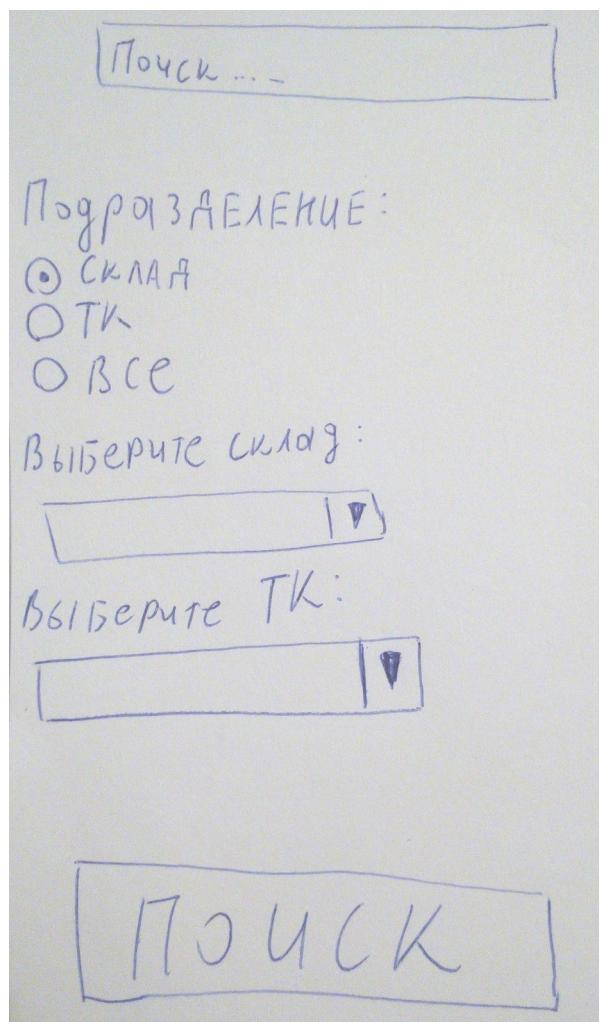
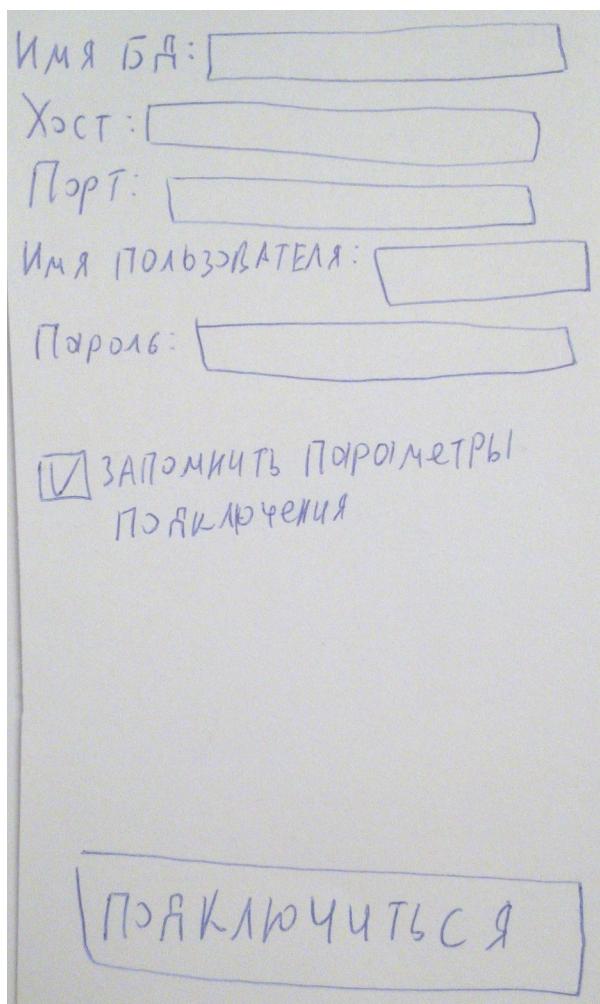


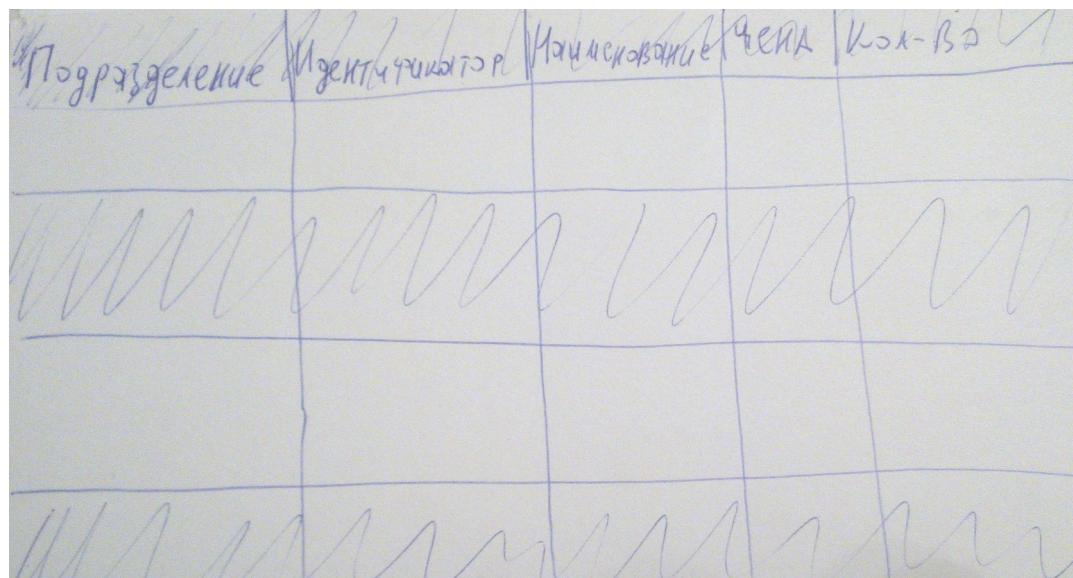
LightSearch Android

[07.09.2018]

