爬虫案例2(下载文件和图片)

利用scrapy提供的2个数据流处理管道

1.FilesPipline 2.ImagesPipline

一.下载matplotlib例子进行文件下载的学习

爬取网址：https://matplotlib.org/examples/index.html#

详情页面：https://matplotlib.org/examples/api/agg_oo.html

步骤1.利用scrapy shell 分析页面

注意然后可以 fetch 重新请求页面 在分析详情页面

思路：分析列表页面,分析详情页面

步骤2:思路实现

创建项目 —— 配置文件处理管道 ——— 指定下载目录 ——— 构建实体类 —— 构建逻辑

1.创建项目

```
MichaelYun:PycharmProjects Yun$ scrapy startproject matplotlib
New Scrapy project 'matplotlib', using template directory '/usr/local/lib/python
3.7/site-packages/scrapy/templates/project', created in:
    /Users/Yun/PycharmProjects/matplotlib

You can start your first spider with:
    cd matplotlib
    scrapy genspider example example.com
MichaelYun:PycharmProjects Yun$ cd matplotlib/
MichaelYun:matplotlib Yun$ scrapy genspider example matplotlib.org
Created spider 'example' using template 'basic' in module:
  matplotlib.spiders.example
MichaelYun:matplotlib Yun$
```

2.配置文件的管理的处理 并 指定下载目录

settings.py

```
#DOWNLOADER_MIDDLEWARES = {
#    'matplotlib.middlewares.MatplotlibDownloaderMiddleware': 543,
#}

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

# Configure item pipelines
# See https://doc.scrapy.org/en/latest/topics/item-pipeline.html
ITEM_PIPELINES = {
    'scrapy.pipelines.files.FilesPipeline': 1,
}
FILES_STORE = 'examples_src'
```

3.配置实体类

```
import scrapy


class MatplotlibItem(scrapy.Item):
    file_urls = scrapy.Field()
    files = scrapy.Field()
```

4.实现逻辑代码

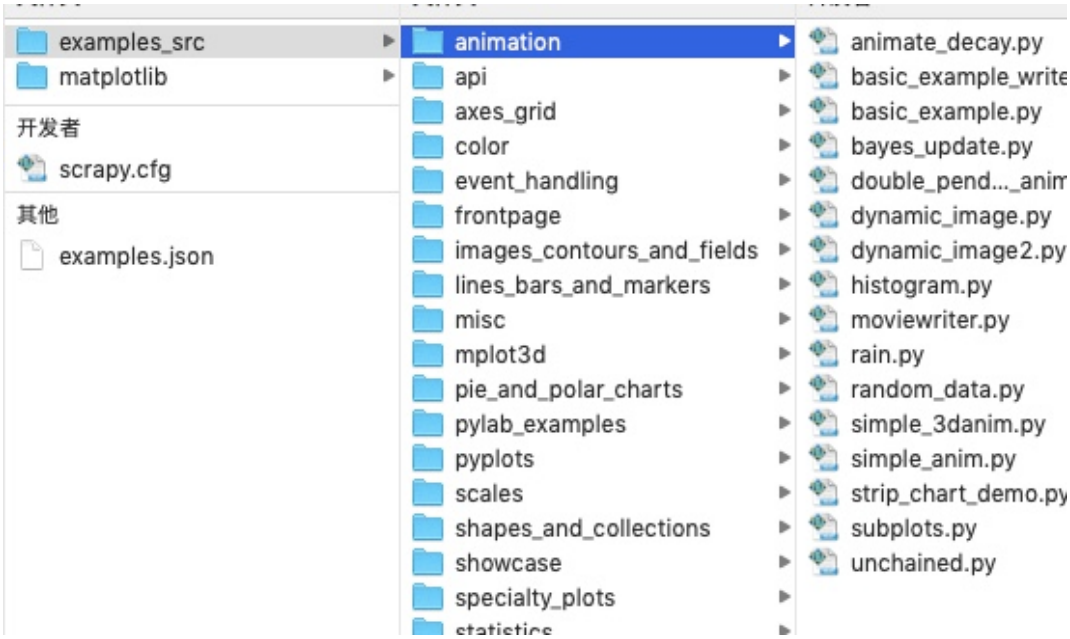
```
# -*- coding: utf-8 -*-
import scrapy
from scrapy.linkextractors import  LinkExtractor
from ..items import  MatplotlibItem


class ExampleSpider(scrapy.Spider):
    name = 'example'
    allowed_domains = ['matplotlib.org']
    start_urls = ['https://matplotlib.org/examples/index.html']

    def parse(self, response):
        le  = LinkExtractor(restrict_css='div.toctree-wrapper.compound',deny='/index.html$') #deny 正则表达式 排除这个地址不提取
        for link in le.extract_links(response):
            yield scrapy.Request(link.url,callback=self.parse_example)

    def parse_example(self, response):
        href = response.css('a.reference.external::attr(href)').extract_first()
        url =response.urljoin(href)
        example = MatplotlibItem()
        example['file_urls'] = [url]    # file_urls 必须是一个集合
        return example
```

