Санкт-Петербургский Государственный Университет
Математико-механический факультет
Кафедра системного программирования

Кудряшова Анна Александровна

Загрузка и хранение информации для приложения “Дом вещей”

Курсовая работа

Научный руководитель:
ст. преп. Баклановский М.В.

Санкт-Петербург
2017

Оглавление

Введение

В современном мире человек пользуется разнообразными видами цифровой и бытовой техники. И с каждым днем число и сложность устройств возрастает. В связи с этим, становится все труднее помнить важную информацию о каждом устройстве.

Для решения данной проблемы было создано приложение “Дом вещей”. С помощью данного приложения можно хранить всю информацию об устройстве: место покупки, дата покупки, цена, инструкция по эксплуатации, фирма изготовитель и так далее. Пользователю не нужно самому вводить информацию, она будет загружена приложением по производителю и номеру модели.

Используя данное приложение, человек в первую очередь экономит свое время. Ему не нужно искать инструкцию или параметры устройства, они уже сохранены в приложении. Не нужно запоминать цену, дату и место покупки устройства, приложение помнит об этом за вас.

В настоящее время данную функциональность имеют приложения для складского учета, но они используются в промышленных масштабах и избыточны для бытового использования. Мобильное приложение на базе Android позволяет иметь доступ к информации в любой момент времени.

Цели и задачи

Целью курсовой работы являлась реализация функциональности мобильного приложения по загрузке и хранению информации о цифровых и бытовых устройствах пользователя.

Для достижения поставленной цели нужно было выполнить следующие задачи:

- сделать обзор методов парсинга и краулинга веб-страниц
- реализовать поиск по ключевым словам средствами поисковых систем
- реализовать парсинг страниц с целью извлечения необходимой информации
- реализовать хранение информации в базе данных
- реализовать скачивание документов через скрипт
- реализовать скачивание изображения
- реализовать работу загрузчика на сервере
- интегрировать систему в android-приложение

Обзор предметной области

Краулер или паук (англ. *crawler*) — программа предназначенная для перебора страниц Интернета. Она анализирует содержимое страницы, сохраняет его в некотором специальном виде.

Веб-скрепинг (англ. *web-scraping*) — процесс извлечения информации из интернета. С целью ее хранения и дальнейшего использования.

Единый указатель ресурса (англ. *Uniform Resource Locator, URL*) — единообразный локатор (определитель местонахождения) ресурса. URL служит стандартизированным способом записи адреса ресурса в сети Интернет.

Обзор инструментов

В качестве языка реализации был выбран Python.

На Python существует несколько инструментов и библиотек для краулинга сайтов. Для выбора наиболее подходящего инструмента была составлена сравнительная таблица для библиотеки BeautifulSoup и фреймворка Scrapy *таблица 1*

Таблица 1

Функциональность	BeautifulSoup	Scrapy
Отправка запросов	+ (необходима доп. библиотека Request)	+
Проход по ссылкам в глубину	-	+
Защита от распознания бота	+ (необходимо настраивать самостоятельно)	+ (настраивается автоматически)
Загрузка изображений и pdf файлов	+ (необходима доп. библиотека)	+

В результате анализа было выявлено, что наиболее подходящим инструментом является фреймворк Scrapy, так как при его использовании:

- нет необходимости устанавливать дополнительные библиотеки для отправки запросов
- информативная документация
- обширная функциональность

Для выбора способа хранения найденной информации также был произведен сравнительный анализ.

Таблица 2

Функциональность	MongoDB	SQLite+Sqlalchemy
Поддержка кириллицы	+	+
Использование на android	-	+
База представлена одним файлом	-	+
Быстрый отклик	+	+

В результате сравнения была выбрана встраиваемая база данных SQLite и инструментарий по работе с БД Sqlalchemy.

Реализация

Архитектура загрузчика

Разработанное решение загрузчика состоит из трех основных частей: поиска страницы устройства на Яндекс.Маркет[1], парсинг страницы и загрузки инструкции с сайта. После чего вся информация сохраняется в базу данных. Взаимодействие этих компонент схематично показано на рисунке 1.

