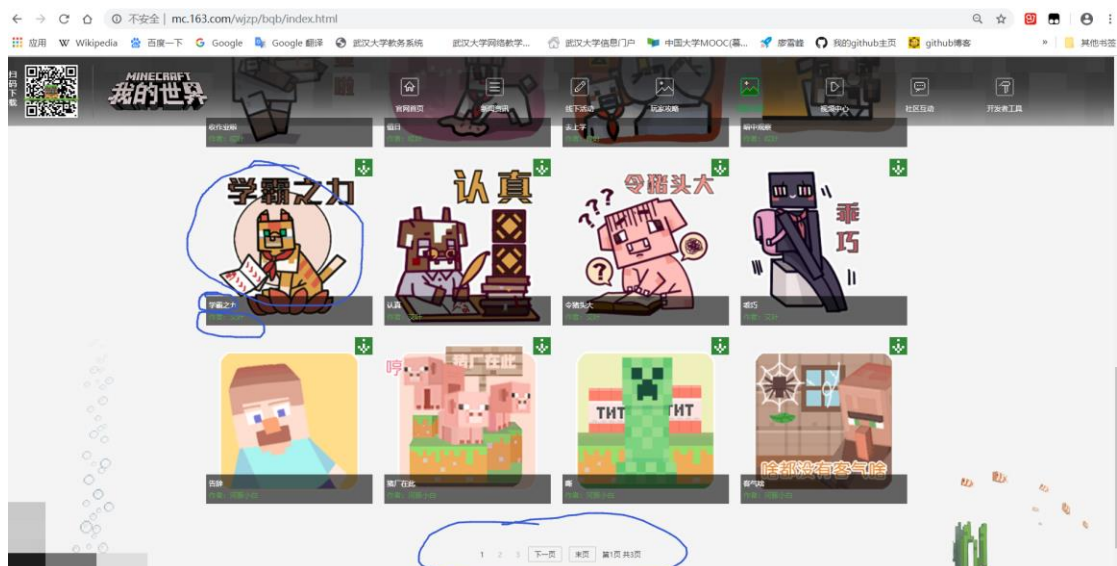


使用 Scrapy 爬取数据信息:

使用 Scrapy+xpath 爬取我的世界官网的表情包。网址为: <http://mc.163.com/wjzp/bqb/index.html>。爬取每一张图片和对应的图片名, 作者。

题目分析:

这里爬取该网站的表情包, 以及对应的名字和作者。



并实现翻页操作。

共 48 张图。

实验具体步骤:

1. 新建一个工程, 名为 myfirst

```
(base) C:\Users\12943\Desktop\信息内容安全\实验\实验1-爬虫>scrapy startproject myfirst
New Scrapy project 'myfirst', using template directory 'd:\anaconda\anaconda\lib\site-packages\scrapy\templates\project', created in:
C:\Users\12943\Desktop\信息内容安全\实验\实验1-爬虫\myfirst

You can start your first spider with:

cd myfirst
scrapy genspider example example.com

(base) C:\Users\12943\Desktop\信息内容安全\实验\实验1-爬虫>
```