Итак, создав клиента для ПК, я решил создать клиента для Android. До этого я знал, что для Android существует Android Studio, и имел небольшое представление о том, как пишутся приложения для этой операционной системы: где-то год назад я написал приложение Sudoku Solution — простенькая программа, которая решала за пользователя судоку. Но теперь мне предстояло написать программу на другом уровне, и я приступил.

Мощности моего ноутбука не хватило для Android Studio, поэтому я установил ее на свой ПК и стал изучать. Через несколько дней у меня уже получился примерный экран авторизации и экран поиска товара. Я себе представлял программу так:

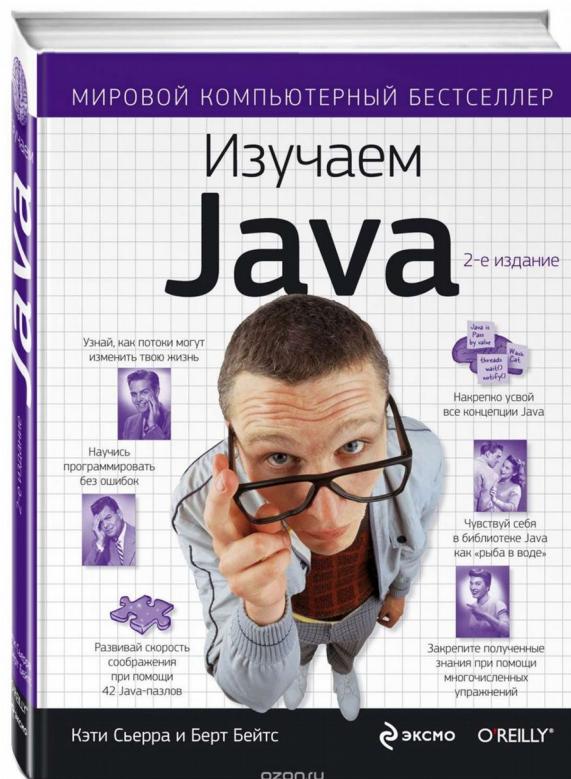




На первой картинке экран авторизации, на второй — экран поиска, на третьей — экран вывода результата.

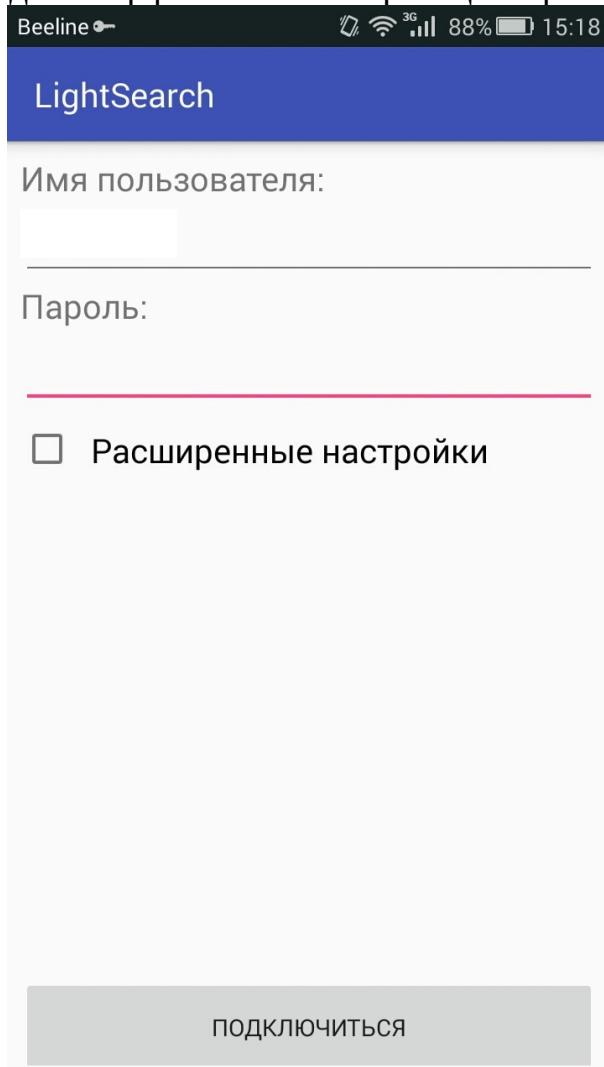
И вот, сделав такой интерфейс в своем проекте, я подключил JDBC драйвер в мой проект через Gradle — и это была моя первая ошибка. В Android используется немного другая версия Java, как я выяснил позднее, и не поддерживает JDBC-драйверы. И тогда я стал в тупик. Я стал искать Jaybird на Android — и нашел какой-то проект, который занимался этой задачей, но он давно не обновлялся и работал очень странно. Поэтому я стал думать, как можно решить данную проблему.

