专栏：014：客观，你要的实战就在这里.

E - 爬虫技术 爬虫知识

> **用理工科思维看待这个世界**

系列爬虫专栏

初学者，尽力实现最小化学习系统

主题：Scrapy 实战，并分别存储在MySQL 和 Mongodb中

0：目标说明

- Scrapy 基础教程
  你要的最佳实战
- 刘未鹏博客
  点我啊
- 目标：获取刘未鹏博客全站博文
    - 文章标题：`Title`
    - 文章发布时间：`Time`
    - 文章全文：`Content`
    - 文章的链接：`Url`
- 思路：
    - 分析首页和翻页的组成
    - 抓取全部的文章链接
    - 在获取的全部链接的基础上解析需要的标题，发布时间，全文和链接

1：目标分解

`Scrapy` 支持 `xpath`

- 全部链接获取

```
# 首页和剩余的页获取链接的xpath有点差异
each_page_data = selector.xpath('//div[@id="index-featured1"]/ul/li/h3[@class="entry-title"]/a/@href').extract()
each_page_data_other = selector.xpath('//div[@id="content"]/div/ul/li/h3[@class="entry-title"]/a/@href').extract()
# 全部的url放在一个列表里：item_url
```

- 文章标题

```
title = selector.xpath('//div[@id="content"]/div/h1[@class="entry-title"]/a/text()').extract()
```

- 文章发布时间

```
time = selector.xpath('//div[@id="content"]/div/div[@class="entry-info"]/abbr/text()').extract()
```

- 文章全文

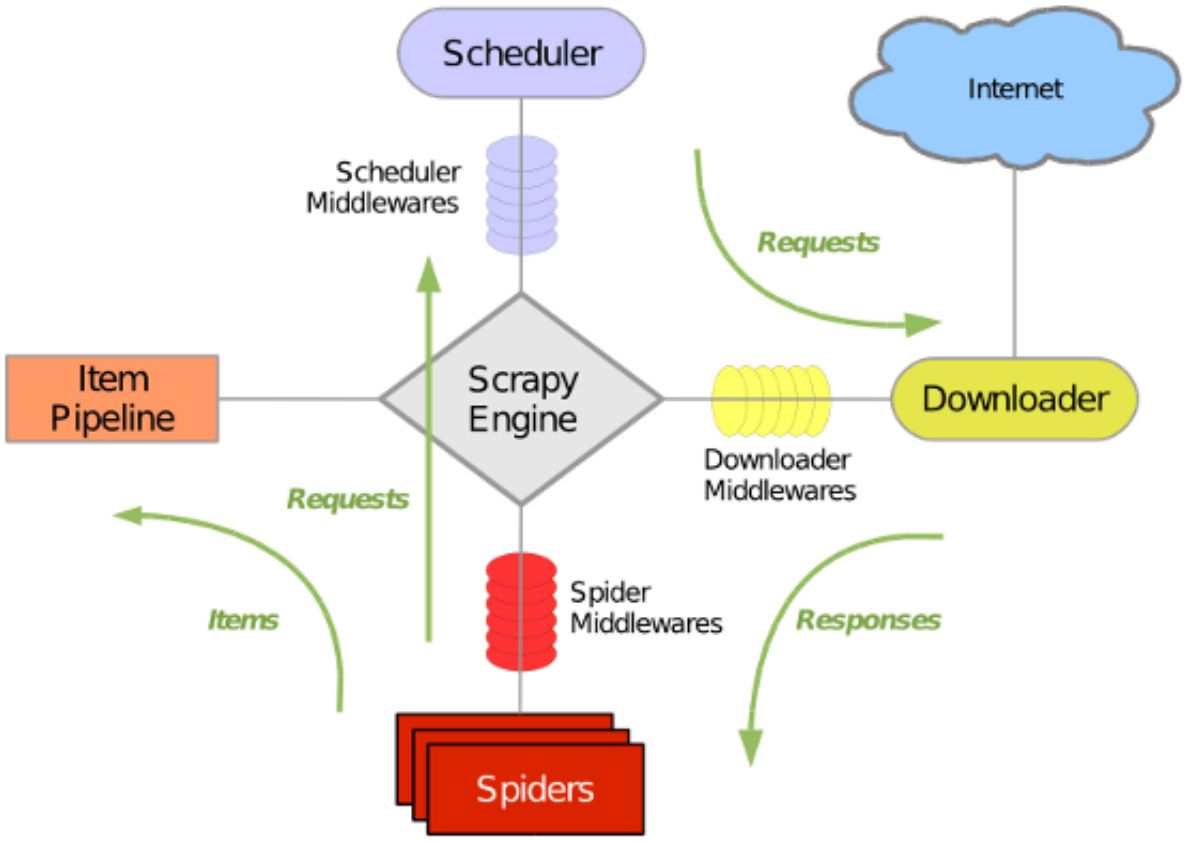
```
content = selector.xpath('//div[@id="content"]/div/div[@class="entry-content clearfix"]/p/text()').extract()
```