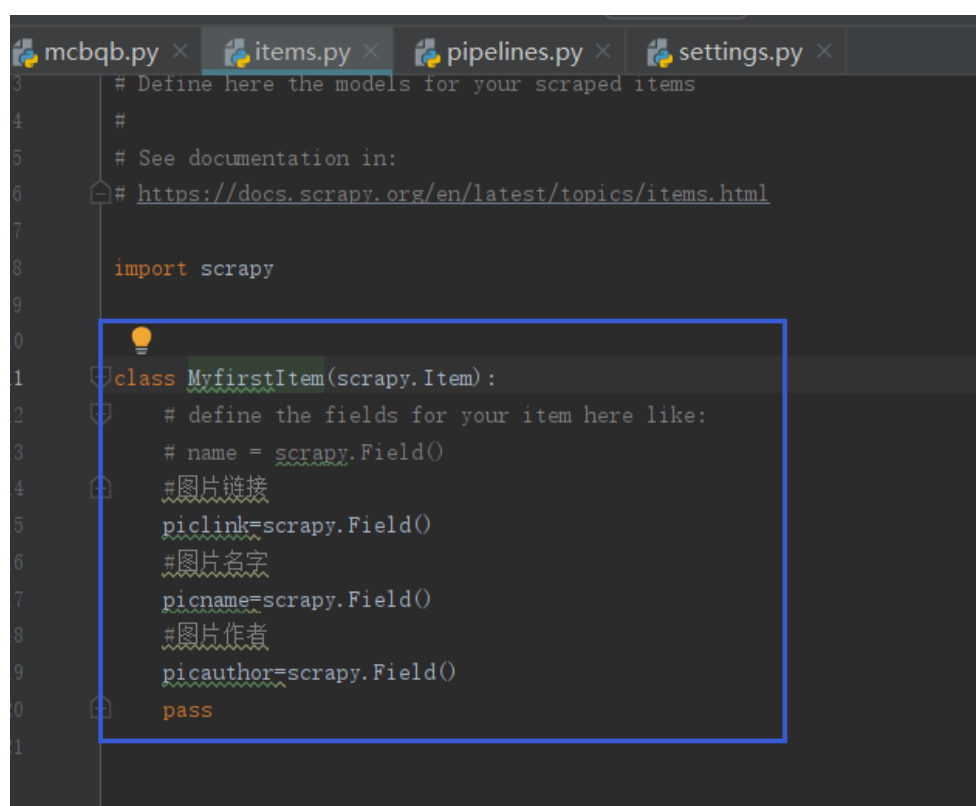
进入到该目录下，新建一个名为 mcbqb 的爬虫。

```
(base) C:\Users\12943\Desktop\信息内容安全\实验\实验1-爬虫>cd myfirst

(base) C:\Users\12943\Desktop\信息内容安全\实验\实验1-爬虫\myfirst>scrapy genspider mcbqb mc.163.com
Created spider 'mcbqb' using template 'basic' in module:
myfirst.spiders.mcbqb
```