И я вспомнил о том, что мой наставник по языку Java (да, у меня такой есть, и я очень рад этому :)) посоветовал мне книгу как старт для программирования на этом языке: «Изучаем Java» авторов Кэти Съерра и Берт Бейтс:



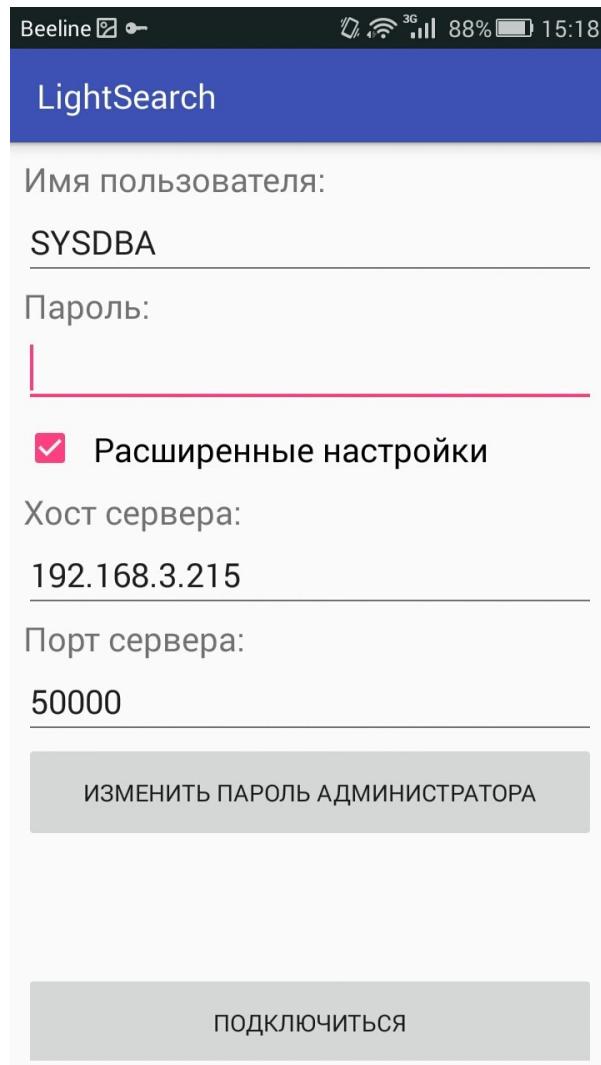
И он мне говорил, что там очень хорошо написано про сокеты. И тогда меня осенило: можно сделать сервер на Java, который будет общаться с моим Android-клиентом, и делать всю работу, связанную с бд! Сомнений было много: а заработает ли это? Как мне подключаться к серверу? Все ли получится? Но глаза боялись, а руки стали уже делать — конечно же, сначала я приступил к написанию сервера. Описание того, как я делал сервер можно прочитать в документе LightSearchServer ([ссылка](#)).

И вот, сервер более-менее написан, ушло примерно недели две, и теперь можно приступить к Android-клиенту. Первое, что я сделал — изменил интерфейс окна авторизации. Я понял, что клиенту ни к чему знать имя бд, хост, порт — это должен знать сервер. Что необходимо знать пользователю — это имя пользователя и пароль. Но мой руководитель практики подсказал мне, что можно сделать функцию, открывающую дополнительные настройки (хост и порт сервера), и доступ к ней сделать через пароль, который будет знать администратор. И тогда интерфейс окна авторизации принял следующий вид:



Флажок «Сохранить параметры подключения» я также убрал, так как нашел более лучшую альтернативу — использовать SharedPreferences. Очень удобно и довольно безопасно. Из параметров подключения пользователя я храню в ней лишь имя пользователя.

Потом я подумал, как можно сделать расширенные настройки администратора, и сделал их через флажок, придумав следующий механизм: при первой инициализации приложение попросит ввести пароль администратора. Потом она запомнит его SHA256-хэш в Shared Preferences. И теперь при нажатии на данный флажок будет выскакивать окно с вводом пароля администратора. После этого откроются дополнительные настройки:



Как видно на скриншоте выше, открываются не только поля хоста и порта сервера: еще и добавляется кнопка «Изменить пароль администратора». В первый раз я ее забыл добавить, но потом вспомнил о ней :)

В поле хост сервера я поставил фильтр, чтобы можно было вводить только строку в виде ip-адреса. Для этого я добавил в xml-файл макета окна авторизации в элементе поля ввода хоста сервера следующий код:

```
android:inputType="number"  
android:digits="0123456789."
```

И все заработало!

Для порта добавил следующий код:

```
android:inputType="number"
```

Этого вполне достаточно, ведь этими функциями будет пользоваться только администратор, и дополнительная фильтрация по моему мнению не нужна.

Теперь я приступил к окну поиска. И тут я столкнулся с проблемами. Первая проблема заключалась в том, что Android по умолчанию запрещает для приложения доступ в интернет. Я прочитал в интернете по поводу этой проблемы, и она решается довольно легко: необходимо лишь добавить в манифест приложения вот такую строчку:

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

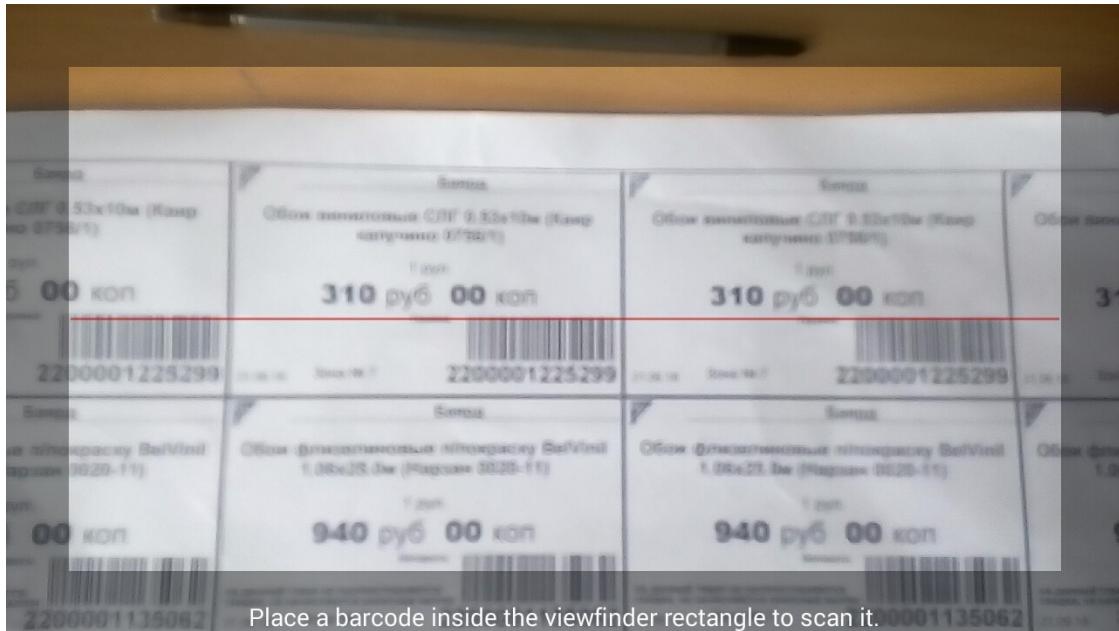
Теперь проблема касалось того, что Android не позволяет работать в одном потоке UI и сокеты — для сокетов нужен отдельный поток. Хорошо, сделал отдельный поток. Но вот снова незадача: программа ругается на то, что я использую объекты UI в этом потоке, а ведь мне надо как-то вывести ошибку на экран, если что-то случится — нет соединения, неверное имя пользователя или пароль и т.д. И я снова пошел в интернет за решением данного вопроса, и там посоветовали два решения — через асинхронные задачи или через метод runOnUiThread. Я выбрал второй вариант по причине того, что он мне был более понятен.

И вот: все написано, можно тестировать. Тестировал на своем смартфоне. Подключился к VPN компании через свой смартфон, чтобы можно было подключиться к серверу. Конечно, первый тест не прошел успешно, как и второй, и третий, и четвертый: было много мелких ошибок и недочетов, и я их исправлял по мере их «всплыивания». Больше всего конечно потратил времени на устранения недочетов между взаимодействием клиента и сервера, ведь как я описывал в документе про сервер([ссылка](#)), все построено по принципу «команда — ответ», и бывало такое, что команду не так написал, или JSON не так построил. И на третий экран, где отображается таблица, тоже потратил достаточно времени — изучал материалы по созданию динамических столбцов в Android, создание заголовочных столбцов, выравнивание таблицы, чтобы информация в ячейках читалась удобно, ну и чтобы таблица выглядела аккуратно. В итоге у меня получилось, но информация не вся помещалась в ячейки. И тогда я придумал следующую штуку: если нажать на строчку, то высветится окошко, в котором будет подробно написана информация.

Реализовать было ее не так проблематично, и все работало отлично.

Я показал программу своему руководителю практики — и он сказал, что в данном клиенте нет необходимости искать по наименованию или по части наименования, а достаточно лишь искать по штрих-коду. И дал мне еще одно задание: реализовать считывание штрих-кода через камеру смартфона.

В интернете я нашел бесплатную библиотеку Zxing. Подключил ее к своему проекту, посмотрел, как его активировать, добавил кнопку «Считать штрихкод» на экран поиска, привязал к ней обработчик — и вот результат:



Все работало! Ах, да, чуть не забыл: для поля поиска теперь нужно добавить условие, что можно вводить только цифры, и делать проверку на то, чтобы их было введено не менее пяти.

Теперь окно поиска стало выглядеть так:

Beeline 3G 88% 15:20

LightSearch

Поиск:

204404245290

СКАНИРОВАТЬ ШТРИХКОД

Склады

ТК

Все

Выберите склад:

<Все>

Выберите ТК:

<Все>

ПОИСК

А экран таблица — вот так:

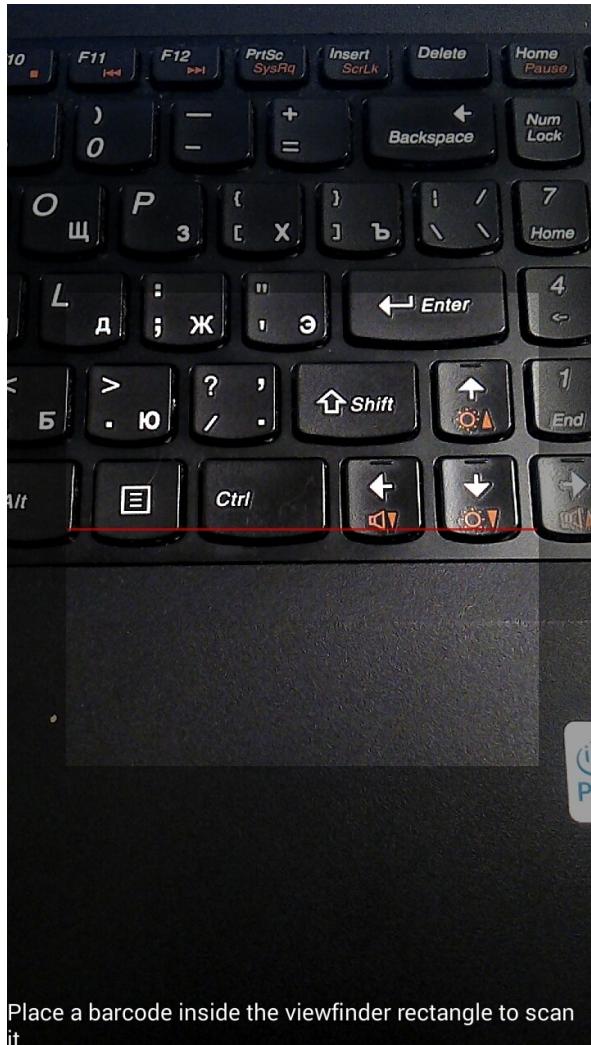
Подразделение	Идентификатор	Наименование	Цена	Кол-во
Склад Комплекс 6	1135062	Обои флизелиновые п/ покраску BelVinil 1.06x25.0м (Нарзан 0020-11)	940.0	13.0
Склад Комплекс 4	1135062	Обои флизелиновые п/ покраску BelVinil 1.06x25.0м (Нарзан 0020-11)	940.0	1.0
Склад 200	1135062	Обои флизелиновые п/ покраску BelVinil 1.06x25.0м (Нарзан 0020-11)	940.0	0.0

Если нажать на строчку — то получаем вот такой результат:

Подразделение	Идентификатор	Наименование	Цена	Кол-во
Склад Комплекс 6	1135062	Обои флизелиновые п/ покраску BelVinil 1.06x25.0м (Нарзан 0020-11)	940.0	13.0
Склад Комплекс 4	1135062	Обои флизелиновые п/ покраску BelVinil 1.06x25.0м (Нарзан 0020-11)	940.0	1.0
Склад 200	1135062	Обои флизелиновые п/ покраску BelVinil 1.06x25.0м (Нарзан 0020-11)	940.0	0.0

Руководитель практики подсказал очень важную функцию, после того, как я показал ему еще раз программу: возможность использовать сканер в портретном режиме. По умолчанию доступен только ландшафтный режим.

Тогда я стал искать решение, конечно же, в интернете. И нашел! Правда, пришлось искать довольно долго: решение было предоставлено по сути кусками, и пришлось их складывать друг с другом, чтобы получить результат:



Портретный режим работает!

Руководитель практики скачал себе на смартфон мое приложение для тестов. И он сделал еще одно очень важное замечание: при подключении, если не включен VPN, не включен сервер, или если плохая связь, то программа бесконечно висит при подключении: бесконечное окошко с надписью «Подключение» и бегущими синими шариками... Тогда он мне посоветовал сделать так: в цикле несколько раз пытаться подключиться к серверу, например 3 раза по 10 секунд. Если все попытки исчерпаны, то высвечивать окошко с информацией, что подключение не установлено.

Когда я сел писать эту функцию, то у меня выключили свет, и дали его примерно через несколько часов. Я включаю свой ПК, открываю Android Studio, пишу функцию, жму на зеленый треугольник — и Android Studio не собирает приложение. Какая-то ошибка во внутреннем JSON-файле самой студии! Я не нашел ни одного внятного решения, кроме как одного: удалить проект и все с начала написать. Конечно ситуация из неприятных, но выбора не было:

пришлось копировать код из старого проекта в новый. После примерно получаса у меня уже был «старый-новый» проект, и он заработал.

После этого я уже не так активно занимался этим проектом — практика уже давно подошла к концу, было начало августа, и я только исправлял мелкие недочеты, когда они всплывали. Но зато потом, мой руководитель практики, который сейчас является уже руководителем проекта для меня, показал данное приложение генеральному директору компании «Баярд» — ему понравилось данное приложение, и он хотел бы, чтобы я и дальше развивал его! В приложении нужна еще одна функция, помимо поиска — создание мягкого чека. Это задание довольно тяжелое, но не менее интересное. Для его выполнения необходимо вынести всю бизнес-логику за сервер, чтобы весь комплекс приложений LightSearch был независим от какого-либо предприятия. Всю часть, связанную с сервером, я описал в соответствующем документе ([ссылка](#)). Здесь же будет вся информация, связанная с Android-клиентом.

Схема, которую я нарисовал для общего представления о проекте:



Как видно, схема не предполагала принципа «команда-ответ». Теперь insert в базу сервер делать не будет, а будет передавать команде базе, и сама база будет делать insert, и затем давать ответ серверу. Также будет произведен перенос select`ов из сервера в бд.

