

# Day09

## Day08回顾

### *selenium+phantomjs/chrome/firefox*

#### ■ 设置无界面模式 (chromedriver | firefox)

```
1 options = webdriver.ChromeOptions()
2 options.add_argument('--headless')
3
4 browser = webdriver.Chrome(options=options)
5 browser.get(url)
```

#### ■ browser执行JS脚本

```
1 browser.execute_script(
2     'window.scrollTo(0,document.body.scrollHeight)'
3 )
4 time.sleep(2)
```

#### ■ selenium常用操作

```
1 # 1、键盘操作
2 from selenium.webdriver.common.keys import Keys
3 node.send_keys(Keys.SPACE)
4 node.send_keys(Keys.CONTROL, 'a')
5 node.send_keys(Keys.CONTROL, 'c')
6 node.send_keys(Keys.CONTROL, 'v')
7 node.send_keys(Keys.ENTER)
8
9 # 2、鼠标操作
10 from selenium.webdriver import ActionChains
11 mouse_action = ActionChains(browser)
12 mouse_action.move_to_element(node)
13 mouse_action.perform()
14
15 # 3、切换句柄
16 all_handles = browser.window_handles
17 browser.switch_to.window(all_handles[1])
18
19 # 4、iframe子框架
20 browser.switch_to.iframe(iframe_element)
21
```

```
22 # 5、Web客户端验证
23 url = 'http://用户名:密码@正常地址'
```

## execjs模块使用

```
1 # 1、安装
2 sudo pip3 install pyexecjs
3
4 # 2、使用
5 with open('file.js','r') as f:
6     js = f.read()
7
8 obj = execjs.compile(js)
9 result = obj.eval('string')
```

## Day09笔记

## scrapy框架

### ▪ 定义

```
1 异步处理框架,可配置和可扩展程度非常高,Python中使用最广泛的爬虫框架
```

### ▪ 安装

```
1 # Ubuntu安装
2 1、安装依赖包
3     1、sudo apt-get install libffi-dev
4     2、sudo apt-get install libssl-dev
5     3、sudo apt-get install libxml2-dev
6     4、sudo apt-get install python3-dev
7     5、sudo apt-get install libxslt1-dev
8     6、sudo apt-get install zlib1g-dev
9     7、sudo pip3 install -I -U service_identity
10 2、安装scrapy框架
11     1、sudo pip3 install Scrapy
```

```
1 # Windows安装
2 cmd命令行(管理员): python -m pip install Scrapy
3 # Error: Microsoft Visual C++ 14.0 is required xxx
```

### ▪ Scrapy框架五大组件

```

1 1、引擎(Engine)      : 整个框架核心
2 2、调度器(Scheduler) : 维护请求队列
3 3、下载器(Downloader): 获取响应对象
4 4、爬虫文件(Spider)  : 数据解析提取
5 5、项目管道(Pipeline): 数据入库处理
6 *****
7 # 下载器中间件(Downloader Middlewares) : 引擎->下载器,包装请求(随机代理等)
8 # 蜘蛛中间件(Spider Middlewares) : 引擎->爬虫文件,可修改响应对象属性

```

## ■ scrapy爬虫工作流程

```

1 # 爬虫项目启动
2 1、由引擎向爬虫程序索要第一个要爬取的URL,交给调度器去入队列
3 2、调度器处理请求后出队列,通过下载器中间件交给下载器去下载
4 3、下载器得到响应对象后,通过蜘蛛中间件交给爬虫程序
5 4、爬虫程序进行数据提取:
6     1、数据交给管道文件去入库处理
7     2、对于需要继续跟进的URL,再次交给调度器入队列,依次循环

```

## ■ scrapy常用命令

```

1 # 1、创建爬虫项目
2 scrapy startproject 项目名
3 # 2、创建爬虫文件
4 scrapy genspider 爬虫名 域名
5 # 3、运行爬虫
6 scrapy crawl 爬虫名

```

## ■ scrapy项目目录结构

```

1 Baidu          # 项目文件夹
2 └─ Baidu       # 项目目录
3 │   └─ items.py # 定义数据结构
4 │   └─ middlewares.py # 中间件
5 │   └─ pipelines.py # 数据处理
6 │   └─ settings.py  # 全局配置
7 │   └─ spiders
8 │       └─ baidu.py # 爬虫文件
9 └─ scrapy.cfg     # 项目基本配置文件

```

## ■ 全局配置文件settings.py详解

```

1 # 1、定义User-Agent
2 USER_AGENT = 'Mozilla/5.0'
3 # 2、是否遵循robots协议,一般设置为False
4 ROBOTSTXT_OBEY = False
5 # 3、最大并发量,默认为16
6 CONCURRENT_REQUESTS = 32
7 # 4、下载延迟时间
8 DOWNLOAD_DELAY = 1
9 # 5、请求头,此处也可以添加User-Agent
10 DEFAULT_REQUEST_HEADERS={}
11 # 6、项目管道

