УДК 004.415.25+004.42+004.514

АЛГОРИТМ ГЕНЕРАЦИИ КОДА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ НА ОСНОВЕ ДАННЫХ ГРАФИЧЕСКОГО РЕДАКТОРА

А. Ю. Усачёв

Высшая школа информационных технологий и интеллектуальных систем Казанского (Приволжского) федерального университета

usacheow.ar@gmail.com

Аннотация

Работа посвящена разработке алгоритма генерации кода пользовательских интерфейсов нативных Android-приложений на основе данных графического редактора. Рассмотрена проблема негативного воздействия на время разработки продукта выполнения шаблонных действий и предложен программный инструмент для решения данной проблемы.

Ключевые слова: *пользовательский интерфейс, графический редактор, алгоритм генерации, мобильные приложения*

ВВЕДЕНИЕ

Длительность процесса разработки мобильных приложений напрямую влияет на сроки разработки и итоговую стоимость конечного продукта. Ниже приведены работы, выполнение которых занимает наибольшее время [1]:

- анализ требований и формулирование технического задания;
- разработка дизайна и пользовательского интерфейса;
- вёрстка макетов пользовательского интерфейса;
- написание кода и тестирование.

Автоматизация этапов жизненного цикла разработки мобильных приложений, требующих выполнения шаблонных действий, позволит значительно сократить затраты на создание программного продукта.

На текущий момент процесс создания пользовательского интерфейса требует написания программного кода для каждого элемента интерфейса по предоставленному дизайну экранов разрабатываемого мобильного приложе-

ния. Согласно официальному сайту программы Sketch [2], большинство мобильных дизайнеров выполняет свою работу в данном графическом редакторе. Файл программного инструмента Skecth может быть представлен в виде набора JSON-файлов, в которых содержится вся информация о дизайне экранов приложения, что делает возможной автоматизацию данного процесса путем анализа и обработки полученных JSON-файлов.

Разработка программного инструмента, автоматизирующего процесс разработки пользовательского интерфейса, сопровождается решением следующих задач:

- определение перечня элементов, которые поддаются генерации программой, и разработка требований к структуре sketch-проекта;
- разработка алгоритма для генерации кода пользовательского интерфейса на основе данных графического редактора;
- разработка графического интерфейса для взаимодействия пользователя с программой.

АНАЛИЗ ИЗВЕСТНЫХ РЕШЕНИЙ ПРОБЛЕМЫ

На данный момент есть два программных инструмента, способных на генерацию элементов интерфейса.

Первый инструмент носит название Zeplin, он является связующим звеном между дизайнером и программистом, позволяя импортировать sketch-проект для представления его в таком виде, который позволит сократить время разработчика на взаимодействие с данным проектом [3]. Кроме того, Zeplin поддерживает автоматическую генерацию одного элемента интерфейса – текстового поля, в то время как пользовательский интерфейс может состоять из множества различных элементов.

Вторым инструментом является продукт под названием Supernova [4], к плюсам которого можно отнести интуитивно понятный пользовательский интерфейс. Однако все элементы, генерируемые данной программой, состоят исключительно из текстового поля и картинки. Отсюда вытекает тот факт, что данное решение подходит только для генерации макетов в тех случаях, когда требуется представить макеты пользовательских интерфейсов на мобильном устройстве, а для дальнейшей работы они не подходят.

Следовательно, можно сделать вывод, что программного инструмента, который обладал бы возможностью генерации верстки макета пользовательского интерфейса в полном объёме, не существует.

