



Проектирование мобильного приложения для поиска междугородних рейсов автобусного транспорта с помощью API Яндекс.расписания

А.Н. Вильданов

*Нефтекамский филиал ФГБОУ ВО «Башкирский государственный университет»,
г. Нефтекамск*

Аннотация: Описаны методология и технические приемы, используемые для создания мобильного приложения, работающего на базе ОС Android, для поиска междугородних автобусных пассажирских рейсов и попутных автомашин, с помощью API, предоставляемого компанией Яндекс. Разработан агрегатор, который по запросу пользователя выводит расписание автобусов и попутных рейсов. Описанная методология может оказаться полезной при проектировании и разработке современных мобильных приложений.

Ключевые слова: Java, Android, парсинг, Jsoup, API Яндекс.

1. Постановка задачи

В наше время мобильный телефон практически у каждого всегда имеется под рукой. Это обусловило стремительный рост удельного веса разработок приложений именно для мобильных платформ. Роль мобильных приложений в сегодняшней повседневной жизни трудно переоценить. Разрабатываются приложения для решения самых разнообразных задач: навигация внутри зданий [1], меню для кафе быстрого питания [2], мобильный спасатель для вызова экстренных служб [3], и т.д.

Сегодня, несмотря на наличие в достаточном количестве и широком разнообразии как личного транспорта, так и услуг такси, автобусный транспорт остается одним из наиболее востребованных способов передвижения. Ведь многие из нас при поездках в другой населенный пункт пользуются услугами пассажирских автобусов. Это обычно и дешевле, и надежнее, и безопаснее. Главное – не опоздать на свой рейс. Несомненно, задача составления приложения, которое могло бы быстро найти и показать актуальное время нужного рейса, является востребованной.

Необходимую информацию часто можно получить на соответствующем специализированном веб-сайте. Но пользование сайтом не

всегда оказывается удобным. У пассажира под рукой чаще только сотовый телефон с мобильным браузером, а не компьютер или планшет. Разработчикам же сайтов не всегда хватает времени и желания адаптировать сайт под мобильное устройство, а также предусмотреть различные «вкусности» для пользователя: запоминание выбранных рейсов, хранение результатов поиска, и т.д.

Эти и другие возможности можно добавить в мобильное приложение. В данной работе мы рассмотрим технические аспекты реализации разработки мобильного приложения, которое по запросу пользователя обращается на серверы Яндекса с расписанием рейсов, находит список попутных ему рейсов и выводит пользователю.

2. Алгоритмы и инструменты решения

Как вообще мобильное приложение берет информацию с сайта (или с сервера)? Можно выделить несколько основных подходов к проектированию подобных приложений.

1. Мобильное приложение загружает те же страницы сайта, к которым обращаются и обычные пользователи сайта с браузерами. Затем производится анализ текста со скачанных страниц, выделяется и обрабатывается нужная информация, и отображается в приложении с помощью виджетов.

Недостаток такого подхода – на странице может быть много лишней информации, рекламы, и пр., что может существенно повлиять как на скорость загрузки страницы, так и на ее лексический анализ, и в конечном итоге отрицательно скажется на скорости работы приложения.

Кроме того, даже сравнительно небольшое изменение структуры сайта может привести к тому, что приложение не будет работать. Разработчикам приходится менять алгоритм анализа страницы и выпускать новую версию приложения.

2. Разработчик мобильного приложения создает сайт-«зеркало» того сайта, с которого собирается брать информацию. Новый сайт периодически мониторит сайт-источник и копирует нужную информацию себе. Мобильное приложение обращается не к сайту, а к его зеркалу.

Это необходимо, например, когда сайт работает слишком медленно, или нужна авторизация, и т.д. Правда, информация на сайте-«зеркале» может немного устаревать. Но зато зеркало может собирать информацию с нескольких сайтов. Это уже значительно расширяет возможности мобильного приложения, ведь часто сайты-агрегаторы пользуются бóльшим спросом, чем сайты-оригиналы. Сегодня агрегируют или пытаются агрегировать практически все: электриков и сантехников, репетиторов, сауны, турфирмы, типографии, отели, вакансии и пр. [4].

3. Разработчики или владельцы сайта заранее предполагают, что их информация будет использоваться на других сайтах или приложениях, и предусмотрели API для разработчиков. Мобильное приложение шлет на сервер сайта специально построенные запросы, и получает в ответ только необходимую ему информацию (без новостей, рекламы и пр.). API имеются у крупных социальных сетей (Вконтакте, Одноклассники), почтовых и поисковых сервисов (Yandex, Google, Mail.ru и пр.)

