续抓取下一个页面。它负责处理所有Responses,从中分析提取数据，获取Item字段需要的数据，并将需要跟进的URL提交给引擎，再次进入Scheduler(调度器).

二、Scrapy五大基本构成:实体管道

(5)、 实体管道(Item Pipeline):

实体管道，用于处理爬虫(spider)提取的实体。主要的功能是持久化实体、验证实体的有效性、清除不需要的信息。它负责处理Spider中获取到的Item，并进行进行后期处理（详细分析、过滤、存储等）的地方。

二、Scrapy其他构成：下载中间件、Spider中间件

(6)、下载中间件（Downloader Middlewares）：

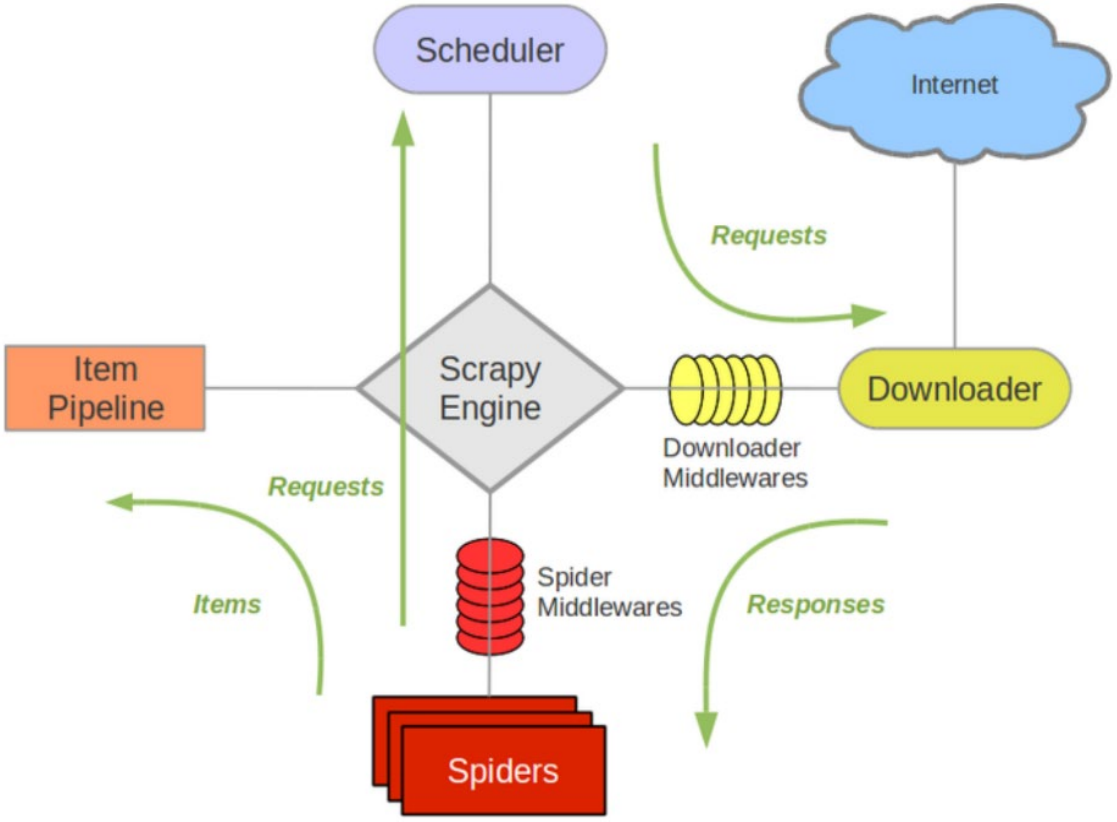
你可以当作是一个可以自定义扩展下载功能的组件。

(7)、Spider中间件（Spider Middlewares）：

你可以理解为是一个可以自定扩展和操作引擎和Spider中间通信的功能组件（比如进入Spider的Responses;和从Spider出去的Requests）

三、整体架构图

Scrapy架构图(绿线是数据流向)



Scrapy运行流程大概如下：

1.引擎从调度器中取出一个链接(URL)用于接下来的抓取

2.引擎把URL封装成一个请求(Request)传给下载器

3.下载器把资源下载下来，并封装成应答包(Response)