ФОРМУЛИРОВАНИЕ ТРЕБОВАНИЙ К РЕШЕНИЮ

При составлении технических требований было решено определить начальную группу элементов интерфейса android-приложений, которые будут определяться на экране интерфейса и на основе которых будет составляться файл макета в формате xml, требуемом для использования в android-проектах. Таким образом, было решено взять основные и наиболее часто используемые элементы [5]:

- *Toolbar* – это панель, расположенная в верхней части экрана, на которой отображается заголовок данного экрана либо логотип приложения и кнопки действий, например, открытие меню, переход назад или какие-либо специфичные для данного экрана действия (Рис. 1);

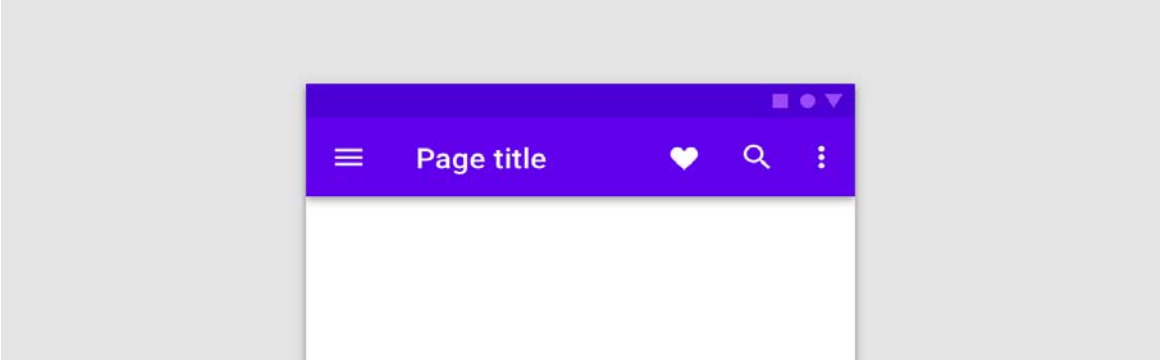

Рис. 1. Toolbar

- *TabLayout* – это элемент, на котором расположены вкладки данного экрана, которые могут быть представлены в виде текста и/или иконки (Рис. 2);
- *AppBarLayout* – элемент интерфейса, схожий с LinearLayout, использующийся для задания поведения дочерних элементов при прокручивании экрана указанием дополнительных флагов;
- *ScrollView* – элемент, использующийся для прокрутки содержимого. Может иметь только один дочерний элемент;
- *LinearLayout* – это диспетчер компоновки, который позволяет располагать внутри себя элементы последовательно друг за другом (вертикально либо горизонтально), указав соответствующий флаг в атрибутах (Рис. 3);

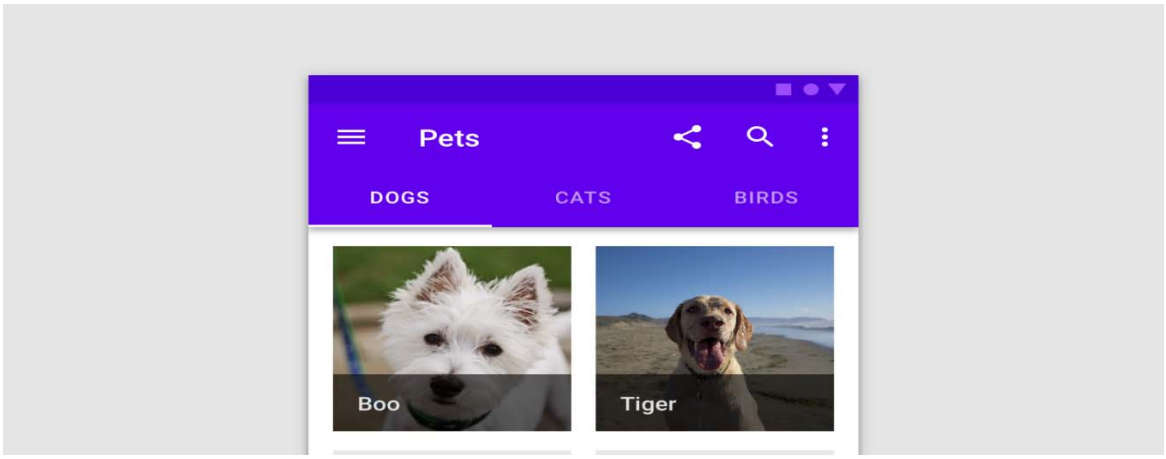

Рис. 2. TabLayout

Рис. 3. LinearLayout: горизонтальный и вертикальный

- *CardView* – элемент, предназначенный для отображения содержимого в виде карточки с заданной высотой и радиусом закругления углов (Рис. 4);