Сюда же можно отнести также наличие у сайта RSS-ленты, информация в которой обычно уже структурирована в xml-формате, что облегчает ее получение и обработку. Этот подход (RSS-лента) уже постепенно устаревает.

Работа с API лишена недостатков из предыдущих пунктов, так как обычно изменение структуры сайта никак не связано с API, меняется только способ вывода информации. API обращается непосредственно к серверу, и информация, полученная API, является актуальной.



Рассмотренные подходы могут и пересекаться, например, иногда разработчик мобильного приложения может сам пойти и договориться с владельцами сайта, чтобы они разработали и предоставили ему такое API, если, конечно, это принесет какую-либо выгоду для последних. Наконец, вполне может иметь место и такой случай, когда работа происходит внутри одного предприятия, но за сайт и приложение отвечают разные разработчики.

В данной работе будет использоваться третий подход, основанный на API-технологиях российской транснациональной компании Яндекс. Яндекс предоставляет разнообразные API для решения широкого круга задач: API Карт, API Расписаний, API Аудиторий, API Яндекс.Погоды, API Вебмастера, и т.д. Для решения поставленной задачи нам понадобится Яндекс.Расписание – сервис с актуальными расписаниями самолётов, поездов, электричек, автобусов, теплоходов и паромов. Через бесплатный API Яндекс.Расписаний можно получать следующие данные для всех видов транспорта, представленных на сервисе:

- расписание рейсов между выбранными станциями;
- расписание рейсов, отправляющихся от указанной станции;
- маршрут с указанием времени в пути и всех остановок;
- контактную информацию перевозчиков;
- список ближайших к заданной точке станций.

Эти данные API Яндекс.Расписаний выдает в актуальных сегодня текстовых форматах, таких, как JSON или XML. Для получения доступа к информации нужно получить ключ. Его можно получить бесплатно в кабинете разработчика, и активировать его (обычно на это уходит 3-10 дней).

Итак, нам нужно загрузить страницу с результатами поиска, обработать ее и отобразить. Здесь мы уже вплотную подошли к такому понятию, как парсинг.

Сегодня парсинг сайтов является эффективным решением для автоматизации процесса получения и обработки информации. Под парсингом обычно понимают получение текстового содержимого страниц сайта с последующим разбором с целью извлечения данных [5]. Разбор структурированного текста, в свою очередь, может быть осуществлен с помощью регулярных выражений или программных библиотек, предназначенных для восстановления его древовидной структуры. Но регулярные выражения для сложных текстов теряют наглядность, их становится трудно воспринимать и отлаживать. Построение структурного дерева текста требует больших затрат памяти и является излишним для многих практических задач [6].

Для выполнения запросов к серверу и парсинга содержимого веб-страницы на языке Java будем использовать библиотеку Jsoup [7]. Она обеспечивает удобный API для работы с данными, используя DOM, CSS, и JQuery-подобные методы. Jsoup предназначена, в первую очередь, для работы с HTML-содержимым веб-страницы, но вполне подойдет и для анализа XML.

Для обработки полученного ответа в формате JSON на языке Java удобно использовать Gson. Gson - это небольшая Java-библиотека, которая позволяет конвертировать Java объекты в их JSON представление и наоборот.

В качестве программной среды разработки мобильного приложения выберем Android Studio. Сегодня это официальная от компании Google среда разработки под Android. Android Studio – результат сотрудничества компаний JetBrains и Google, и Google рекомендует именно эту среду для разработки приложений [8].

Android Studio позволяет создавать на языке программирования высокого уровня Java нативные, т.е. «родные» для операционной системы Android приложения. Нативность, например, означает, что разработанное в

AS приложение имеет достаточно высокую скорость отклика в работе и минимум «глюков» и вылетов. Конечно, скорость работы приложения будет зависеть и от скорости работы сервера.

3. Реализация

Для получения расписания рейсов между станциями в API Яндекс.Расписаний используется достаточно простой и интуитивно понятный по содержанию и форме запрос. Этот запрос позволяет получить список рейсов, следующих от указанной станции отправления к указанной станции прибытия и информацию по каждому рейсу. Основные параметры запроса приведены в таблице №1:

Таблица № 1

Параметры запроса рейсов между станциями в API Яндекс.Расписаний

Параметр	Описание
apikey	Ключ доступа к API.
from	Код станции отправления.
to	Код станции прибытия.
format	Формат ответа. Поддерживаемые значения: <ul style="list-style-type: none">• json (по умолчанию);• xml.
lang	Язык возвращаемой информации, например RU – Россия;
date	Дата, на которую необходимо получить список рейсов в формате YYYY-MM-DD. По умолчанию возвращается информация обо всех будущих рейсах.
transport_types	Тип транспортного средства. Например <ul style="list-style-type: none">• bus — автобус;

Пример запроса:

<https://api.rasp.yandex.net/v3.0/search/?apikey={ключ}&format=xml&from=c146&to=c213&lang=ru&page=1&date=2015-09-02>



Ответ сервера представляет собой список рейсов с информацией по каждому рейсу. Возможные форматы ответа: JSON, XML.