4.爬虫解析Response

5.解析出实体（Item）,则交给实体管道进行进一步的处理

6.解析出的是链接（URL）,则把URL交给调度器等待抓取

制作 Scrapy 爬虫 一共需要4步：

1.新建项目 (scrapy startproject xxx)：新建一个新的爬虫项目

2.明确目标 （编写items.py）：明确你想要抓取的目标

3.制作爬虫 （spiders/xxspider.py）：制作爬虫开始爬取网页

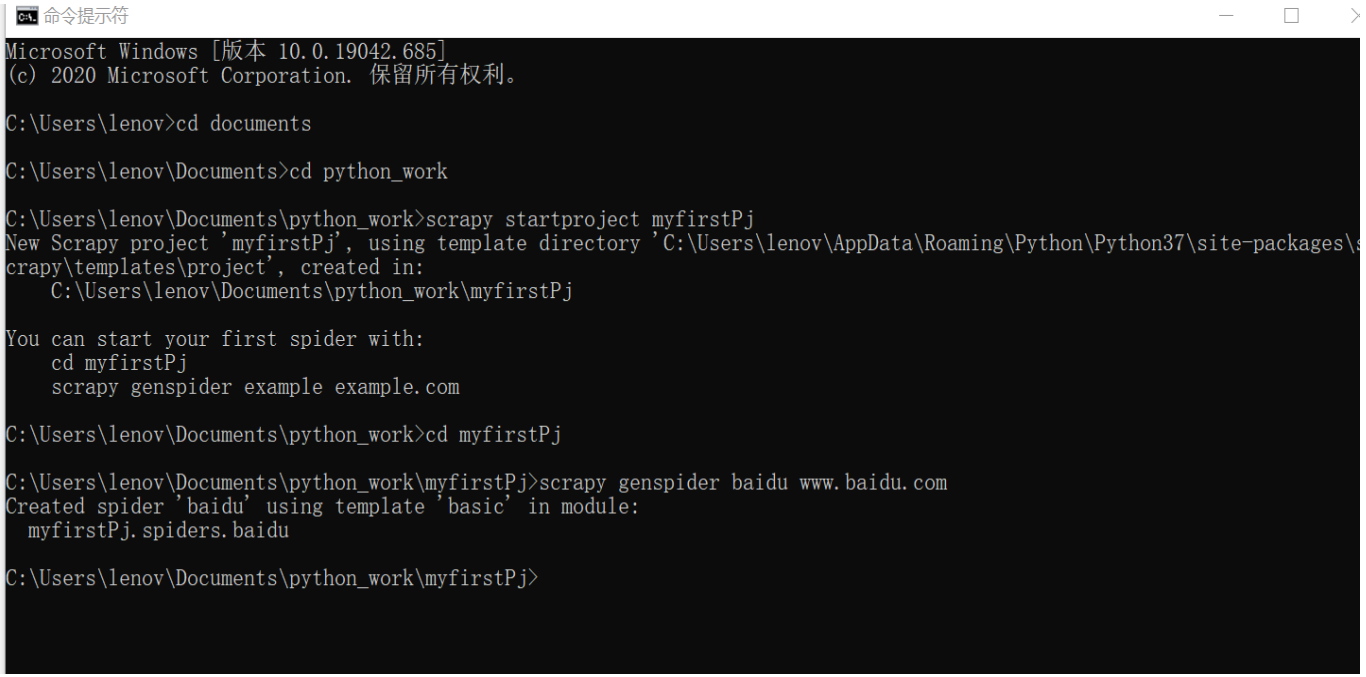
4.存储内容 （pipelines.py）：设计管道存储爬取内容

四、Scrapy安装以及生成项目

ubuntu，打开一个终端，输入pip install scrapy(或pip3 install scrapy）
widows，打开一个cmd，输入pip install scrapy，前提是你装了pip

将会用到以下命令：
scrapy startproject 项目名　　（创建新项目）
scrapy genspider 爬虫名 域名　　（创建爬虫程序）
scrapy crawl 爬虫名　　（运行爬虫程序）

