

Day08回顾

selenium切换句柄

1、适用网站

1 | 页面中点开链接出现新的页面，但是浏览器对象browser还是之前页面的对象

2、应对方案

```
1  # 获取当前所有句柄（窗口）
2  all_handles = browser.window_handles
3  # 切换到新的窗口
4  browser.switch_to_window(all_handles[1])
```

创建项目流程

```
1  1、 scrapy startproject Tencent
2  2、 cd Tencent
3  3、 scrapy genspider tencent tencent.com
4  4、 items.py(定义爬取数据结构)
5  5、 tencent.py（写爬虫文件）
6  6、 pipelines.py(数据处理)
7  7、 settings.py(全局配置)
8  8、 终端: scrapy crawl tencent
```

响应对象属性及方法

```
1 # 属性
2 1、response.text : 获取响应内容
3 2、response.body : 获取bytes数据类型
4 3、response.xpath('')
5
6 # response.xpath('')调用方法
7 1、结果 : 列表,元素为选择器对象
8 2、.extract() : 提取文本内容,将列表中所有元素序列化为Unicode字符串
9 3、.extract_first() : 提取列表中第1个文本内容
10 4、.get() : 提取列表中第1个文本内容
```

爬虫项目启动方式

■ 方式一

```
1 从爬虫文件(spider)的start_urls变量中遍历URL地址,把下载器返回的响应对象(response)交给爬虫文件的
  parse()函数处理
2 # start_urls = ['http://www.baidu.com/', 'http://www.sina.com.cn']
```

■ 方式二

```
1 重写start_requests()方法,从此方法中获取URL,交给指定的callback解析函数处理
2
3 1、# 去掉start_urls变量
4 2、def start_requests(self):
5     # 生成要爬取的URL地址,利用scrapy.Request()方法交给调度器 **
```

Day09笔记

日志变量及日志级别(settings.py)

```
1 # 日志相关变量
2 LOG_LEVEL = ''
3 LOG_FILE = '文件名.log'
4
5 # 日志级别
6 5 CRITICAL : 严重错误
7 4 ERROR    : 普通错误
8 3 WARNING  : 警告
9 2 INFO     : 一般信息
10 1 DEBUG    : 调试信息
11 # 注意: 只显示当前级别的日志和比当前级别日志更严重的
```

数据持久化存储(MySQL)

实现步骤

```
1 1、在setting.py中定义相关变量
2 2、pipelines.py中新建管道类，并导入settings模块
3     def open_spider(self, spider):
4         # 爬虫开始执行1次,用于数据库连接
5     def process_item(self, item, spider):
6         # 用于处理抓取的item数据
7     def close_spider(self, spider):
8         # 爬虫结束时执行1次,用于断开数据库连接
9 3、settings.py中添加此管道
10     ITEM_PIPELINES = {'':200}
11
12 # 注意：process_item() 函数中一定要 return item ***
```

练习

把猫眼电影数据存储到MySQL数据库中

保存为csv、json文件

■ 命令格式

```
1 scrapy crawl maoyan -o maoyan.csv
2 scrapy crawl maoyan -o maoyan.json
3 # settings.py FEED_EXPORT_ENCODING = 'utf-8'
```

盗墓笔记小说抓取案例（三级页面）

■ 目标

```
1 # 抓取目标网站中盗墓笔记1-8中所有章节的所有小说的具体内容，保存到本地文件
2 1、网址：http://www.daomubiji.com/
```

■ 准备工作xpath

```
1 1、一级页面xpath（此处响应做了处理）：//ul[@class="sub-menu"]/li/a/@href
2 2、二级页面xpath：/html/body/section/div[2]/div/article
3 基准xpath：//article
4 3、三级页面xpath：response.xpath('//article[@class="article-content"]//p/text()).extract()
```

■ 项目实施

1. 创建项目及爬虫文件

```
1 创建项目 : Daomu
2 创建爬虫 : daomu daomubiji.com
```

2. 定义要爬取的数据结构 (把数据交给管道)

```
1 import scrapy
2
3 class DaomuItem(scrapy.Item):
4     # 卷名
5     volume_name = scrapy.Field()
6     # 章节数
7     zh_num = scrapy.Field()
8     # 章节名
9     zh_name = scrapy.Field()
10    # 章节链接
11    zh_link = scrapy.Field()
12    # 小说内容
13    zh_content = scrapy.Field()
```

3. 爬虫文件实现数据抓取

```
1
```

4. 管道文件实现数据处理

```
1
```

图片管道(360图片抓取案例)

■ 目标

```
1 www.so.com -> 图片 -> 美女
```

■ 抓取网络数据包

```
1 2、F12抓包,抓取到json地址 和 查询参数(QueryString)
2     url = 'http://image.so.com/zj?ch=beauty&sn={}&listtype=new&temp=1'.format(str(sn))
3     ch: beauty
4     sn: 90
5     listtype: new
6     temp: 1
```

■ 项目实施

1. 创建爬虫项目和爬虫文件

```
1 scrapy startproject So
2 cd So
3 scrapy genspider so image.so.com
```

2. 定义要爬取的数据结构(items.py)

```
1 img_link = scrapy.Field()
```

3. 爬虫文件实现图片链接抓取

```
1 |
```

4. 管道文件 (pipelines.py)

```
1 |
```

5. 设置settings.py

```
1 |
```

6. 创建run.py运行爬虫

```
1 |
```

scrapy shell的使用

■ 基本使用

```
1 1、 scrapy shell URL地址
2 *2、 request.headers : 请求头(字典)
3 *3、 request.meta : item数据传递, 定义代理(字典)
4 4、 response.text : 字符串
5 5、 response.body : bytes
6 6、 response.xpath('')
```

■ scrapy.Request()

```
1 1、 url
2 2、 callback
3 3、 headers
4 4、 meta : 传递数据, 定义代理
5 5、 dont_filter : 是否忽略域组限制
6 默认False, 检查allowed_domains['']
```

设置中间件(随机User-Agent)

少量User-Agent切换

■ 方法一

```
1 # settings.py
2 USER_AGENT = ''
3 DEFAULT_REQUEST_HEADERS = {}
```

■ 方法二

```
1 # spider
2 yield scrapy.Request(url, callback=函数名, headers={})
```

大量User-Agent切换 (中间件)

■ middlewares.py设置中间件

```
1 1、获取User-Agent
2 # 方法1：新建useragents.py,存放大量User-Agent, random模块随机切换
3 # 方法2：安装fake_useragent模块(sudo pip3 install fake_useragent)
4 from fake_useragent import UserAgent
5 ua_obj = UserAgent()
6 ua = ua_obj.random
7 2、middlewares.py新建中间件类
8 class RandomUseragentMiddleware(object):
9     def process_request(self, request, spider):
10         ua = UserAgent()
11         request.headers['User-Agent'] = ua.random
12 3、settings.py添加此下载器中间件
13 DOWNLOADER_MIDDLEWARES = {'': 优先级}
```

设置中间件(随机代理)

```
1 request.meta['proxy'] = 'http://127.0.0.1:8888'
2 ** 使用代理尝试 **
```

