MongoDB: Mongobooster is free for non-commercial use, or robomongo.

互联网上有无穷无尽的信息，人工收集和分析是一件费时费力的事情。如何让解放人力，让电脑来自动处理，就需要编程了。这里选用 Python, 及用 Python 写的 framework Scrapy 来做一个小项目，爬取小米 AppStore 网站上的 Android app 信息。

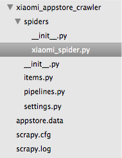
**1.**新建 **Scrapy**项目

在 terminal 上，activate自己想用的 Python virtualenv, “pip install” scrapy (必备) 及 pymongo (如果需要将结果存入 MongoDB database).

然后cd 到项目路径，执行以下命令行：

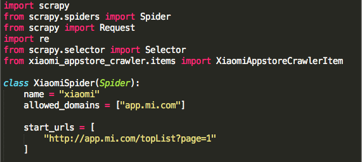
scrapy startproject xiaomi\_appstore\_crawler

就会在当前路径上生成以下目录结构 (除了xiaomi\_spider.py)：



**2. xiaomi\_spider.py**

这里放着自定义爬网 spider 的主要代码。



首先import 相关的 classes 和 functions, 然后定义一个 class XiaomiSpider, 用来定义该 spider 的主要功能。

在 XiaomiSpider 的最前端，定义了 name, allowed\_domains, start\_urls, 看字面就可以知道是做什么用了。

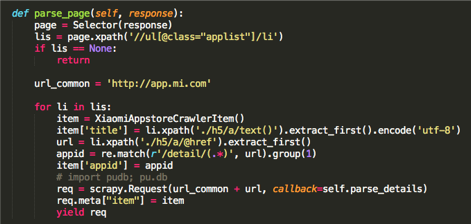
然后定义 parse function



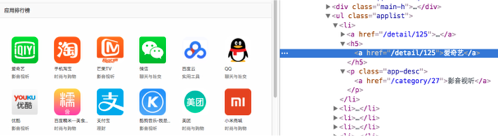
这是 spider 最先调用的 function. 它分析了 appstore top-list的首页，提取出有多少页数。如下图所示，页码数是在主页面的 <div class=”pages”> 里的 <a> 的 text。 同学们可以看一下 code, 思考一下是如何提取出总页数的。其中，xpath 是通过网页树状结构找到信息节点。然后在 for loop 里，调用 parse\_page 来爬取每一页上的 apps 信息。



这是 parse\_page function, 它分析了网页上和 app 信息有关的部分，获取了诸如 appid, title 等信息。



结合网页的如下结构图，就可以和 python code 对照，了解如何通过 xpath 来找到想要的信息。

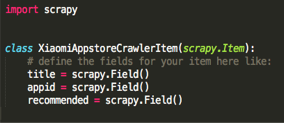


 为了取得更详细的 app 详细，parse\_page 中用了 parse\_details 来 callback. 它可以获得每个 app 的相关 app 列表。



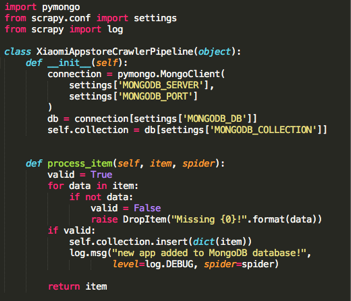
**3. items.py**

 items.py 定义了需要爬取的信息项。这也是存入数据库中的数据 column.



**4. piplines.py**

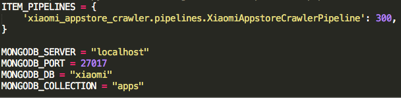
 piplines.py 定义了如何处理爬取到的数据。这里，我们把数据保存到 MongoDB 中。\_\_init\_\_ 函数中定义了基本的 MongoDB 参数。process\_item 检验每一组数据是否完备。



**5. settings.py**



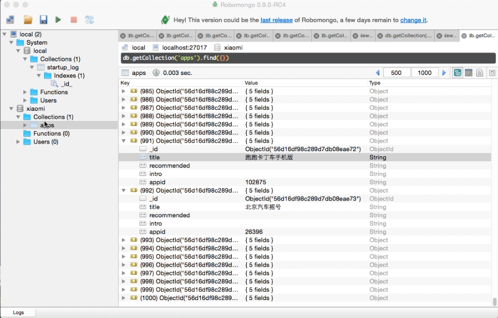
DOWNLOAD\_DEPAY=1



settings.py 是配置文件。大部分配置是 commented out 的，只有需要用时才 enable。DOWNLOAD\_DELAY 用来让爬取每条纪录中间有停顿，防止被 appstore server 认为是恶意行为而封锁。

**6. 结果**

存入 MongoDB 的数据可以用 Robomongo 查看, 如下图所示。



来源： 太阁小组