```

```
12 ITEM_PIPELINES={
13     '项目目录名.pipelines.类名':300
14 }
```

## ■ 创建爬虫项目步骤

```
1 1、新建项目：scrapy startproject 项目名
2 2、cd 项目文件夹
3 3、新建爬虫文件：scrapy genspider 文件名 域名
4 4、明确目标(items.py)
5 5、写爬虫程序(文件名.py)
6 6、管道文件(pipelines.py)
7 7、全局配置(settings.py)
8 8、运行爬虫：scrapy crawl 爬虫名
```

## ■ pycharm运行爬虫项目

```
1 1、创建begin.py(和scrapy.cfg文件同目录)
2 2、begin.py中内容:
3     from scrapy import cmdline
4     cmdline.execute('scrapy crawl maoyan'.split())
```

# 小试牛刀

## ■ 目标

```
1 打开百度首页，把 '百度一下，你就知道' 抓取下来，从终端输出
2 /html/head/title/text()
```

## ■ 实现步骤

### 1、创建项目Baidu 和 爬虫文件baidu

```
1 |
```

### 2、编写爬虫文件baidu.py, xpath提取数据

```
1 |
```

### 3、全局配置settings.py

```
1 |
```

### 4、创建run.py (和scrapy.cfg同目录)

```
1 |
```

## 5、启动爬虫

```
1 |
```

### 思考运行过程

## 猫眼电影案例

### ■ 目标

```
1 | URL: 百度搜索 -> 猫眼电影 -> 榜单 -> top100榜
2 | 内容: 电影名称、电影主演、上映时间
```

### ■ 实现步骤

#### 1、创建项目和爬虫文件

```
1 |
```

#### 2、定义要爬取的数据结构 (items.py)

```
1 |
```

#### 3、编写爬虫文件 (maoyan.py)

```
1 | 1、基准xpath, 匹配每个电影信息节点对象列表
2 | dd_list = response.xpath('//dl[@class="board-wrapper"]/dd')
3 | 2、for dd in dd_list:
4 |     电影名称 = dd.xpath('./a/@title')
5 |     电影主演 = dd.xpath('./p[@class="star"]/text()')
6 |     上映时间 = dd.xpath('./p[@class="releasetime"]/text()')
```

#### 代码实现一

```
1 |
```

#### 代码实现二

```
1 |
```

#### 4、定义管道文件 (pipelines.py)

```
1 |
```

#### 5、全局配置文件 (settings.py)

```
1 |
```

## 6. 创建并运行文件 (run.py)

```
1 |
```

# 知识点汇总

## ▪ 节点对象.xpath()

```
1 1、列表,元素为选择器 ['<selector data='A'>]  
2 2、列表.extract() : 序列化列表中所有选择器为Unicode字符串 ['A','B','C']  
3 3、列表.extract_first() 或者 get() :获取列表中第1个序列化的元素(字符串)
```

## ▪ pipelines.py中必须有1个函数叫process\_item

```
1 def process_item(self,item,spider):  
2     return item  
3 # 必须返回item,此返回值会传给下一个管道的此函数继续处理
```

## ▪ 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
```

## ▪ 管道文件使用

```
1 1、在爬虫文件中为items.py中类做实例化,用爬下来的数据给对象赋值  
2     from ..items import MaoyanItem  
3     item = MaoyanItem()  
4 2、管道文件 (pipelines.py)  
5 3、开启管道 (settings.py)  
6     ITEM_PIPELINES = { '项目目录名.pipelines.类名':优先级 }
```

# 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 close_spider(self, spider):
6         # 爬虫结束时执行1次,用于断开数据库连接
7 3、settings.py中添加此管道
8     ITEM_PIPELINES = {'':200}
9
10 # 注意：process_item() 函数中一定要 return item ***
```

## 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中设置导出编码
4 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:
2 a节点: //li[contains(@id,"menu-item-20")]/a
3 title: ./text()
4 link: ./@href
5
6 2、二级页面
7 基准xpath: //article
8 for循环遍历后:
9     name=article.xpath('./a/text()').get()
10    link=article.xpath('./a/@href').get()
11
12 3、三级页面xpath: response.xpath('//article[@class="article-content"]//p/text()').extract()
13 # 结果: ['p1', 'p2', 'p3', '']
```

### ■ 项目实施

### 1、创建项目及爬虫文件

1 |

### 2、定义要爬取的数据结构 - items.py

1 |

### 3、爬虫文件实现数据抓取 - daomu.py

1 |

### 4、管道文件实现数据处理 - pipelines.py

1 |

### 5、全局配置 - setting.py

### 6、运行文件 - run.py

## 今日作业

- 1 1、scrapy框架有哪几大组件？以及各个组件之间是如何工作的？
- 2 2、腾讯招聘尝试改写为scrapy
- 3 response.text ：获取页面响应内容
- 4 3、豆瓣电影尝试改为scrapy