新建项目(scrapy startproject)：
进入自定义的项目目录中，运行下列命令：
scrapy startproject myfirstPj
cd my firstPj
scrapy genspider baidu www.baidu.com



四、Scrapy安装以及生成项目

创建后目录大致页如下

```
|-ProjectName            #项目文件夹
  |-ProjectName          #项目目录
    |-items.py           #定义数据结构
    |-middlewares.py     #中间件
    |-pipelines.py       #数据处理
    |-settings.py        #全局配置
    |-spiders
      |-__init__.py      #爬虫文件
      |-baidu.py
  |-scrapy.cfg           #项目基本配置文件
```

spiders下的baidu.py是scrapy自动为我们生成的：

meiju.py | items.py | settings.py | pipelines.py | baidu.py

```python
import scrapy


class BaiduSpider(scrapy.Spider):
    name = 'baidu'
    allowed_domains = ['www.baidu.com']
    start_urls = ['http://www.baidu.com/']

    def parse(self, response):
        pass

```

四、Scrapy安装以及生成项目

下面再看一下spdier项目的配置文件，打开文件settings.py

BOT_NAME：项目名

USER_AGENT：默认是注释的，这个东西非常重要，如果不写很容易被判断为电脑

ROBOTSTXT_OBEY：是否遵循机器人协议，默认是true，需要改为false，否则很多东西爬不了

CONCURRENT_REQUESTS：最大并发数，很好理解，就是同时允许开启多少个爬虫线程

DOWNLOAD_DELAY：下载延迟时间，单位是秒，控制爬虫爬取的频率，根据你的项目调整，不要太快也不要太慢，默认是3秒，即爬一个停3秒，设置为1秒性价比较高，如果要爬取的文件较多，写零点几秒也行

COOKIES_ENABLED：是否保存COOKIES，默认关闭，开机可以记录爬取过程中的COKIE，非常好用的一个参数

DEFAULT_REQUEST_HEADERS：默认请求头，上面写了一个USER_AGENT，其实这个东西就是放在请求头里面的，这个东西可以根据你爬取的内容做相应设置。

ITEM_PIPELINES：项目管道，300为优先级，越低越爬取的优先度越高

u.py ✖ items.py ✖ settings.py ✖ pipelines.py ✖ baidu.py ✖ settings.py ✖ pipelines.py ✖

```
# Scrapy settings for myfirstPj project
#
# For simplicity, this file contains only settings considered important or
# commonly used. You can find more settings consulting the documentation:
#
#     https://docs.scrapy.org/en/latest/topics/settings.html
#     https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
#     https://docs.scrapy.org/en/latest/topics/spider-middleware.html

BOT_NAME = 'myfirstPj'

SPIDER_MODULES = ['myfirstPj.spiders']
NEWSPIDER_MODULE = 'myfirstPj.spiders'


# Crawl responsibly by identifying yourself (and your website) on the user-agent
#USER_AGENT = 'myfirstPj (+http://www.yourdomain.com)'

# Obey robots.txt rules
ROBOTSTXT_OBEY =False
```

四、Scrapy安装以及生成项目

到这里我们尝试用scrapy做一下爬取，打开spider.py下的baidu.py(取决于你scrapy genspider 爬虫名 域名时输入的爬虫名）输入一下代码，我们使用xpath提取百度首页的标题title：

保存 全部保存 还原 大闭 后退 前进 编译 生成 执行 颜色选择器

meiju.py × items.py × settings.py × pipelines.py × baidu.py × settings.py × pipelines.py × run.py × settings.p

```
1  import scrapy
2
3
4  class BaiduSpider(scrapy.Spider):
5      name = 'baidu'
6      allowed_domains = ['www.baidu.com']
7      start_urls = ['http://www.baidu.com/']
8
9      def parse(self, response):
10         tile=response.xpath('//html/head/title/text()')
11         print(tile)
12
```