Предположительный интерфейс программы (на следующей странице):

АВТОРИЗАЦИЯ

Имя пользователя:

Пароль:

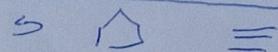
Расширенные настройки

Хост сервера:

Порт сервера:

[Изменить пароль администратора]

[Подключиться]



Мягкий чек

Поиск:

[СЧИТАТЬ ШТРИХ-КОД]

ШТРИХ-КОД (коинкарование)	
~~~~~	<input type="radio"/> 15 (⊕)
~~~~~	<input type="radio"/> 1 (⊕)
~~~~~	<input type="radio"/> 9 (⊕)

[ЗАКРЫТЬ МЯГКИЙ ЧЕК]



## Поиск товара

Поиск:

[СКАНИРОВАТЬ ШТРИХ-КОД]

- Склады
- ТК
- Все

Выберите склад:  
<Все>

Выберите ТК:  
<Все>

[Поиск]

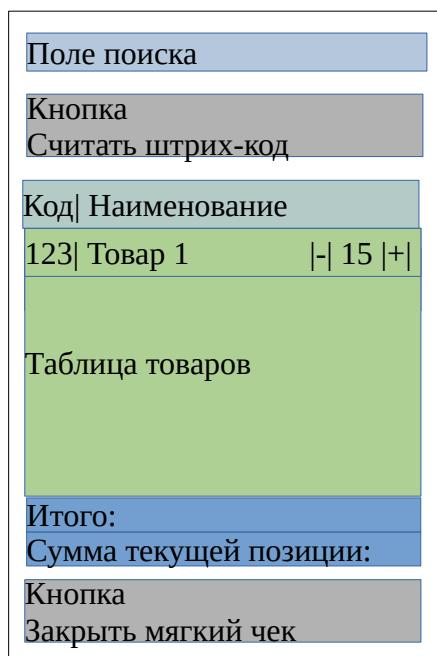


При подключении происходит переход из окна авторизации в окна «Мягкий чек» и «Поиск товара», переход между которыми осуществляется через свайп экрана вправо или влево. По умолчанию экран «Мягкий чек» является первым.

Сценарии мягкого чека:

1) Пользователь нажимает на кнопку «Создать мягкий чек», приложение открывает экран формирования мягкого чека. Пользователь в поле ввода штрих-кода вводит штрих-код товара, или считывает его при помощи камеры смартфона, нажав на кнопку «Считать штрих-код». В таблице товаров добавляется новая запись с соответствующим товаром. В строке отображается штрих-код и наименование товара. Справа от этой информации расположены кнопки «-» и «+», которыми можно регулировать количество товара. Между этими кнопками отображается количество товара. Можно написать количество вручную, нажав на количество товара. При нажатии на строку в поле «Сумма текущей позиции» отображается сумма выбранной в таблице позиции. В поле «Итого» отображается сумма всех позиций в таблице. Пользователь нажимает кнопку «Закрыть мягкий чек», чек передается на сервер, сервер передает чек в базу данных, база данных передает ответ серверу, сервер передает ответ приложению, приложение отображает на экране полученный ответ.

Схема интерфейса сценария 1:

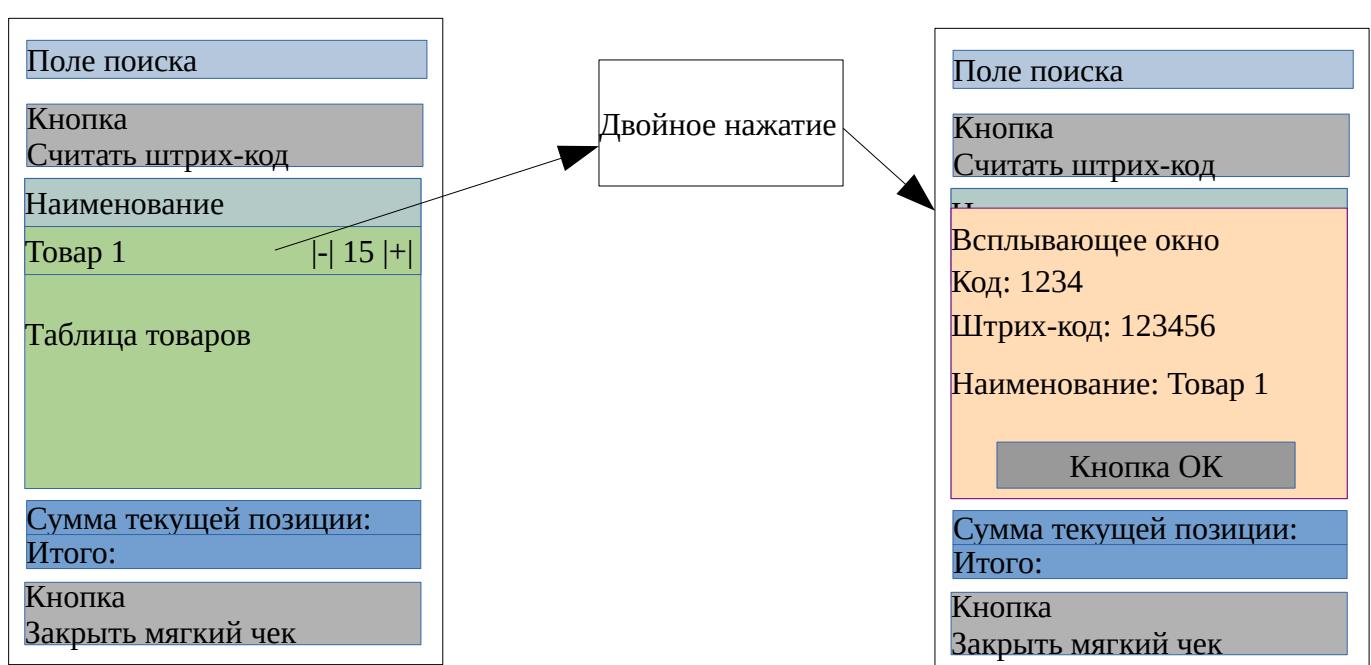


Минус данного подхода: так как экран смартфона ограничен, то поля «Код» и «Наименование» не будут полностью отображаться на экране. Переносить на новую строку символы, которые не вместились на экран, нельзя, так как информация в таблице станет неудобной для восприятия.