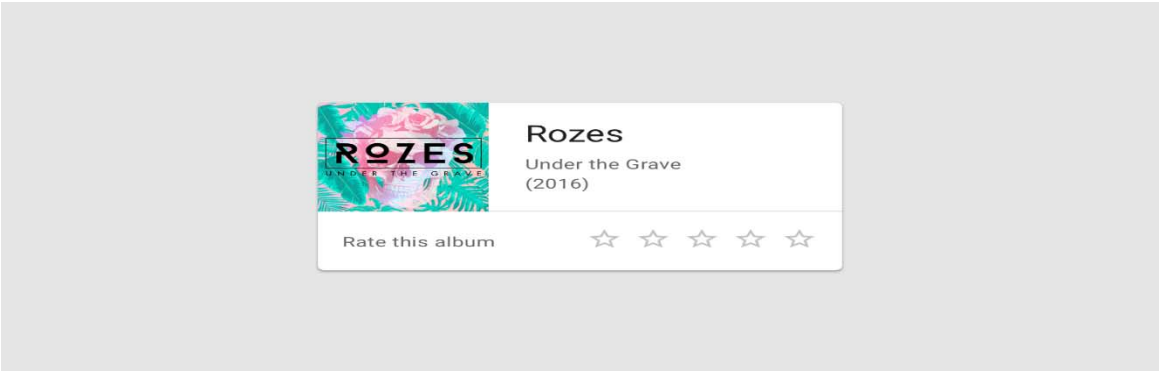

Рис. 4. CardView

- RecyclerView – это элемент, который служит для отображения списка данных (Рис. 5). Может быть вертикальным или горизонтальным, что определяется соответствующим значением в атрибутах. Элементы могут быть различны-

ми, благодаря чему можно разделить список на группы, отображая заголовок для каждой из них;

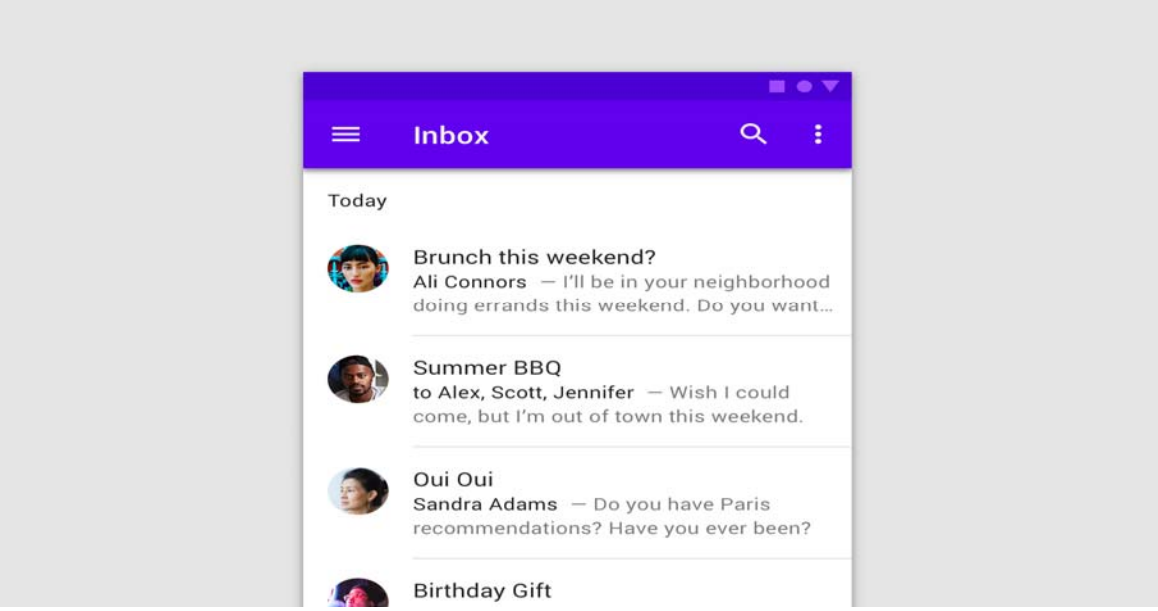

Рис. 5. RecyclerView

- SwipeRefreshLayout – это контейнер, который может содержать только один дочерний элемент (Рис. 6). Используется для отображения обновляемого списка элементов;