额外的:

改进文件额外的下载

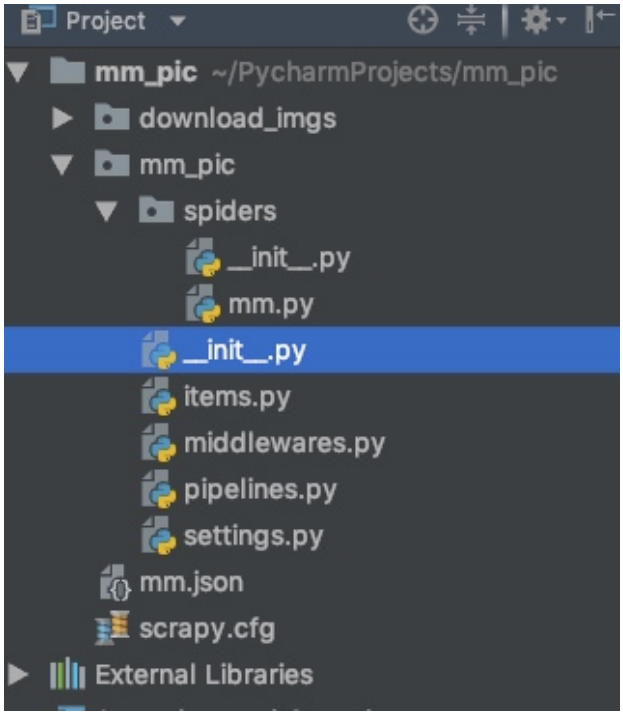
```python
from scrapy.pipelines.files import FilesPipeline
from urllib.parse import urlparse
from os.path import basename,dirname,join

class MatplotlibPipeline(FilesPipeline):
    def file_path(self, request, response=None, info=None):
        path = urlparse(request.url).path
        print(path)
        return join(basename(dirname(path)), basename(path))
```

```python
ITEM_PIPELINES = {
   # 'scrapy.pipelines.files.FilesPipeline': 1,
    'matplotlib.pipelines.MatplotlibPipeline':1,
}
```



————————分隔线(下载校花网图片)

步骤1.创建项目



步骤2 设置管道和位置

```
ITEM_PIPELINES = {
    'mm_pic.pipelines.MmPicPipeline': 1,
#    'scrapy.pipelines.images.ImagesPipeline':1
}

IMAGES_STORE = 'download_imgs'
```

步骤3 实现自定义管道

```python
# -*- coding: utf-8 -*-

# Define your item pipelines here
#
# Don't forget to add your pipeline to the ITEM_PIPELINES setting
# See: https://doc.scrapy.org/en/latest/topics/item-pipeline.html

from scrapy.pipelines.images import ImagesPipeline
from scrapy import Request
class MmPicPipeline(ImagesPipeline):
    def get_media_requests(self, item, info):
        # for image_url in item['image_urls']:
        #    yield Request(,meta={'mm_name':item['mm_name']})
        yield Request(item['image_urls'][0], meta={'mm': item['mm_name']})

    def file_path(self, request, response=None, info=None):
        file = (request.meta['mm'])[0:2]
        filename = request.meta['mm']
        MM_FileName = u'full/{0}/{1}.jpg'.format(file,filename)
        return MM_FileName

# 顺序从上至下
```

步骤4.实现逻辑 处理路径-提取路径-返回对应的实体

```python
# -*- coding: utf-8 -*-
import scrapy
import  re
from ..items import MmPicItem
from scrapy.linkextractors import LinkExtractor


class MmSpider(scrapy.Spider):
    name = 'mm'
    allowed_domains = ['www.xiaohuar.com']
    start_urls = ['http://www.xiaohuar.com/hua/']

    def parse(self, response):
        img_names = response.css('div.item_t img::attr(alt)').extract()
        img_urls =  response.css('div.item_t img::attr(src)').extract()
        #   处理不带地址的链接
        for index in range(len(img_urls)):
            if re.match('/d',img_urls[index]):
                img_urls[index] = response.urljoin(img_urls[index])
        for index in range(len(img_names)):
            mm = MmPicItem()
            mm['mm_name'] = img_names[index]
            mm['image_urls'] = [img_urls[index]]
            yield mm
        #   接着捕获下一页
        le = LinkExtractor(restrict_css='div.page_num a:nth-last-child(2)')
        links = le.extract_links(response)
        if links[0].text == '下一页':
            next_url = links[0].url
            yield scrapy.Request(next_url, callback=self.parse)


```

步骤5.执行操作

scrapy crawl mm -o mm.json