打开一个终端cmd，输入scrapy crawl baidu(爬虫名），回车就可以看到一大堆输出信息，而其中就包括我们要的内容：

```
C:\Users\lenov\Documents\python_work\myfirstPj>scrapy crawl baidu
2021-10-07 16:04:20 [scrapy.utils.log] INFO: Scrapy 2.5.1 started (bot
```

```
)
2021-10-09 21:06:18 [scrapy.extensions.telnet] INFO: Telnet console listening on 127.0.0.1:6023
2021-10-09 21:06:18 [scrapy.core.engine] DEBUG: Crawled (200) <GET http://www.baidu.com/> (referer: None)
[<Selector xpath='//html/head/title/text()' data='百度一下，你就知道'>]
2021-10-09 21:06:18 [scrapy.core.engine] INFO: Closing spider (finished)
2021-10-09 21:06:18 [scrapy.statscollectors] INFO: Dumping Scrapy stats:
{'downloader/request_bytes': 213,
 'downloader/request_count': 1,
```

四、Scrapy安装以及生成项目：Xpath

其中用到了xpath，让我们来了解下xpath：
XPath，全称 XML Path Language，即 XML 路径语言，它是一门在 XML 文档中查找信息的语言。最初是用来搜寻 XML 文档的，但同样适用于 HTML 文档的搜索。所以在做爬虫时完全可以使用 XPath 做相应的信息抽取。XPath 的选择功能十分强大，它提供了非常简洁明了的路径选择表达式。另外，它还提供了超过 100 个内建函数，用于字符串、数值、时间的匹配以及节点、序列的处理等，几乎所有想要定位的节点都可以用 XPath 来选择。

官方文档：https://www.w3.org/TR/xpath/

接下来我们将会从以下几个方面对Xpath进行介绍：
一、XPath 常用规则
二、在谷歌浏览器安装XPath插件
三、使用Xpath解析
四、Xpath匹配示例
五、实战：图书馆数据搜索

四、Scrapy安装以及生成项目： Xpath

一、XPath 常用规则：

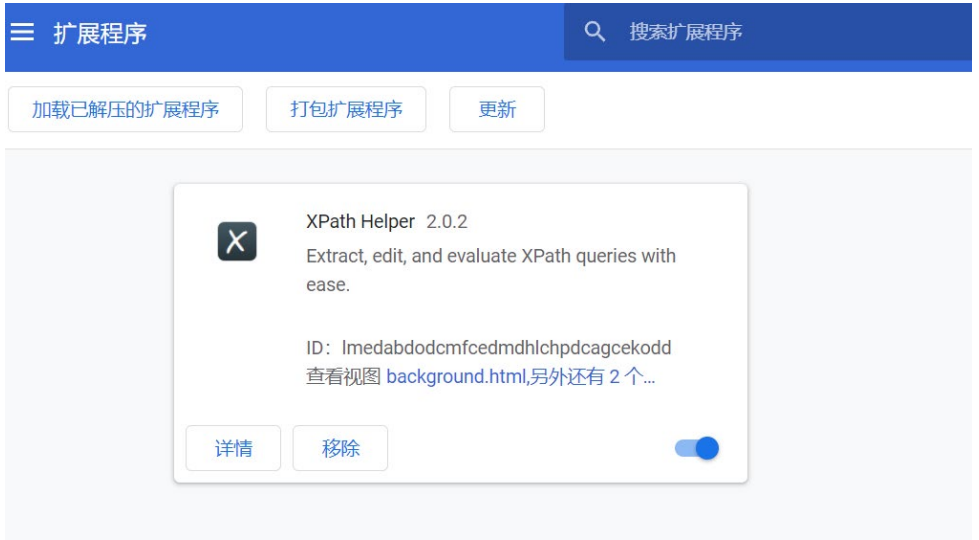
表达式	描述
nodename	选取此节点的所有子节点
/	从当前节点选取直接子节点
//	从当前节点选取子孙节点
.	选取当前节点
..	选取当前节点的父节点
@	选取属性