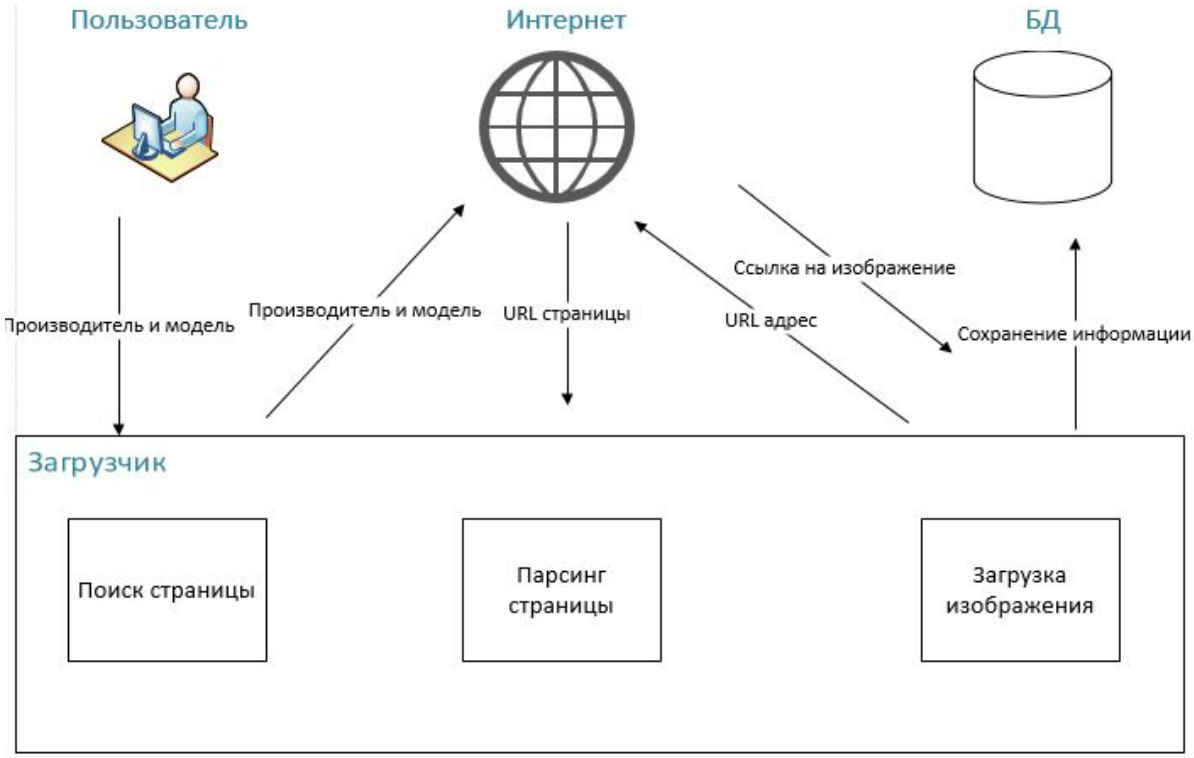

Рисунок 1. Архитектура загрузчика.

Пользователь вводит название фирмы производителя и модель устройства, которое он хочет добавить. Эти данные передаются загрузчику и по ним осуществляется поиск страницы устройства на Яндекс.Маркет. Результатом работы данного этапа является получение URL адреса страницы данного устройства. На следующем этапе происходит парсинг найденной страницы с целью

[1] https://market.yandex.ru/

извлечения необходимой информации. Третьим этапом в работе загрузчика является получение инструкции в формате pdf с сайта. На заключительном этапе вся найденная информация сохраняется в базу данных.

Рассмотрим реализацию каждой из компонент подробнее.

Поиск страницы

Было рассмотрено несколько способов реализации данной задачи. Первый — использование библиотеки xgoogle, которая осуществляет поиск страниц используя поисковую систему Google. Второй способ был найден при анализе URL адреса страницы с результатами поиска. Производитель и модель подставляются в URL адрес в качестве атрибутов. В итоге было решено использовать второй способ, так как он не требует использования дополнительных библиотек.

Парсинг страницы

Парсинг страницы осуществляется с помощью фреймворка Scrapy[1]. Поиск необходимой информации производится по содержимому текста средствами синтаксиса xpath[2]. Пример кода по извлечению информации со страницы на Яндекс.Маркет представлен в Приложении 1.

Скачивание инструкции и изображения

Для загрузки инструкции было рассмотрено несколько возможных источников:

1. Сайт производителя
2. Сайты интернет-магазинов

В первом случае реализовать загрузку инструкции программно не представляется возможным, так как структура каждого сайта уникальна, и с помощью краулера получить эту информацию не представляется возможным.

Во втором случае были рассмотрены несколько интернет-магазинов:

- МВидео[2]
- Юлмарт[3]
- МедиаМаркт[4]
- DNS[5]

В случае с МВидео и Юлмарт получить доступ к инструкции с помощью scrapy не представляется возможным, так как она находится не на отдельной странице, а на одной из вкладок на текущей странице. Структурно все вкладки — это один элемент. Пример представлен на рисунке 2.