- 文章链接

```
url = selector.xpath('//div[@id="content"]/div/h1[@class="entry-title"]/a/@href').extract()
```

使用Scrapy 框架的基本教程：
翻译版教程

- 一般步骤
    - 新建项目
    - 定义Item：`items.py`文件是定义的抓取目标
    - 编写spider: `spiders`文件夹是用来编写爬虫文件
    - `settings.py`文件是用来编写配置文件比如头部信息，一些常量，比如MySQL用户，端口等
    - `pipelines.py`文件是用来编写存储数据操作，比如MySQL数据库的操作，mongodb数据库的操作
- Scrapy 框架的原理
  经典说明文档




```
* 引擎scrapy
* 调度器 scheduler
* 下载器 downloader
* 爬虫 spider
* 项目管道 pipeline
```

运行流程：
Scrapy运行流程大概如下：
首先，引擎从调度器中取出一个链接(URL)用于接下来的抓取
引擎把URL封装成一个请求(Request)传给下载器，下载器把资源下载下来，并封装成应答包(Response)
然后，爬虫解析Response
若是解析出实体（Item）,则交给实体管道进行进一步的处理。
若是解析出的是链接（URL）,则把URL交给Scheduler等待抓取

2：目标实战

- 编写Items 文件定义抓取目标

```
class LiuweipengItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    Title = scrapy.Field() # 标题
    Time = scrapy.Field() # 发布时间
    Url = scrapy.Field() # 文章链接
    Content = scrapy.Field() # 文章内容
```

- 编写爬虫程序

```
# 获取整个网站的文章链接
class BlogSpider(Spider):
    name = "liuweipeng"
    start_urls = ["http://mindhacks.cn/","http://mindhacks.cn/page/2/", "ht
tp://mindhacks.cn/page/3/", "http://mindhacks.cn/page/4/"]
    def parse(self, response):
        url_item = []
        selector = Selector(response)
        each_page_data = selector.xpath('//div[@id="index-featured1"]/ul/l
i/h3[@class="entry-title"]/a/@href').extract()
        each_page_data_other = selector.xpath('//div[@id="content"]/div/ul/
li/h3[@class="entry-title"]/a/@href').extract()
        url_item.extend(each_page_data)
        url_item.extend(each_page_data_other)
        for one in url_item:
            yield Request(one, callback=self.parse_detail)

#-------------------------------------------------------------------------
---------------
# 对获取的链接进行内容的解析
    def parse_detail(self, response):
        Item = LiuweipengItem()
        selector = Selector(response)
        title = selector.xpath('//div[@id="content"]/div/h1[@class="entry-t
itle"]/a/text()').extract()
        time = selector.xpath('//div[@id="content"]/div/div[@class="entry-i
nfo"]/abbr/text()').extract()
        content = selector.xpath('//div[@id="content"]/div/div[@class="entr
y-content clearfix"]/p/text()').extract()
        url = selector.xpath('//div[@id="content"]/div/h1[@class="entry-tit
le"]/a/@href').extract()
        print(content)
        for title, time, content, url in zip(title, time, content, url):
            Item["Title"] = title
            Item["Time"] = time
            Item["Content"] = content
            Item["Url"] = url
        yield Item
```