四、Scrapy安装以及生成项目： Xpath

二、在谷歌浏览器安装XPath插件：

1.下载xpath helper插件：（见附件压缩包）
解压压缩包，在google浏览器点击设置，更多工具，扩展程序：

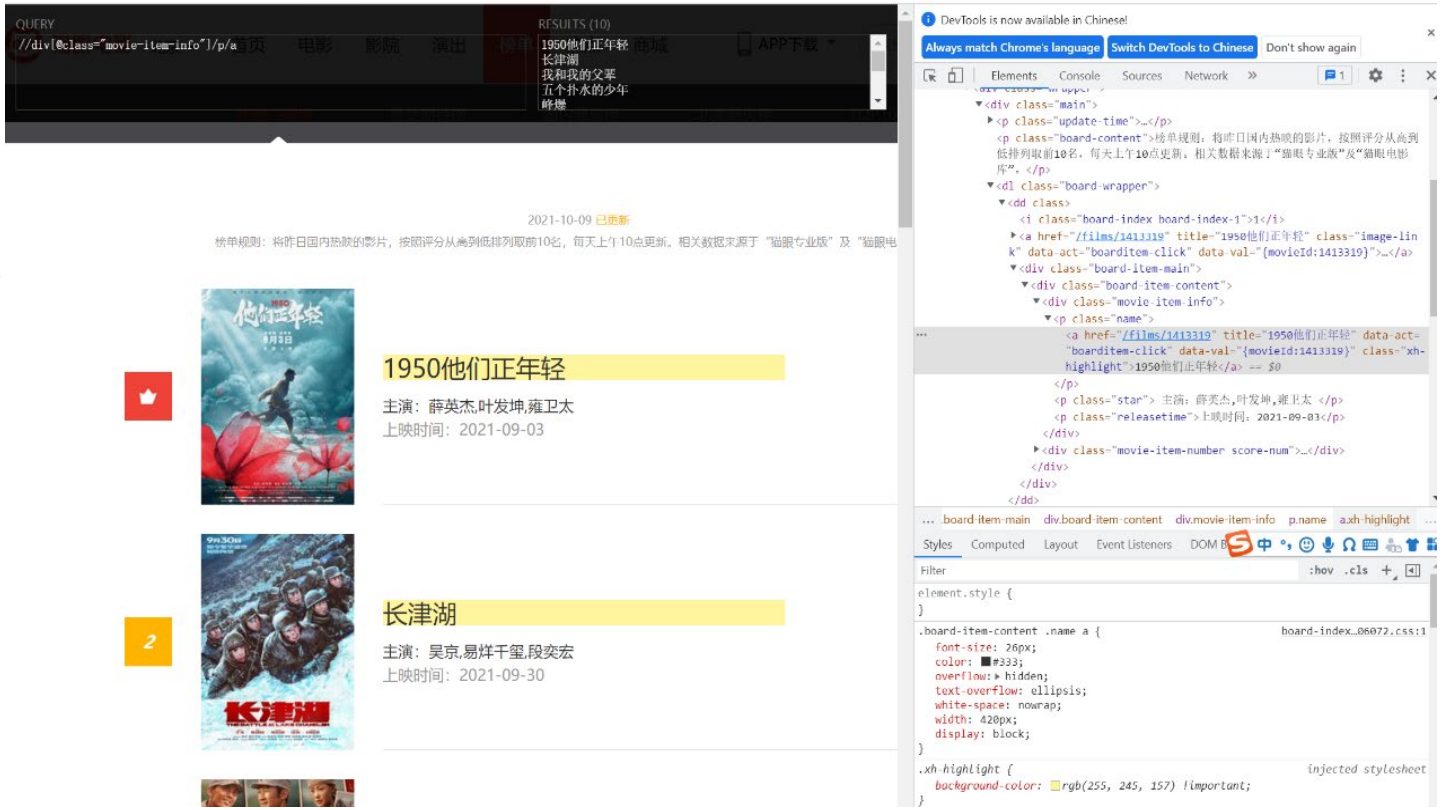
2.点击开发者模式，将刚才解压的文件中的.crx文件直接拖到这个界面，浏览器会自动安装。安装后如图：



四、Scrapy安装以及生成项目：Xpath

三、使用Xpath解析：

完成了前面的操作后，我们来看看Xpath的简单使用，我们拿一个网站来做测试，测试页面为猫眼电影网：https://maoyan.com/board
进入到页面，右键打开检查，我们观察到电影名是在<div class=""movie-item-info>下的p标签下的a链接的内容。