Код города можно узнать, отправив запрос вида

`https://yandex.ru/bus/api/v1/suggest?part=нефт`

с фрагментом названия города («нефт»). Ответ в формате JSON имеет достаточно простую структуру:

```
{
  "items": [{
    "raspId": "c11184",
    "displayName": "Нефтеюганск",
    "description": "г. Нефтеюганск, Ханты-Мансийский автономный
округ - Югра, Россия"
  }, {
    "raspId": "c11114",
    "displayName": "Нефтекамск",
    "description": "г. Нефтекамск, Республика Башкортостан, Россия"
  }, {
    "raspId": "c20655",
    "displayName": "Нефтекумск",
    "description": "г. Нефтекумск, Ставропольский край, Россия"
  }
]
```

Этот же ответ используется для создания выпадающего списка при наборе названия городов пользователем.

Как видим, информацию по имеющимся по данному маршруту рейсам можно получить достаточно просто. Достаточно много времени при разработке для Android уходит на разработку удобного интерфейса. Для придания приложению современного вида (со «шторкой» и «гамбургером») будем использовать класс `drawerLayout` (рис. 1). За основу приложения возьмем шаблон, разработанный в [9].

Поиск рейсов осуществляется с помощью соответствующего пункта меню (рис. 2). Приложение позволяет искать расписание на определенную дату, или же получить сводное (на все дни) расписание. Во втором случае отобразятся все имеющиеся рейсы между указанными пунктами, с днями недели, по которым эти рейсы осуществляются. Разница между этими двумя способами выбора расписания заключается еще и в том, что, вообще говоря, некоторые рейсы могут быть отменены, например, из-за совпадения даты рейса с праздничным днем, и т.п. В сводном расписании эти изменения не отображаются. Поэтому, если уже выбран день поездки, желательнее искать актуальные рейсы на конкретную дату.

Ну и, наконец, обещанные «плюшки». Во-первых, как и было обещано, приложение запоминает выбранные рейсы. Во-вторых, результаты последнего поиска также сохраняются в памяти телефона и доступны оффлайн в пункте меню «Избранное».

В-третьих, разработанное приложение позволяет искать рейсы еще и в `VlaBlaCar`. `VlaBlaCar` – это достаточно крупный интернет-сервис для поиска попутчиков, площадка, объединяющая водителей и пассажиров, которым по пути. Результаты поиска автобусных рейсов и поездок в `VlaBlaCar` объединяются и упорядочиваются по времени. Интерфейс результатов поиска представлен на рис. 3:

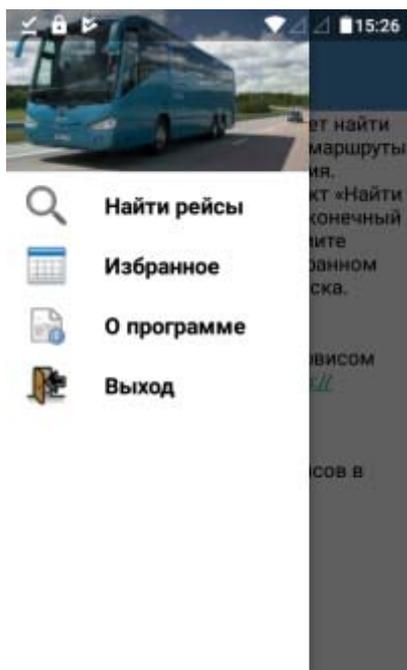


Рис. 1 – Главное меню приложения

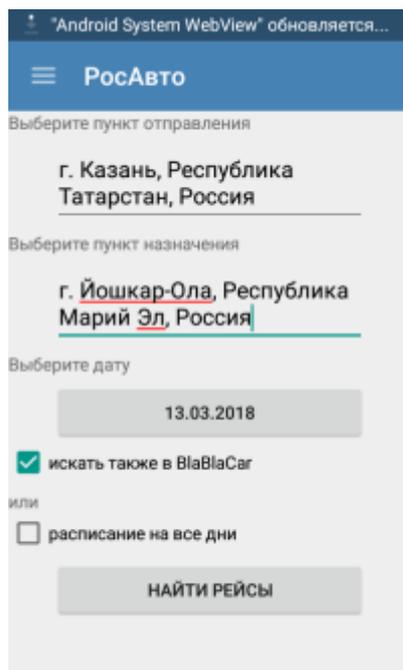


Рис. 2 – Меню поиска в разработанном приложении

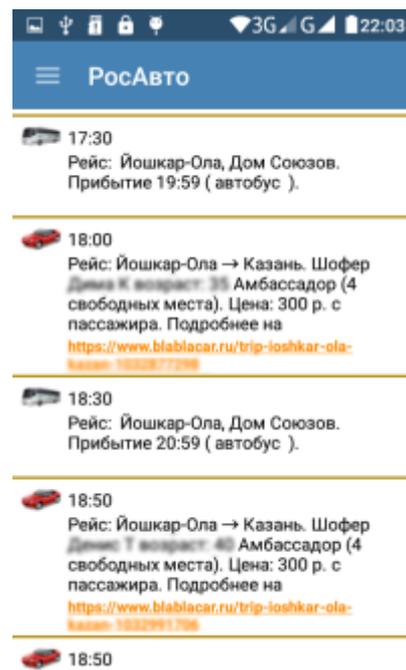


Рис. 3 – Отображение результатов поиска

Поиск рейсов в BlaBlaCar программируется совершенно аналогичным образом. Для отображения списка рейсов используется виджет ListView, который представляет собой представляет прокручиваемый список элементов [10].

Для успешной работы приложения нужно не забыть внести изменения в файл AndroidManifest.xml. Нам необходимо разрешение на доступ в интернет для того, чтобы получать информацию сайтов [11]:

```
<uses-permission android:name="android.permission.INTERNET" />
```

Процесс загрузки расписания рейсов происходит в отдельном потоке, который создается с помощью класса AsyncTask. Связано это с тем, что в Android нельзя выполнять тяжелые задачи в основном потоке. Подробнее

про класс AsyncTask и примеры работы с ним можно прочитать, например, в [12].

Ознакомиться с разработанным приложением можно бесплатно по адресу <https://play.google.com/store/apps/details?id=com.vildanov.rusauto> в Google Play.

Заключение

Львиная доля разрабатываемых сегодня мобильных приложений доставляет необходимую информацию пользователю с помощью сети Интернет, с помощью запросов к серверам за необходимой информацией. Рассмотренные в работе алгоритмы и методология разработки приложения для поиска междугородних автобусных пассажирских рейсов и попутных автомашин с помощью API, предоставляемого компанией Яндекс, могут быть востребованы для разработки подобных мобильных приложений.

Конечно, крупные автостанции имеют свой портал, где можно ознакомиться с рейсами и даже купить билет. Яндекс расписание удобно тем, что небольшие частные компании, занимающиеся пассажирскими перевозками, часто не имеющие своего сайта, вполне могут включить сюда расписание своих рейсов. Тем самым Яндекс выступает агрегатором информации о поставщиках перевозок и может дать более полноценную информацию о рейсах, чем любая другая, отдельно взятая компания на рынке перевозок.

К недостатку приложения (и сервиса Яндекс.расписания) можно отнести то, что расписание, полученное с помощью API, может быть не всегда актуальным. Например, рейс может быть отменен из-за попадания на праздничный день, в связи с неблагоприятными погодными условиями, и т.п. Для того, чтобы расписание было всегда актуальным, нужно, чтобы автостанции тоже использовали API Яндекс в своих приложениях. В свою очередь, для этого Яндекс требуется разработать дополнительное удобное



API для автовокзалов и диспетчеров автостанций (для покупки билета, для указания количества свободных мест, и т.д.), что является довольно нетривиальной задачей.

Для решения этой задачи Яндекс запустил в 2017 г. новый сервис – Яндекс.Автобусы. На Яндекс.Автобусах уже можно не только выбрать, но и купить билеты на автобусные рейсы по России и странам СНГ. Количество направлений постоянно увеличивается, но, конечно, по масштабам всей страны охват небольшой. Будет ли сервис успешным и развиваться, покажет время.

Литература

1. Сафронов Р.Ю. Разработка мобильного приложения для ИОs с использованием технологии «Ibeacon» для решения задач навигации внутри зданий на основе беспроводных сетей // Инженерный вестник Дона, 2014, №4 URL: ivdon.ru/ru/magazine/archive/N4y2014/2726/.

2. Винокуров А.С., Баженов Р.И. Проект разработки мобильного приложения для кафе быстрого питания // Постулат. 2016. №1. – С. 5. – URL: e-postulat.ru/index.php/Postulat/article/view/39/42.

