day07-课堂笔记

回顾

想要使用Scrapy框架去完成代码编写

1. 创建项目

```
Terminal:  Local  +
(spider_py3) E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码>scrapy startproject school
New Scrapy project 'school', using template directory 'e:\py_env\spider_py3\lib\site-packages\scrapy\templates\project', created in:
    E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码\school

You can start your first spider with:
    cd school
    scrapy genspider example example.com

(spider_py3) E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码>^Q^Q
```

2. 进入到项目的目录下，创建爬虫

```
(spider_py3) E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码>cd school

(spider_py3) E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码\school>scrapy genspider sgy www.sxpi.edu.cn
Created spider 'sgy' using template 'basic' in module:
  school.spiders.sgy

(spider_py3) E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码\school>
```

3. 创建爬虫完毕之后，在爬虫文件中书写爬虫代码

```python
import scrapy


class SgySpider(scrapy.Spider):
    name = 'sgy'
    allowed_domains = ['www.sxpi.edu.cn']
    # 一般情况下，要抓取哪一个页面，就将哪一个页面的url地址复制粘贴到此处即可。
    # 起始页的url地址 是一个列表，可以设置多个起始页的url地址。
    start_urls = ['https://www.sxpi.edu.cn/xwzx/gyyw.htm']

    # 专门用于去处理起始页url地址的响应的， 参数中的response就是起始页的url地址的响应对象
    def parse(self, response):
        # 提取数据
        # 获取所有的包含数据的li标签的分组（列表）
        li_list = response.xpath("//div[@class='list_box']/ul/li")
        # 遍历数据的列表，获取每一条数据
        for li in li_list:
            item = {}
            # 获取新闻的标题  直接使用xpath返回的是一个 selecter选择器对象，如果响应获取选择器对象中的文本内容
            # 可以使用  extract_first()  文本字符串内容，  使用 extract() 获取到的是 列表，列表中的元素是 文本字符串内容
            item["title"] = li.xpath("./a/text()").extract_first()
            # 发布时间
            item["publish_time"] = li.xpath("./span/text()").extract_first()
            print(item)
```

4. 运行爬虫

```
(spider_py3) E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码\school>scrapy crawl sgy
2021-09-07 12:38:30 [scrapy.utils.log] INFO: Scrapy 2.5.0 started (bot: school)
2021-09-07 12:38:30 [scrapy.utils.log] INFO: Versions: lxml 4.6.3.0, libxml2 2.9.5, cssselect 1.1.0, parsel 1.6.0, w3lib 1.22.0, Twisted 21.7.0, Python 3.9.2 (tags/v3
1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)], pyOpenSSL 20.0.1 (OpenSSL 1.1.1l  24 Aug 2021), cryptography 3.4.8, Platform Windows-10-10.0.19041-SP0
2021-09-07 12:38:30 [scrapy.utils.log] DEBUG: Using reactor: twisted.internet.selectreactor.SelectReactor
2021-09-07 12:38:30 [scrapy.crawler] INFO: Overridden settings:
{'BOT_NAME': 'school',
 'NEWSPIDER_MODULE': 'school.spiders',
 'ROBOTSTXT_OBEY': True,
```

1. Scrapy 管道保存数据

1.1 管道的基本使用

1. 在爬虫中将获取到的数据使用 `yield` 关键字进行返回给引擎

```python
class SgySpider(scrapy.Spider):
    name = 'sgy'
    allowed_domains = ['www.sxpi.edu.cn']
    # 一般情况下，要抓取哪一个页面，就将哪一个页面的url地址复制粘贴到此处即可。
    # 起始页的url地址 是一个列表，可以设置多个起始页的url地址。
    start_urls = ['https://www.sxpi.edu.cn/xwzx/gyyw.htm']

    # 专门用于去处理起始页url地址的响应的， 参数中的response就是起始页的url地址的响应对象
    def parse(self, response):
        # 提取数据
        # 获取所有的包含数据的li标签的分组（列表）
        li_list = response.xpath("//div[@class='list_box']/ul/li")
        # 遍历数据的列表，获取每一条数据
        for li in li_list:
            item = {}
            # 获取新闻的标题  直接使用xpath返回的是一个 selecter选择器对象，如果响应获取选择器对象中的文本内容
            # 可以使用  extract_first()  文本字符串内容， 使用 extract() 获取到的是 列表，列表中的元素是 文本字符串内容
            item["title"] = li.xpath("./a/text()").extract_first()
            # 发布时间
            item["publish_time"] = li.xpath("./span/text()").extract_first()
            # 在Scrapy框架中 爬虫中获取到的数据 要返回给引擎，那么要使用  yield 关键字来返回数据
            yield item
```