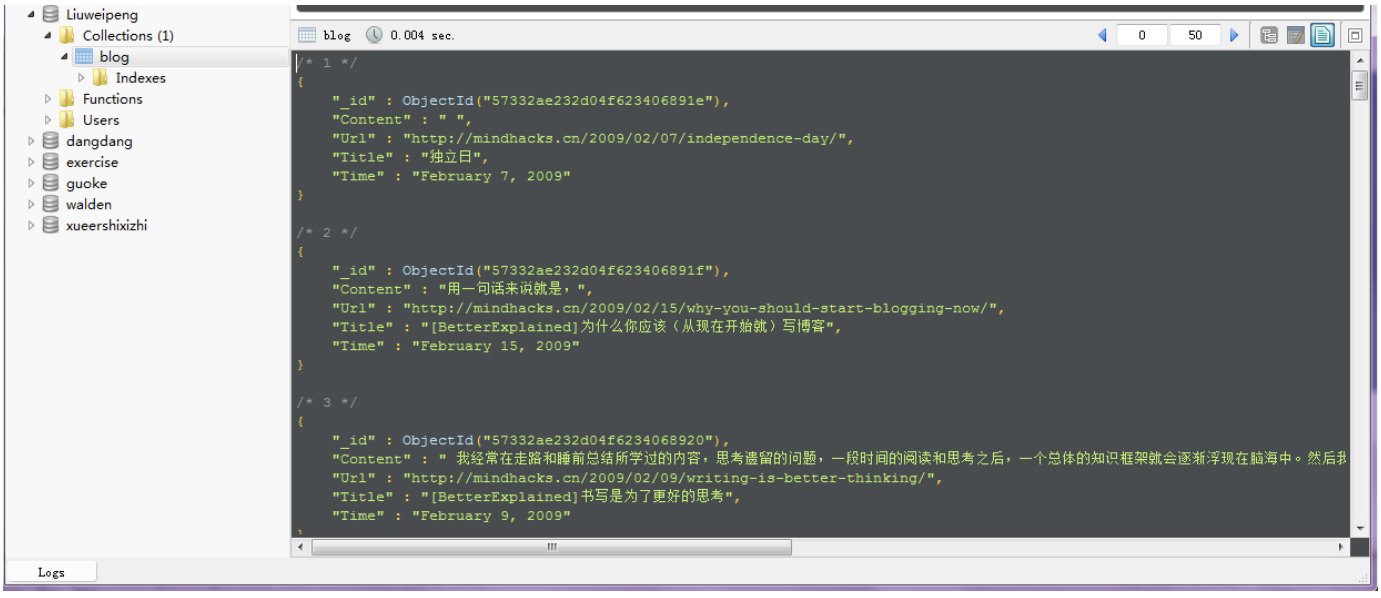
- 编写设置文件（1）：存储mongodb

```
MONGODB_HOST = '127.0.0.1' # localhost
MONGODB_PORT = 27017   # 端口号
MONGODB_DBNAME = 'Liuweipeng' # 数据库名
MONGODB_DOCNAME = 'blog' # 集合名
```

- 编写管道文件，存储数据mongodb

```
import pymongo
import pymysql
from scrapy.conf import settings
class LiuweipengPipeline(object):
    def __init__(self):
        host = settings['MONGODB_HOST']
        port = settings['MONGODB_PORT']
        dbName = settings['MONGODB_DBNAME']
        client = pymongo.MongoClient(host=host, port=port)
        tdb = client[dbName]
        self.post = tdb[settings['MONGODB_DOCNAME']]  # 初始化设置数据链接等信
息
    def process_item(self, item, spider):
        content = dict(item)
        self.post.insert(content)  # 将抓取的数据插入mongodb
```