3. Марченко И. О., Слугина А. Е. Система оповещения в случае дорожно-транспортного происшествия // Инженерный вестник Дона, 2016, №2 URL: ivdon.ru/ru/magazine/archive/n2y2016/3601/.

4. Салтыков Сергей. Сайт-агрегатор. URL: seotuition.ru/Поисковоепродвижение/ОбучениеSEO/Сайтагрегатор.aspx (дата обращения: 03.11.2017).

5. Вильданов А.Н. Парсинг HTML-документа на языке Java (на примере расписания НФ БашГУ) // Достижения и приложения современной информатики, математики и физики: материалы V Всероссийской научно-практической заочной конференции. – г.Нефтекамск, 2016. – С. 17-25. – ISBN 978-5-7477-4280-2.



6. Орещенков Игорь. Извлечение информации из HTML-страниц средствами PHP // Системный администратор. № 1-2 (182-183). Издательский дом «Положевец и партнеры», Москва, 2018. – С. 60-64.

7. Jonathan Hedley. Jsoup Java HTML Parser, with best of DOM, CSS, and jquery. URL: jsoup.org (date of access: 01.11.2017).

8. Аксенов К.В. Обзор современных средств для разработки мобильных приложений // Новые информационные технологии в автоматизированных системах. № 17. Московский институт электроники и математики НИУ ВШЭ, Москва, 2014. – С. 508-513.

9. Suvojit Kar. [suvojitkar/android](https://github.com/suvojitkar/android). URL: github.com/suvojitkar/android (date of access: 11.11.2017).

10. Климов, Александр. Android: ListView. URL: developer.alexanderklimov.ru/android/views/listview.php (дата обращения: 01.11.2017).

11. Сень А.С. Создание новостного агрегатора на Android // XIX Всероссийская студенческая науч.-практ. конф. Нижневартковск. гос. ун-та : сб. ст.– Нижневартковск, 2017. – С. 461-463.

12. AsyncTask. Знакомство, несложный пример // STARTANDROID.RU. URL: startandroid.ru/ru/uroki/vse-uroki-spiskom/149-urok-86-async-task-znakomstvo-neslozhnyj-primer.html (дата обращения: 01.11.2017).

References

1. Safronov R.YU. Inzhenernyj vestnik Dona (Rus). 2014. №4 URL: ivdon.ru/ru/magazine/archive/N4y2014/2726.

2. Vinokurov A.S., Bazhenov R.I. Postulat. 2016. №1. pp. 5. URL: e-postulat.ru/index.php/Postulat/article/view/39/42.

3. Marchenko I. O., Slugina A. E. Inzhenernyj vestnik Dona (Rus). 2016. №2 URL: ivdon.ru/ru/magazine/archive/n2y2016/3601.



4. Saltykov Sergej. Sajt-agregator [site aggregator]. URL: seotuition.ru/Поисковоепродвижение/ОбучениеSEO/Сайтагрегатор.aspx.

5. Vil'danov A.N. Dostizheniya i prilozheniya sovremennoj informatiki, matematiki i fiziki: materialy V Vserossijskoj nauchno-prakticheskoj zaochnoj konferencii [Proc. All-Russian Computer Symp. «Achievements and applications of modern computer science, mathematics and physics»]. g.Neftekamsk, 2016. pp. 17-25. ISBN 978-5-7477-4280-2.

6. Oreshchenkov Igor'. Sistemnyj administrator. № 1-2 (182-183). Izdatel'skij dom «Polozhevec i partnery», Moskva, 2018. pp. 60-64.

7. Jonathan Hedley. Jsoup Java HTML Parser. URL: jsoup.org.

8. Aksenov K.V. Novye informacionnye tekhnologii v avtomatizirovannyh sistemah. № 17. Moskovskij institut ehlektroniki i matematiki NIU VSHEH, Moskva, 2014. pp. 508-513.

9. suvojit kar. URL: github.com/suvojitkar/android.

10. Aleksandr Klimov. Android: ListView. URL: developer.alexanderklimov.ru/android/views/listview.php.

11. Sen' A.S. XIX Vserossijskaya studencheskaya nauch.-prakt. konf. Nizhnevartovsk. gos. un-ta: sb. st. [Proc. All-Russian Student Scientific-Practical Symp. of Nizhnevartovsk University]. Nizhnevartovsk, 2017. pp. 461-463.

12. AsyncTask. STARTANDROID.RU. URL: startandroid.ru/ru/uroki/vse-uroki-spiskom/149-urok-86-asynctask-znakomstvo-neslozhnyj-primer.html.