2. 打开pipelines.py文件在改文件中书写保存数据的代码

```python
from pymongo import MongoClient

# 创建客户端连接对象
client = MongoClient()
# 创建要操作的集合对象
collections = client.school.news


class SchoolPipeline:

    def process_item(self, item, spider):
        """
        此方法就是专门用于去处理数据的方法。而且这个方法在改管道类中必须有
        并且名字必须叫做 process_item
        :param item: 要处理的数据 （对象、字典）
        :param spider: 当前运行的爬虫是哪一个爬虫，那么这个spider就是那个爬虫对象
        :return: 在此方法中必须要有return  的内容，就是我们处理完的数据。
        """
        # 保存到MongoDB数据库中
        print('管道代码执行了.............')
        collections.insert_one(item)
        return item
```

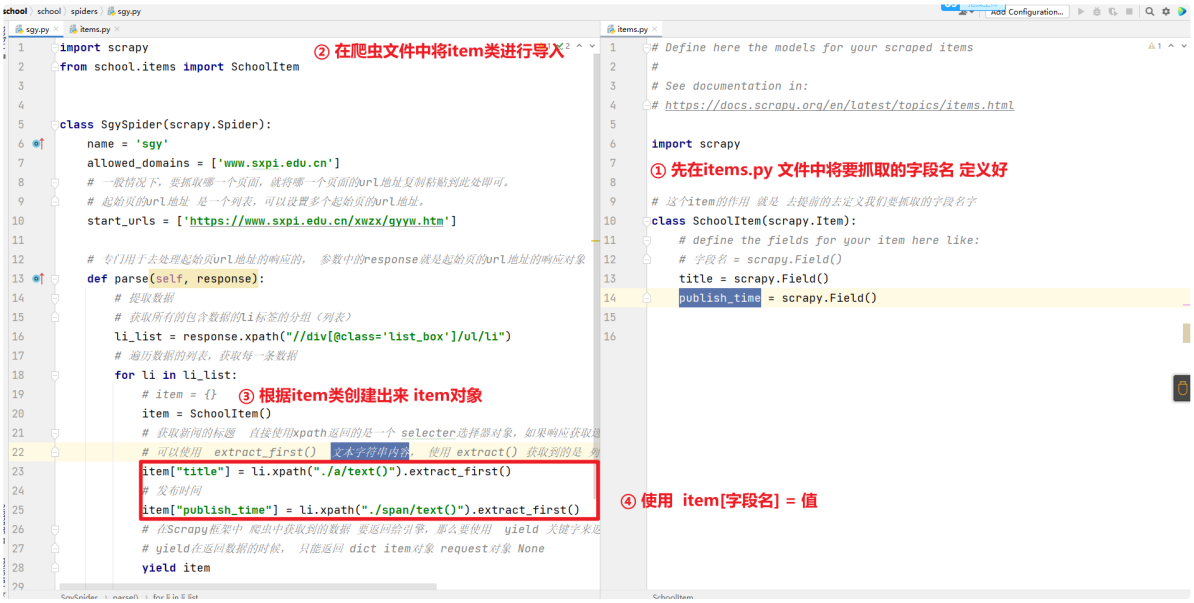
3. 必须要在配置文件settings.py中将管道进行开启，如果不开启管道，那么管道代码是不会执行的。

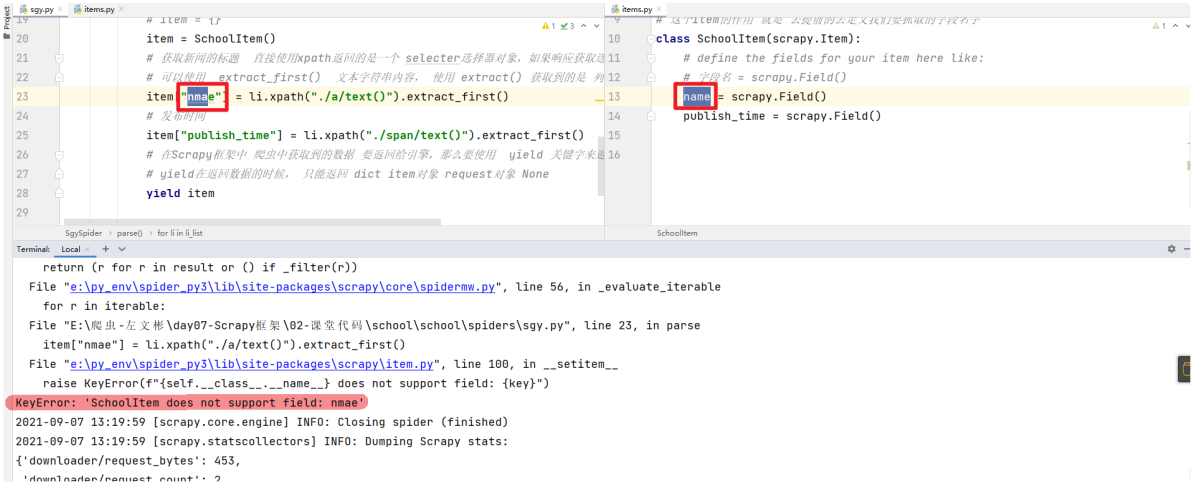
```python
#}


# Configure item pipelines
# 这个配置项 就是专门用于去配置管道的配置项
ITEM_PIPELINES = {
    #   管道类的路径                        1-1000 权重，数字越小表示权重越大，数据会越先经过这个管道。
    'school.pipelines.SchoolPipeline': 300,
}
```