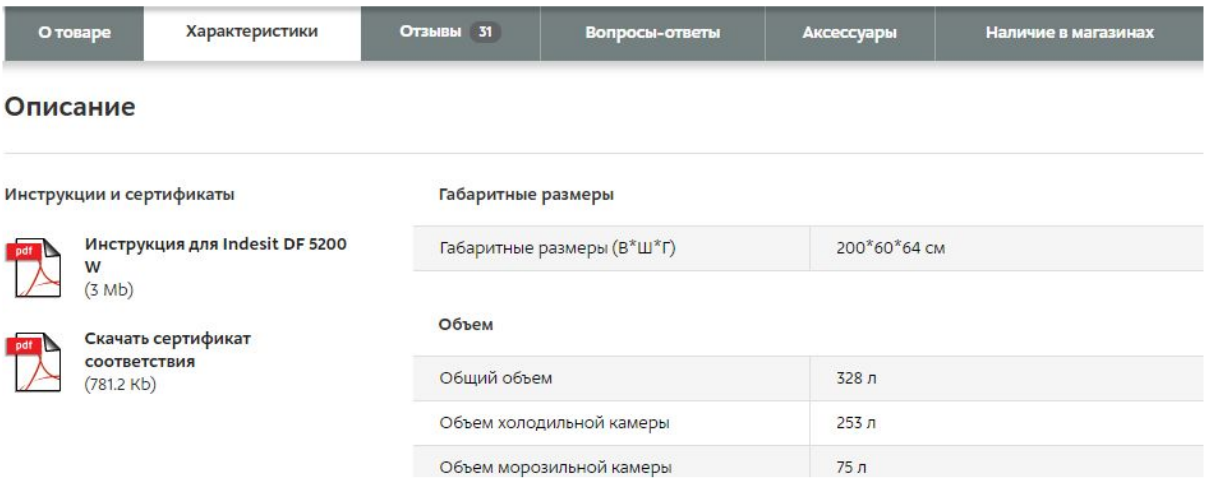

Рисунок 2. Сайт магазина МВидео.

Из-за особенностей работы фреймворка он воспринимает только первую вкладку, даже при указании необходимых настроек. При попытке загрузить инструкцию с МедиаМаркт обнаружился баг в работе фреймворка. Scrapy использует для своей работы событийно-ориентированный сетевой фреймворк Twisted. При попытке получить ответ с сайта возникает ошибка в формате ответа: сайт не посылает сообщение об отсутствии ошибок и Twisted не может продолжать свою работу. По словам разработчиков фреймворка, о данной ошибке в работе известно и она будет устранена в следующих версиях. После решения данной проблемы

[2] http://www.mvideo.ru
[3] https://www.ulmart.ru
[4] https://sankt-peterburg.mediamarkt.ru
[5] http://www.dns-shop.ru

инструкцию можно будет скачать, так как вся информация о товаре представлена без использования вкладок.

В остальных интернет-магазинах возможность скачать инструкцию не предоставляется.

Скачивание изображения осуществляется с Яндекс.Маркет средствами фреймворка Scrapy.

Хранение информации

Для хранения информации используется единая для всех типов устройств концепция, которая включает в себя следующие параметры:

- модель
- производитель
- тип устройства
- параметры(ДхШхВ)
- вес
- ссылка на сайт производителя
- url адрес изображения
- ссылка на страницу устройства на Яндекс.Маркет

Специфичную информацию о товаре пользователь может посмотреть на сайте производителя по предоставленной ему ссылке.

Хранение организовано в базе данных SQLite[3] с использованием инструментария SQLAlchemy[4]. Структура хранения данных об устройстве представлена на рисунке 3. Выделенные поля являются обязательными для заполнения.

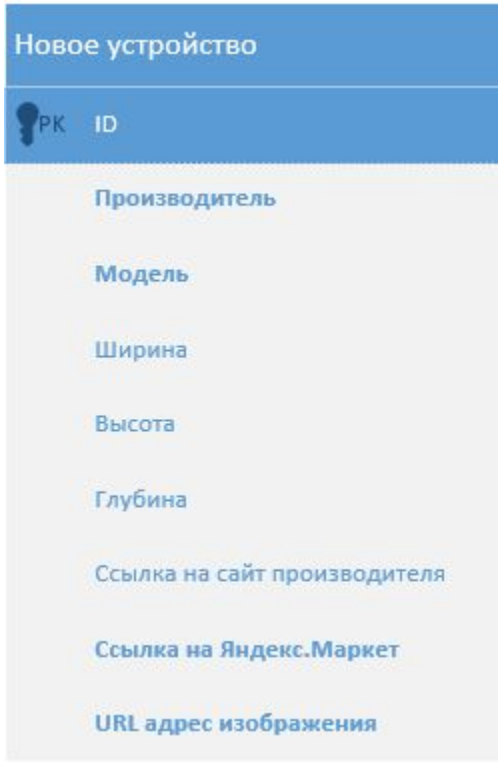

Рисунок 3. Структура элемента базы данных

Веб-сервер

Веб-сервер работает на основе веб-фреймворка Flask[5]. Запуск пауков осуществляется локально с сохранением новой информации в базе данных и последующей ее отправкой приложению. Архитектура взаимодействия сервера и клиента представлена на рисунке 4.