2. 开始修改代码。

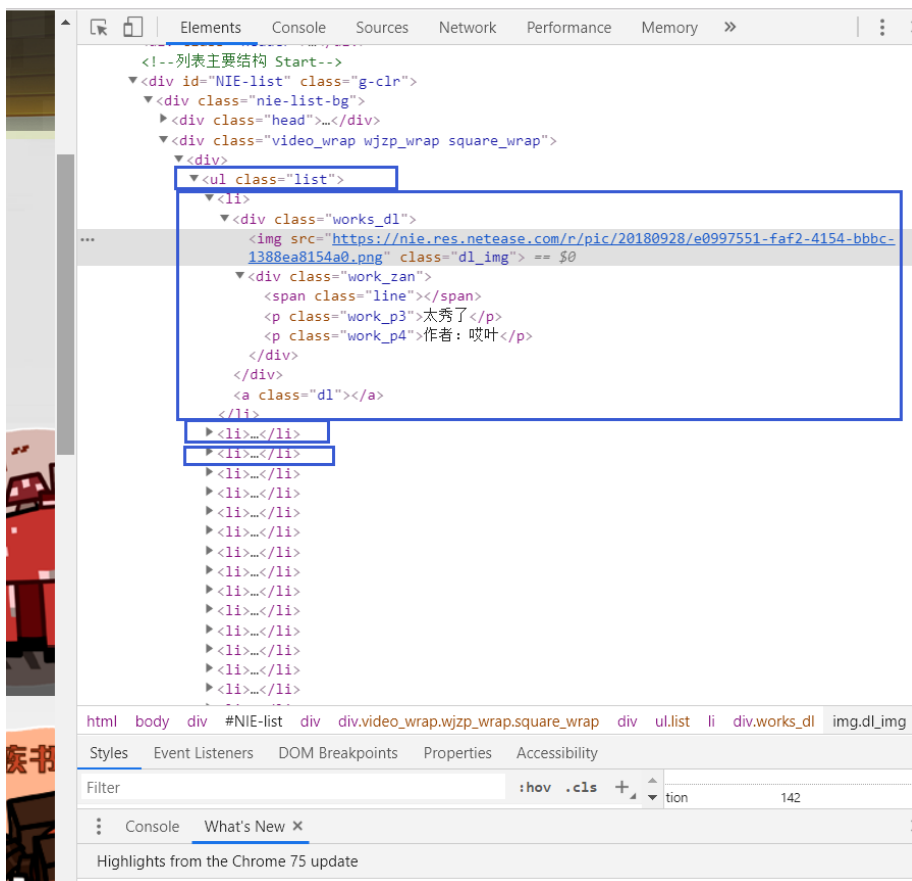
首先在 items.py 中定义所需要的链接地址，名字和作者。这个作用相当于谁使用这个类，谁就变为了一个字典，key 值就是我们定义的这些东西。



```
mcbqb.py x items.py x pipelines.py x settings.py x
3 # Define here the models for your scraped items
4 #
5 # See documentation in:
6 # https://docs.scrapy.org/en/latest/topics/items.html
7
8 import scrapy
9
10
11 class MyfirstItem(scrapy.Item):
12     # define the fields for your item here like:
13     # name = scrapy.Field()
14     # 图片链接
15     piclink=scrapy.Field()
16     # 图片名字
17     picname=scrapy.Field()
18     # 图片作者
19     picauthor=scrapy.Field()
20     pass
```

3. 之后在 mcbqb.py 中提取网页中所需的数据：

从源代码中可以定位到所需要信息的地址：



循环得到每一个标签:

```
def parse(self, response):
    item = MyfirstItem()
    pics = response.xpath('//ul[@class="list"]/li')
    for pic in pics:
        #提取信息
        item["piclink"] = pic.xpath('./div/img/@src').extract()[0]
        item["picname"] = pic.xpath('./div/div/p[@class="work_p3"]/text()').extract()[0]
        item["picauthor"] = pic.xpath('./div/div/p[@class="work_p4"]/text()').extract()[0]

    yield item
```

接下来进行翻页操作:

通过观察得知, 网页链接有规律, 可通过此获得每一个页面的信息。

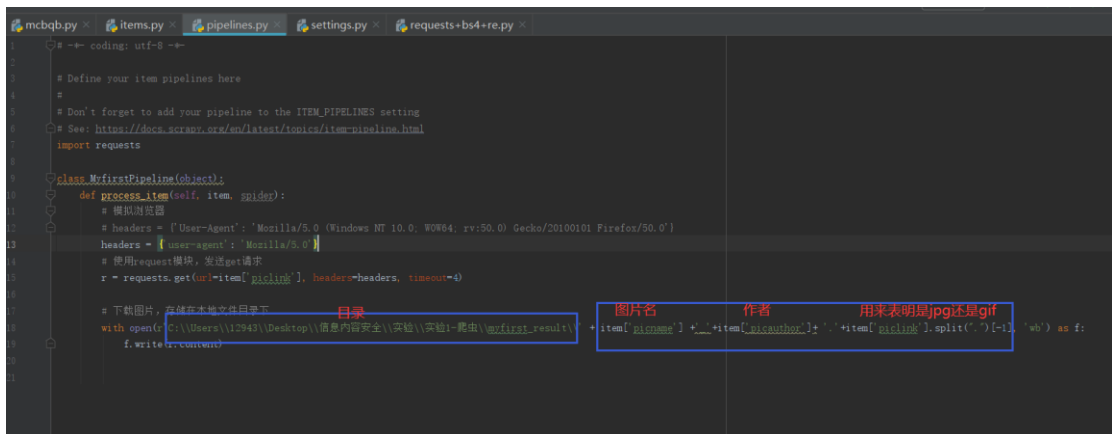
① 不安全 | mc.163.com/wjzp/bqb/index_2.html