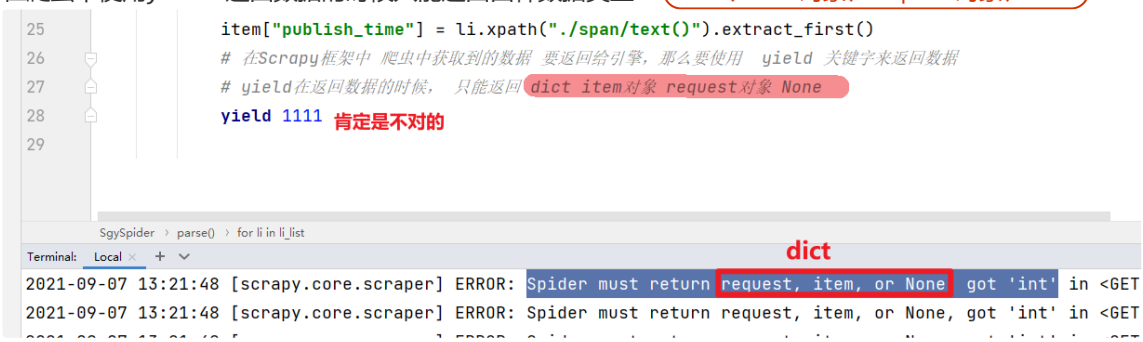
2.1 使用管道的时候注意点

item对象





1. 在爬虫中使用yield 返回数据的时候只能返回四种数据类型：dict、item对象、request对象、None



2. 在使用管道的时候，必须要在配置文件中去开启对应的管道，如果不开启，写的管道代码是不会生效的（不会执行。）

```python
#}

# Enable or disable extensions
# See https://docs.scrapy.org/en/latest/topics/extensions.html
#EXTENSIONS = {
#    'scrapy.extensions.telnet.TelnetConsole': None,
#}

# Configure item pipelines
# 这个配置项 就是专门用于去配置管道的配置项
ITEM_PIPELINES = {
   #  管道类的路径                        1-1000 权重，数字越小表示权重越大，数据会越先经过
   'school.pipelines.SchoolPipeline': 300,
}

# Enable and configure the AutoThrottle extension (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/autothrottle.html
#AUTOTHROTTLE_ENABLED = True
# The initial download delay
#AUTOTHROTTLE_START_DELAY = 5
# The maximum download delay to be set in case of high latencies
#AUTOTHROTTLE_MAX_DELAY = 60
# The average number of requests Scrapy should be sending in parallel to
# each remote server
#AUTOTHROTTLE_TARGET_CONCURRENCY = 1.0
```