2) Пользователь нажимает на кнопку «Создать мягкий чек», приложение открывает экран формирования мягкого чека. Пользователь в поле ввода штрих-кода вводит штрих-код товара, или считывает его при помощи камеры

смартфона, нажав на кнопку «Считать штрих-код». В таблице товаров добавляется новая запись с соответствующим товаром. В строке отображается наименование товара. Справа от этой информации расположены кнопки «-» и «+», которыми можно регулировать количество товара. Между этими кнопками отображается количество товара. Можно написать количество вручную, нажав на количество товара. При нажатии на строку в поле «Сумма текущей позиции» отображается сумма выбранной в таблице позиции. При двойном нажатии на строку в таблице открывается всплывающее окно, в котором написаны полностью наименование, код, и штрих-код товара. В поле «Итого» отображается сумма всех позиций в таблице. Пользователь нажимает кнопку «Закрыть мягкий чек», чек передается на сервер, сервер передает чек в базу данных, база данных передает ответ серверу, сервер передает ответ приложению, приложение отображает на экране полученный ответ.

Схема интерфейса сценария 2:

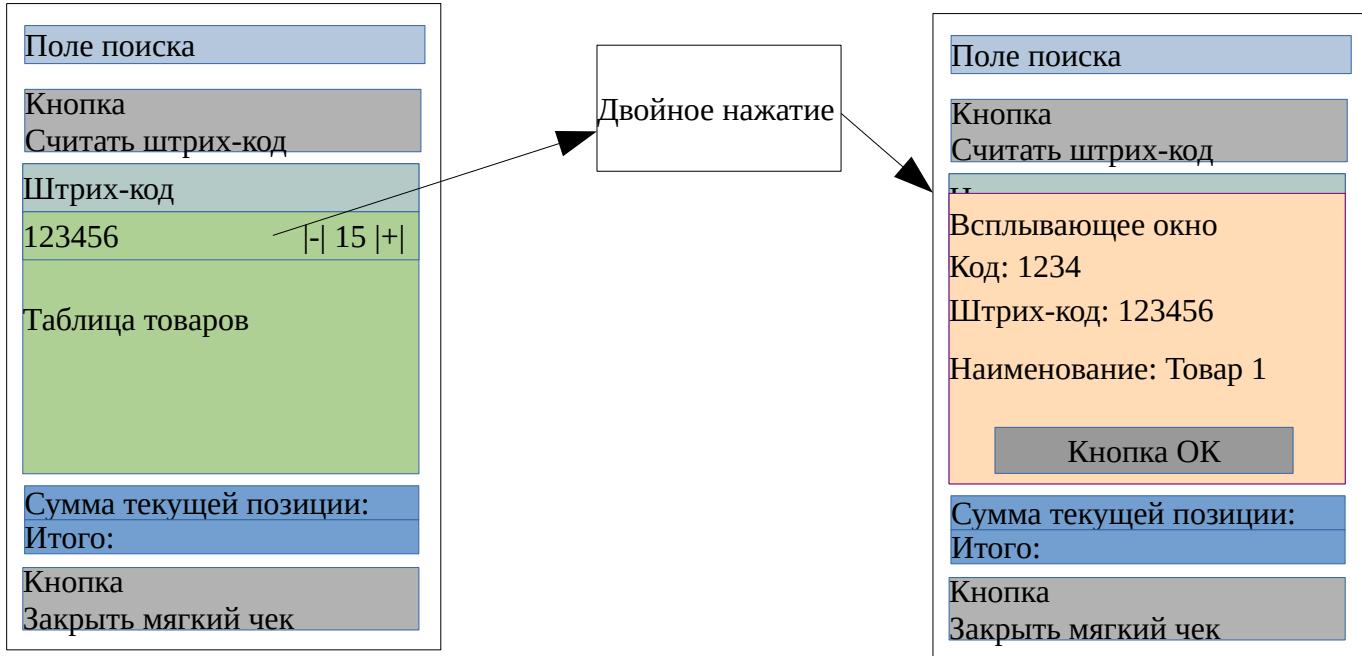


При таком подходе в строке таблицы отображается больше информации, а в всплывающем окне будет отображена полная информация о товаре. Также в всплывающем окне допустимо использовать перенос строки — это не будет неудобно, как в первом подходе.

3) Пользователь нажимает на кнопку «Создать мягкий чек», приложение открывает экран формирования мягкого чека. Пользователь в поле ввода штрих-кода вводит штрих-код товара, или считывает его при помощи камеры смартфона, нажав на кнопку «Считать штрих-код». В таблице товаров добавляется новая запись с соответствующим товаром. В строке отображается штрих-код товара. Справа от этой информации расположены кнопки «-» и «+», которыми можно регулировать количество товара. Между этими кнопками отображается количество товара. Можно написать количество вручную, нажав