① 不安全 | mc.163.com/wjzp/bqb/index_3.html

```
#实现翻页操作
for i in range(2, 4):
    url='http://mc.163.com/wjzp/bqb/index_'+str(i)+'.html'
    yield Request(url,callback=self.parse)
```

也可以根据下一页对应的位置，用 xpath 定位位置得到下一页链接。

4. 在 pipelines.py 中对数据处理:



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

class MyfirstPipeline(object):
    def process_item(self, item, spider):
        # 模拟浏览器
        headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64; rv:50.0) Gecko/20100101 Firefox/50.0'}
        # 使用request模块, 发送get请求
        r = requests.get(item['_piclink'], headers=headers, timeout=4)

        # 下载图片, 并保存在本地文件目录下
        # 目录
        with open('C:\\Users\\12943\\Desktop\\信息内容安全\\实验\\实验1-爬虫\\myfirst_result\\'+item['_picname']+'_'+item['_picauthor']+'.'+item['_piclink'].split('/')[-1]+'.wb') as f:
            f.write(r.content)
```

使用 request 下载图片，并保存在本地一个目录里面，命名规则为：
图片名_作者

5. 在 settings.py 中配置 ITEM_PIPELINES 选项。

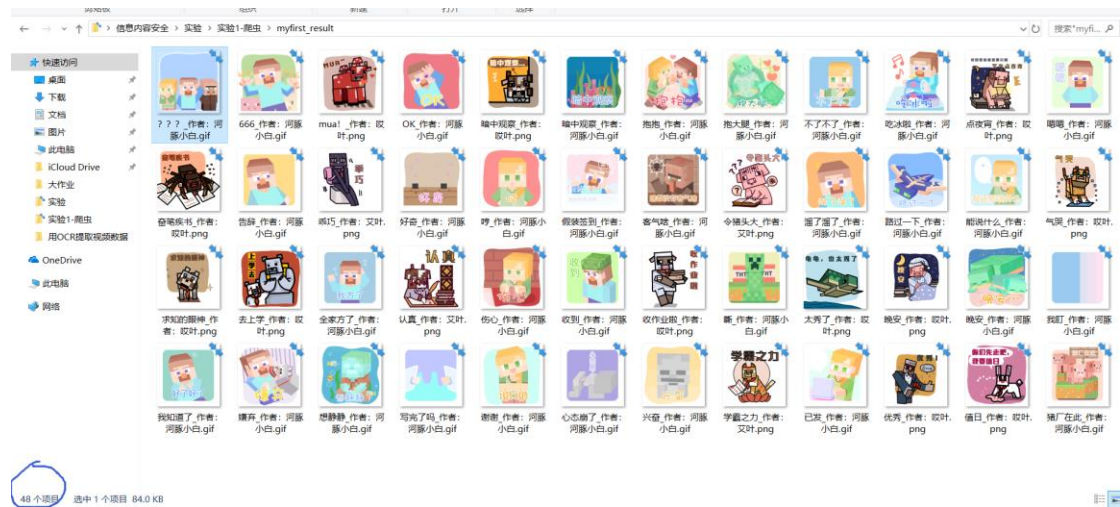
```
# See https://docs.scrapy.org/en/latest/topics/item-pipeline.
ITEM_PIPELINES = {
    'myfirst.pipelines.MyfirstPipeline': 300,
}
```

6. 运行爬虫:

```
(base) C:\Users\12943\Desktop\信息内容安全\实验\实验1-爬虫\myfirst>scrapy crawl mcbqb
```

实验结果及分析：

爬取结果为：



可以看到有 48 个图片，全部爬取成功！