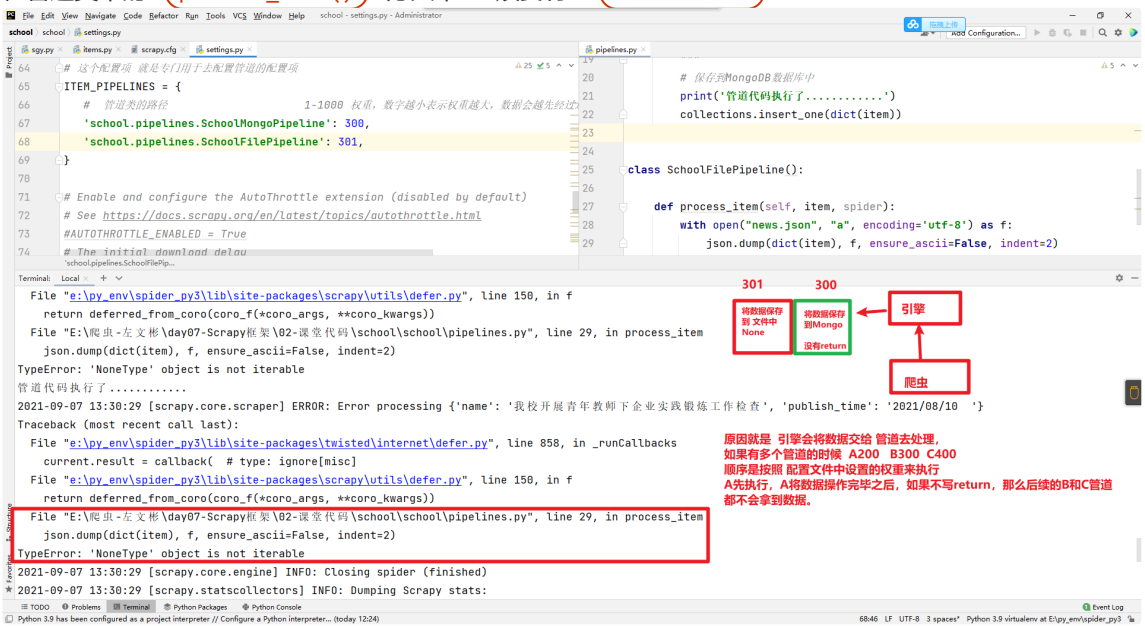
```python
from pymongo import MongoClient

# 创建客户端连接对象
client = MongoClient()
# 创建要操作的集合对象
collections = client.school.news


class SchoolPipeline:

    def process_item(self, item, spider):
        """
        此方法就是专门用于去处理数据的方法，而且这个方法在改管道类中必须有
        并且名字必须叫做 process_item
        :param item: 要处理的数据 （对象、字典）
        :param spider: 当前运行的爬虫是哪一个爬虫，那么这个spider就是那个爬虫对象
        :return: 在此方法中必须要有return 的内容，就是我们处理完的数据。
        """
        # 保存到MongoDB数据库中
        print('管道代码执行了.............')
        collections.insert_one(item)
        return item
```

3. 在管道类中的 `process_item()` 方法中 必须要有 `return item`



```python
# 这个配置项 就是专门用于去配置管道的配置项
ITEM_PIPELINES = {
   #  管道类的路径                        1-1000 权重，数字越小表示权重越大，数据会越先经过
   'school.pipelines.SchoolMongoPipeline': 300,
   'school.pipelines.SchoolFilePipeline': 301,
}

# Enable and configure the AutoThrottle extension (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/autothrottle.html
#AUTOTHROTTLE_ENABLED = True
# The initial download delay
```

```python
        # 保存到MongoDB数据库中
        print('管道代码执行了.............')
        collections.insert_one(dict(item))


class SchoolFilePipeline():

    def process_item(self, item, spider):
        with open("news.json", "a", encoding='utf-8') as f:
            json.dump(dict(item), f, ensure_ascii=False, indent=2)
```

```
  File "e:\py_env\spider_py3\lib\site-packages\scrapy\utils\defer.py", line 150, in f
    return deferred_from_coro(coro_f(*coro_args, **coro_kwargs))
  File "E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码\school\school\pipelines.py", line 29, in process_item
    json.dump(dict(item), f, ensure_ascii=False, indent=2)
TypeError: 'NoneType' object is not iterable
管道代码执行了.............
2021-09-07 13:30:29 [scrapy.core.scraper] ERROR: Error processing {'name': '我校开展青年教师下企业实践锻炼工作检查', 'publish_time': '2021/08/10  '}
Traceback (most recent call last):
  File "e:\py_env\spider_py3\lib\site-packages\twisted\internet\defer.py", line 858, in _runCallbacks
    current.result = callback(  # type: ignore[misc]
  File "e:\py_env\spider_py3\lib\site-packages\scrapy\utils\defer.py", line 150, in f
    return deferred_from_coro(coro_f(*coro_args, **coro_kwargs))
  File "E:\爬虫-左文彬\day07-Scrapy框架\02-课堂代码\school\school\pipelines.py", line 29, in process_item
    json.dump(dict(item), f, ensure_ascii=False, indent=2)
TypeError: 'NoneType' object is not iterable
2021-09-07 13:30:29 [scrapy.core.engine] INFO: Closing spider (finished)
2021-09-07 13:30:29 [scrapy.statscollectors] INFO: Dumping Scrapy stats:
```


原因就是 引擎会将数据交给 管道去处理，
如果有多个管道的时候 A200 B300 C400
顺序是按照 配置文件中设置的权重来执行
A先执行，A将数据操作完毕之后，如果不写return，那么后续的B和C管道都不会拿到数据。

2. Scrapy 构造新的 request对象