所以我们可以写出Xpath的表达式：
//div[@class="movie-item-info"]/p/a
我们打开Xpath匹配，
Query下输入//div[@class="movie-item-info"]/p/a。
Result里面输出结果。

以上是Xpath的第一次尝试，下面来讲Xpath的几个经典示例。

四、Scrapy安装以及生成项目：Xpath

四、Xpath匹配示例：
由于示例过程较为冗长，在示例之前我们先捋清Xpath每一步的目的及讲解思路：

1.查看所有的标签（如p、a、li标签等）

2.查看某标签下的所有标签（如p下的a标签）

3.带属性值的匹配

4.查看某标签下的第n个标签

5.输出某标签的属性值

6.xpath常用函数

7.总结

四、Scrapy安装以及生成项目：Xpath

1.查看所有的标签（如p、a、li标签等）

在一个Html页面中，如果要匹配所有的标签，可以输入：//标签名

打开猫眼电影：https://maoyan.com/board

以p标签为例，//p将会匹配所有的p标签下的内容。//代表从当前节点选取子孙节点，而当前结点就是根节点，所以//p将会匹配根节点下所有p节点。读者可以尝试其他标签。

2.查看某标签下的所有标签（如p下的a标签）

根据分析，电影名都在p标签下的a标签里，所以可以通过//p/a来匹配。我们已经知道了//p的含义，而再加一个/a代表在//p的结果下再找筛选a标签的内容 xpath表达式为：//p//a

四、Scrapy安装以及生成项目：Xpath

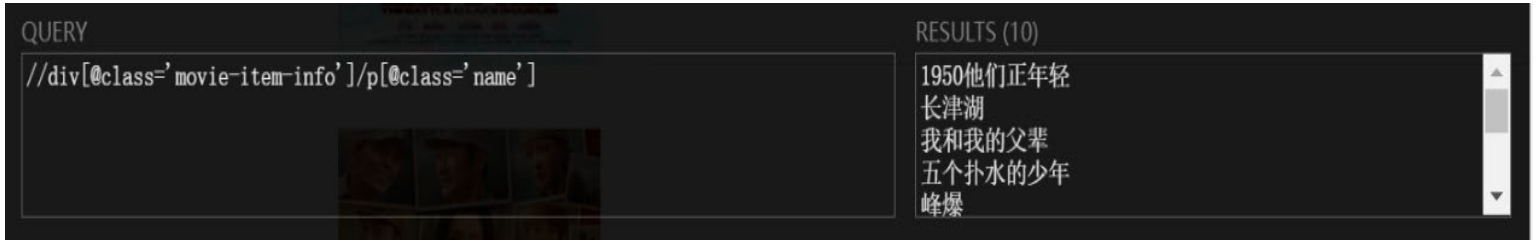
3.带属性值的匹配

如果我们想匹配特定的一个内容，我们可以假如属性值。属性值的格式为：标签名[@属性=“属性值”]，如图所示。

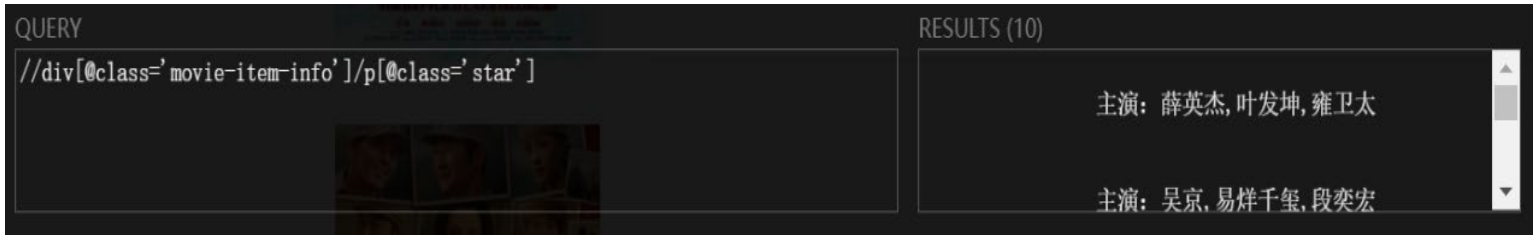
如果我们想匹配title=“我和我的祖国”的电影名，xpath格式为：//p/a[@title="我和我的祖国"]