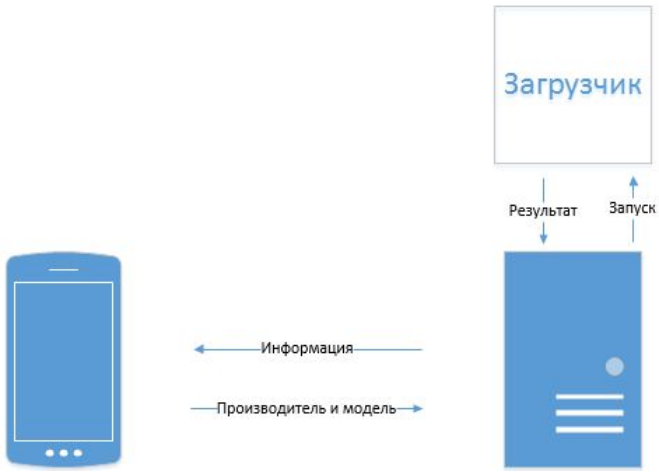

Рисунок 4. Архитектура взаимодействия сервера и клиента.

Страница устройства после загрузки информации представлена на рисунке 5.

Рисунок 5. Скриншот приложения “Дом вещей”.

Результаты

В результате выполнения курсовой работы были выполнены следующие задачи:

- изучена предметная область парсинга сайтов с использованием краулеров
- сделан обзор существующих инструментов для парсинга сайтов на языке Python
- реализован скрипт по поиску, загрузке информации и изображений с html страниц
- изучена работа базы данных SQLite
- реализован скрипт по сохранению найденной информации в базу данных
- реализована работа загрузчика на локальном сервере
- реализовано взаимодействие локального сервера и android приложения

Список литературы

1. Scrapy 1.3 documentation
   https://media.readthedocs.org/pdf/scrapy/latest/scrapy.pdf
2. Jonathan Robie, Don Chamberlin XML Path Language (XPath) 3.0 documentation
3. SQLite documentation
4. Michael Bayer SQLAlchemy documentation
   http://docs.sqlalchemy.org/en/latest/dialects/sqlite.html
5. Armin Ronacher Flask 0.12 documentation
   http://flask.pocoo.org/docs/0.12/

Приложение 1

```
class InfoSpider(CrawlSpider):
    name = "inf"

    allowed_domains = ["market.yandex.ru","avatars.mds.yandex.net"]

    with open("urls.txt", "rt") as f:
        start_urls = [url.strip() for url in f.readlines()]

    rules = (
       Rule(LinkExtractor(allow=('spec')), callback='parse_item'),
    )

    def parse_item(self, response):
        hxs = Selector(response)

        Item = CommonInfoItem()
        model = hxs.xpath("//div[@class='n-title__text']/h1/a/text()").extract()[0]
        link_mfr = hxs.xpath("//div[@class='product-spec-wrap__body']/ul/li/a[contains(text(),'www')]/@href").extract()
        if link_mfr != []:
            Item['link_mfr'] = link_mfr[0]
        else: Item['link_mfr'] = ''

        dimens = hxs.xpath(u'//div[@class="layout__col layout__col_size_p75 n-product-spec-wrap"]/div/'
                           u'dl[dt/span[contains(text(),"\u0428x\u0413x\u0412") '
                           u'or contains(text(),"\u0420\u0430\u0437\u043C\u0435\u0440")]]/dd/span/text()').extract()
        if dimens != []:
            Item['dimensions'] = dimens[0]
            Item['width'] = dimens[0].split("x")[0] + " " + dimens[0].split(" ")[1]
            Item['height'] = dimens[0].split("x")[2]
            Item['deep'] = dimens[0].split("x")[1] + " " + dimens[0].split(" ")[1]
        else:
            Item['dimensions'] = ''
            Item['width'] = ''
            Item['height'] = ''
            Item['deep'] = ''
```

Рисунок 6. Пример кода по парсингу страницы для извлечения ссылки на сайт производителя и параметров устройства.