на количество товара. При нажатии на строку в поле «Сумма текущей позиции» отображается сумма выбранной в таблице позиции. При двойном нажатии на строку в таблице открывается всплывающее окно, в котором написаны полностью наименование, код, и штрих-код товара. В поле «Итого» отображается сумма всех позиций в таблице. Пользователь нажимает кнопку «Закрыть мягкий чек», чек передается на сервер, сервер передает чек в базу данных, база данных передает ответ серверу, сервер передает ответ приложению, приложение отображает на экране полученный ответ.

Схема интерфейса сценария 2:

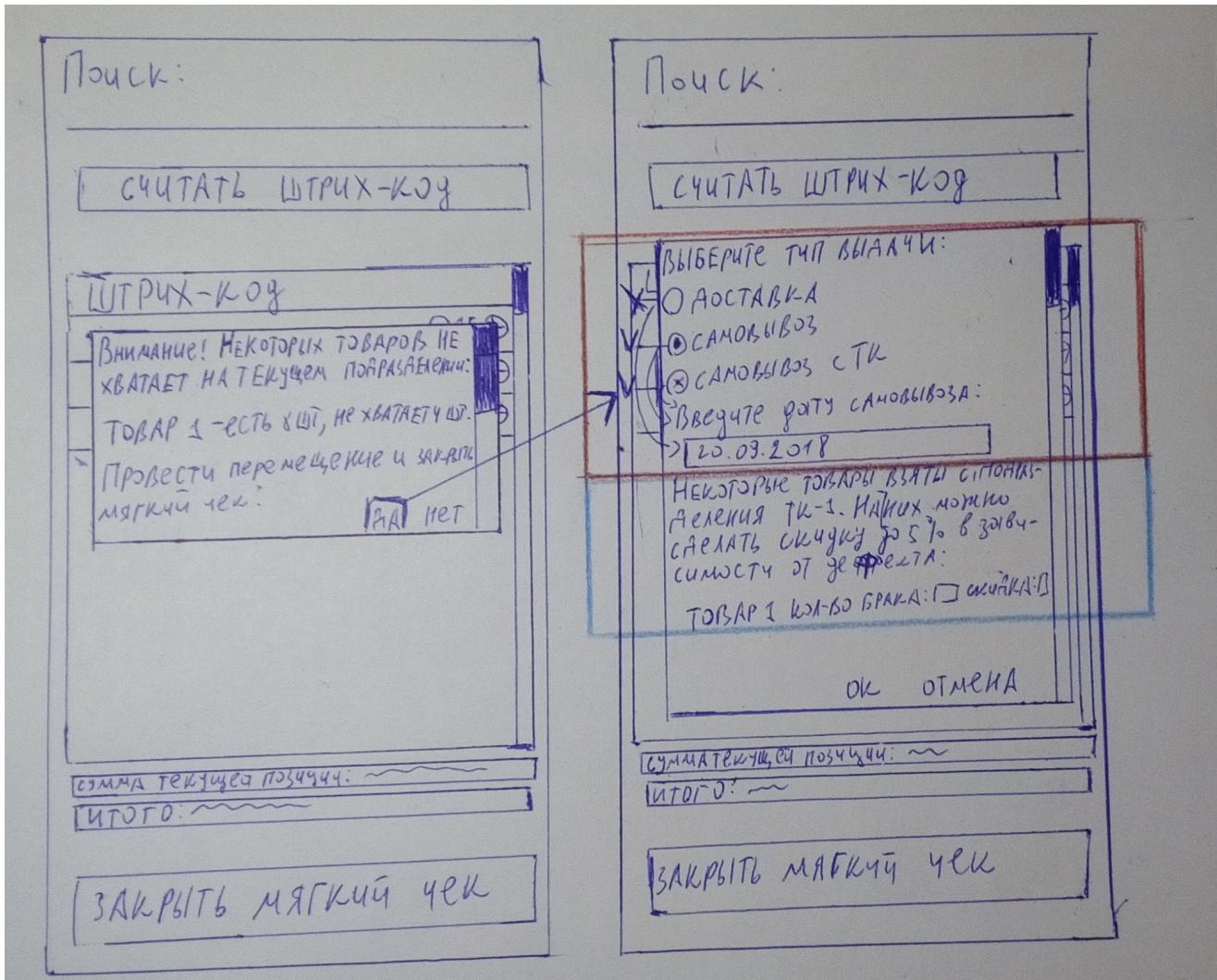


Замечание, касаемо всех подходов: можно менять количество товаров четырьмя способами:

- 1) Нажатием кнопок «-» и «+»;
- 2) Нажатием на количество товара. В этом случае откроется цифровая клавиатура;
- 3) Считать товар при помощи сканера, который уже добавлен в таблицу;
- 4) Написать штрих-код товара, который уже добавлен в таблицу.

[19.09.2018]

Сегодня пришло техническое задание. После его переработки я нарисовал, как будет выглядеть всплывающие окна после нажатия на кнопку «Закрыть мягкий чек»:



Из-за того, что во всплывающем окне будет довольно много элементов, необходимо добавить вертикальную прокрутку. Первое всплывающее окно появляется только тогда, когда товара не хватает в текущем подразделении. Тогда необходимо провести перемещение. Перемещение полностью автоматизировано на предприятии, поэтому клиент просто будет посыпать команду серверу, что необходимо провести перемещение.

При нажатии на кнопку «Да» в первом окне открывается еще одно всплывающее окно, в котором необходимо указать тип выдачи. Если тип выдачи — самовывоз и самовывоз с ТК, то необходимо указать дату самовывоза. Если перемещение необходимо, то может возникнуть ситуация, когда товар берется с какого-то склада, но он с дефектами, и поэтому на него можно сделать скидку. В этом случае необходимо указать количество дефектного товара данной позиции (на картинке я написал кол-во брака только для того, чтобы все слова

вместились в одну строчку :). После нажатия на кнопку «OK» мягкий чек закрывается.

Если перемещение не потребуется, то будет выведено только то, что выделено на картинке красным, и кнопки «OK» и «Отмена».

Если перемещение требуется, то будет выведено то, что выделено и красным, и синим, и кнопки «OK» и «Отмена».