效果显示：



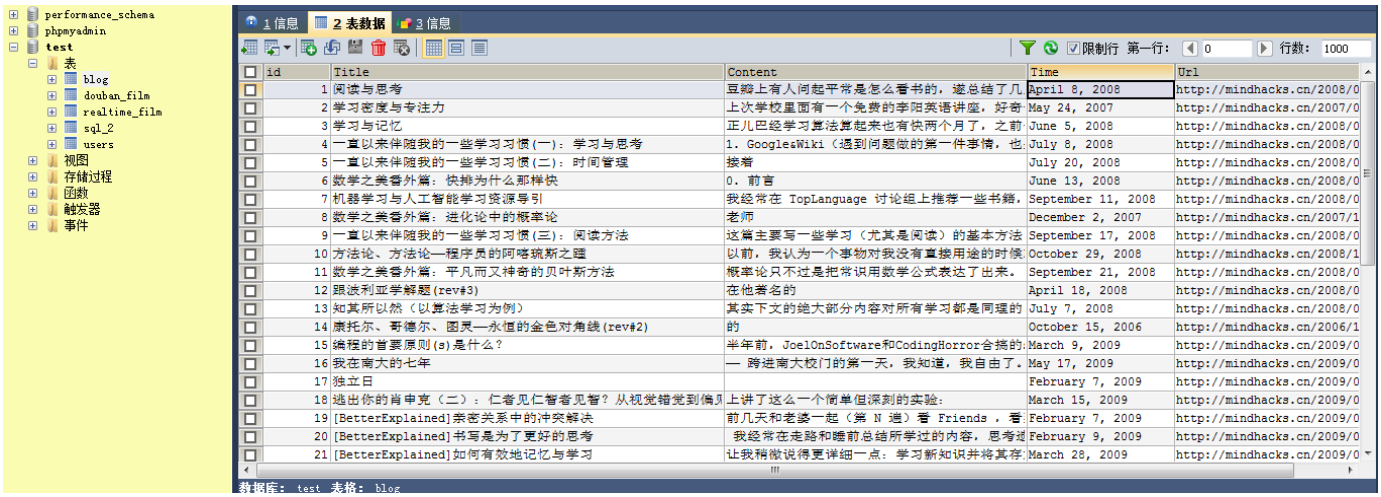
- 存储方式2：mysql

```
# 管道文件编写方式改变为：
# 这里导入的是pymysql
    def __init__(self):
        self.connection = pymysql.connect(host='localhost',
                                 user='root',
                                 password='123456',
                                 port=3306,
                                 db='test',
                                 charset='utf8')
        pass
    def process_item(self, item, spider):
        with self.connection.cursor() as cursor:
            sql = "INSERT INTO `blog`(`Title`, `Time`, `Content`, `Url`) VA
LUES (%s, %s, %s, %s)"
            cursor.execute(sql, (item['Title'],item["Time"], item["Conten
t"],item["Url"]))
        self.connection.commit()
```

- 需要在本地创建数据表：

```
# 在test数据库中创建一个blog的数据表，定义字段如下所示：
CREATE TABLE `blog` (
    `id` INT(11) NOT NULL AUTO_INCREMENT,
    `Title` VARCHAR(255) COLLATE utf8_bin NOT NULL,
    `Content` VARCHAR(255) COLLATE utf8_bin NOT NULL,
    `Time` VARCHAR(255) COLLATE utf8_bin NOT NULL,
    `Url` VARCHAR(255) COLLATE utf8_bin NOT NULL,
    PRIMARY KEY (`id`)
) ENGINE=INNODB DEFAULT CHARSET=utf8 COLLATE=utf8_bin
AUTO_INCREMENT=1 ;
```

效果显示2：



完整版代码：不点不知道bug

3：总结全文

使用`Scrapy`框架实现抓取博客，并分别使用两种存储方式。
目标分析的很详细了。
再补一句：任何实用性的东西都解决不了你所面临的实际问题，但为什么还有看？为了经验，为了通过阅读抓取别人的经验，虽然还需批判思维看待

崇尚的思维是：
了解这是什么。
知道应该怎么做。
学会亲自动手。(事实上这是我第一次使用Scrapy 框架存储在mysql中，还是遇到了好些问题)

> 关于本人：
> 只有一个职业：学生
> 只有一个任务：学习
> 在这条路上，充满无尽的困境，我希望成为一个精神世界丰满的人。

–