**web垂直类通用爬虫解决方案**

## 爬虫概述

随着互联网的发展, 搜索引擎在人们的生活工作中无疑扮演着重要的角色，而网络爬虫则是搜索引擎技术的最基础部分, 但爬虫爬取网页的基本步骤大致相同，主要分为:

1. 人工给定一个URL作为入口，从这里开始爬取。

网络爬虫一般从蝴蝶型左边结构出发。这里有一些门户网站的主页，而门户网站中包含大量有价值的链接。

1. 用运行队列和完成队列来保存不同状态的链接

对于大型数据量而言，内存中的队列是不够的，通常采用MQ队列或者数据库或者K-V（redis）方案之类的。用这种方法既可以进行海量的数据抓取，还可以拥有断点续抓功能。

1. 线程从运行队列读取队首URL，如果存在，则继续执行，反之则停止爬取。
2. 每处理完一个URL，将其放入完成队列，防止重复访问。
3. 每次抓取网页之后分析其中的URL（URL是字符串形式，功能类似指针），将经过过滤的合法链接写入运行队列，等待提取。
4. 重复步骤 3、4、5

## 爬虫的特点

根据自己的业务特点, 从目标网站获取需要的信息.

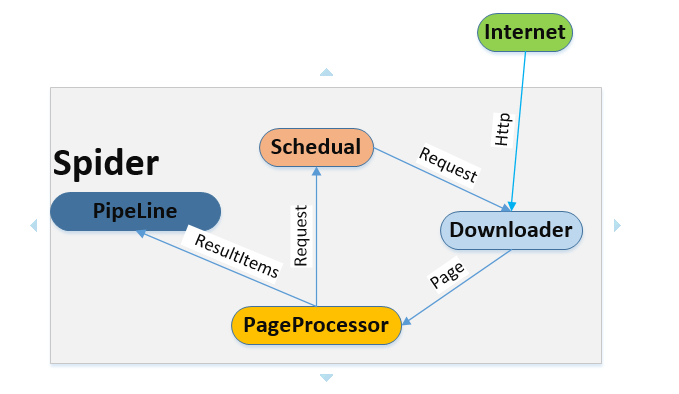
a) 将目标网站的页面尽可能快速的扒下来.

b) 然后解析出有用的内容.

c) 落地存储到db中.

## 技术架构

### 爬虫架构



组件介绍:

**Spider**是爬虫的入口类，Spider的接口调用采用了链式的API设计，其他功能全部通过接口注入Spider实现，下面是启动一个比较复杂的Spider的例子.

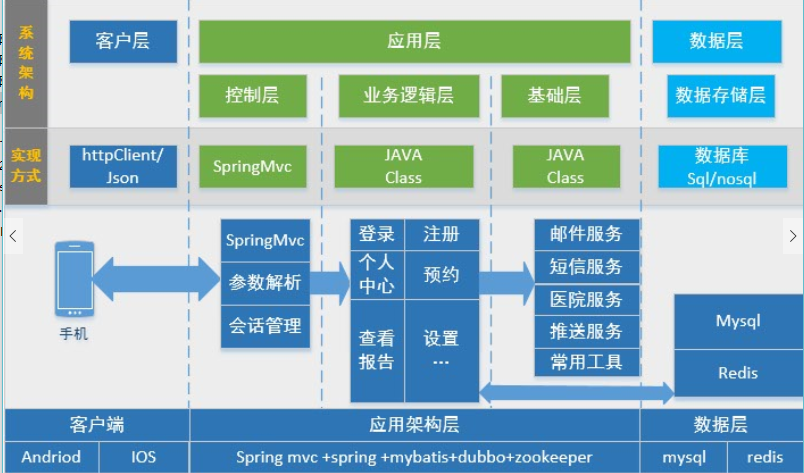
**Downloader**是webmagic中下载页面的接口.

**PageProcessor**接口来实现定制爬虫.

**Scheduler是webmagic的管理模块，通过实现Scheduler可以定制自己的URL管理器.**

**Pipeline是最终抽取结果进行输出和持久化的接口.**

### Web 架构/APP架构



## 爬虫案例

* 京东商城商品的爬取.
* 寻医问药网站疾病和药品库的爬取.

## 爬虫的维护

根据爬虫的特点，主要会涉及到目标网站的结构发生改变,导致解析和落地改变.

从以上特点在系统部署的时候会分为:

* spider（爬取模块）
* parser（解析及db入库模块）
* schdule(更新爬取计划任务模块)

模块1）与3）可以打包在同一个jar中集中部署，模块2）单独部署，之所以这样设计，出于以下考虑：

通常最终保存内容的db小型项目中只会有一个，尽管爬虫支持多线程并发爬取，可以分布式的高效狂爬，但是db是一个慢速的IO瓶颈，如果把 "爬取->解析->入库"全放在一个模块中按顺序同步处理，最后的结果相当于前面有几个水管收集水源，但是最后进入水库的总管道不给力，整体 的蓄水效率还是很慢。

分开之后，爬取模块可以同时部署多个，然后将得到的html集中存储在1个目录下，再按子目录存储（比如：一个大型网站，会有很多分站，可以实例A 爬上海站，实例B爬北京站...）,这样可以尽可能快的把内容先捞回来。然后由解析模块，再到这个目录下将文件取出来慢慢解析入库，解析成功后将原始文件 删除（或移到其它目录备份，这个看情况而定），如果代码有问题，比如解析规则有bug，导致某些页面解析失败，因为原始html文件已经在本机存储，修正 解析的bug后，可以再试重新解析失败的文件，而不需要重新爬取。

至于爬取模块完成后，如何通知解析模块去开始解析入库，有很多办法，比如消息队列，zookeeper订阅节点状态变化，或者在某个目录下放置一个标记文件 之类的都可以。