以上的方式太局限了，经过分析，我们发下p标签上面一级是 。而div的下的p标签的class决定了输出什么类容。

如<p class=“name”> 输出电影名：



<p class=“start”>输出演员名单：



四、Scrapy安装以及生成项目：Xpath

4.查看某标签下的第n个标签
我们也可以通过[n]来决定要输出第几个标签的内容，不加[n]将输出所有内容。

p[1]输出电影名：

p[2]输出演员表：

四、Scrapy安装以及生成项目：Xpath

5.输出某标签的属性值
如果我们想拿到a标签下的href属性值，按照常规思路可以写出xpath：//div[@class="movie-item-info"]/p/a
但是这样只会拿到a标签下的内容，而不会拿到属性值。正确写法是为：//div[@class="movie-item-info"]/p/a/@href

在a标签后在@属性名：

6.xpath常用函数
6.1 contains（）：匹配属性值中包含某些字符串的节点
如下面的例子，这里的id并不一样，那么我们获取的方式可以通过：//li[contains(@id,"car_")]
<li id="car_bw" >宝马
<li id="car_byd" >比亚迪
6.2 text（）：获取标签里的内容，作为字符串输出

四、Scrapy安装以及生成项目：Xpath

7.总结
//代表从根节点向下找
/代表从当前结点往下找
比如//p匹配到根下的所有p标签，//p/a在从p标签下找a标签
@的使用场景：
1）属性值作为条件
　　//p/a[@title="我和我的祖国"]
2）直接获取属性值
　　//div[@class="movie-item-info"]/p/a/@href
获取文本内容需要加text()
比如//p/a，虽然会输出a标签下的文本内容，但是这个表达式是不严谨的，
如果是想抓取a下的内容，最好写成：//p/a/text()
element还是string:
不加text()的情况或者不以@属性名结尾的情况下，返回的结果都是element，
element是元素节点，如果你在python想将抓取的结果作为String输出，那么加上text()或@属性名

五、实践：图书馆数据爬虫

Scrapy是常用的Python爬虫框架之一，它对数据爬取与数据存储进行了分工，实现了数据的分布式爬取与存储，是一个功能强大的专业爬取框架。

本项目以爬取当当网的图书数据为例，讲解Scrapy中Request请求、异步回调函数、Response响应、Item数据项目、Pipeline数据存储等方面的知识，使用Scrapy设计爬虫程序，快速地爬取几千本图书的数据。

五、实践：图书馆数据爬虫

知识目标

(1) 掌握scrapy分布式数据爬取的基本原理；

(2) 掌握scrapy的Request请求、 Response响应， 回调函数

(3) 掌握Item数据字段的原理；

(4) 掌握Pipeline数据管道存储方法；

(5) 掌握XPath数据查找方法

五、实践：图书馆数据爬虫

能力目标

(1) 能使用scrapy编写分布式爬虫程序

(2) 能使用Request请求获取Response响应，能使用回调函数

(3) 能使用Item类定义数据并使用Pipeline管道存储数据

(4) 能使用XPath查找与解析数据。

五、实践：图书馆数据爬虫

实战视频（45’）

https://www.icourse163.org/learn/ZIIT-1002925008?tid=1468345444#/learn/content?type=detail&id=1251430717&sm=1

网络爬虫与法理法规

爬虫作为一种爬取网络数据的技术，同学们要正确使用网络爬虫技术。网络不是法外之地，爬虫不是牟利工具。同学们要遵守法律法规，合法有序地爬取数据，正确合理地使用爬取到的数据，不能危及网络安全，不能侵犯他人的知识产权。

网络无边际，安全有界限，依法治网、依法办网、依法上网，让互联网在法治轨道上健康运行是维护网络安全、净化网络生态的重要环节。我国为网络安全与知识产权建立了一系列的法律法规，如《中华人民共和国民法典》等。

总结

- scrapy爬虫
- scrapy五大基本构成（引擎 调度 下载 爬虫 实体管道）
- scrapy整体架构（项目 目标 制作 存储）
- scrapy安装以及生成项目（Xpath）
- 案例分析