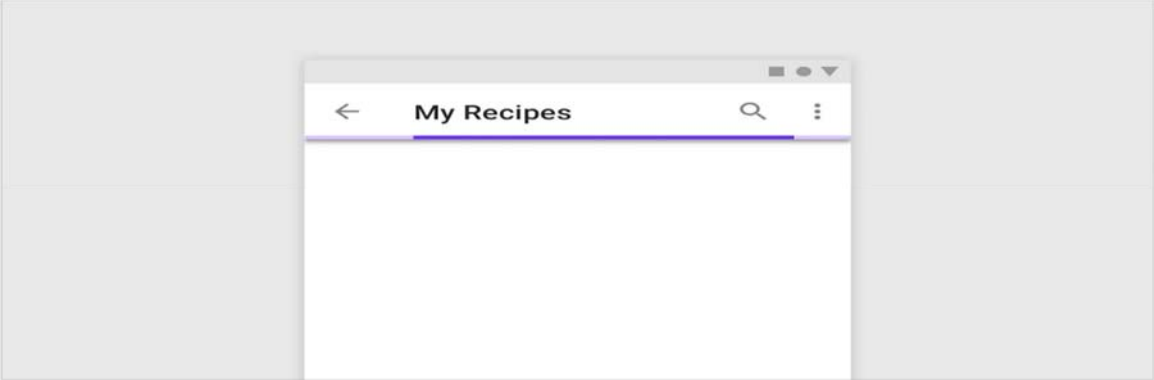

Рис. 6. SwipeRefreshLayout

- ViewPager – это контейнер, который содержит внутри себя несколько страниц. Данный компонент используется в связке с компонентом TabLayout: вместе они дают возможность пролистывать экраны с вкладками;
- Button – это кнопка, которая может содержать текст и реагировать на нажатия (Рис. 7);

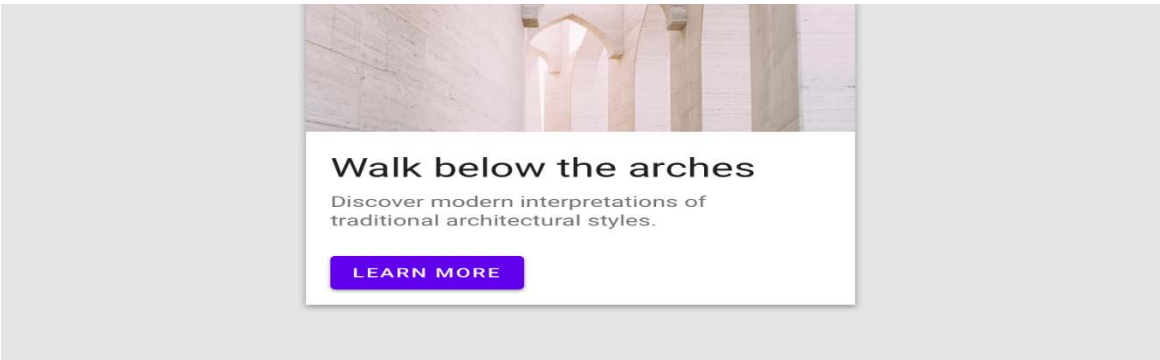

Рис. 7. Кнопка

- EditText – это поле для ввода текста, которое может быть в двух состояниях: активном и неактивном (Рис. 8). Может содержать подсказку о требуемой информации. В зависимости от указанного параметра в атрибутах может позволять ввод ограниченных групп символов;

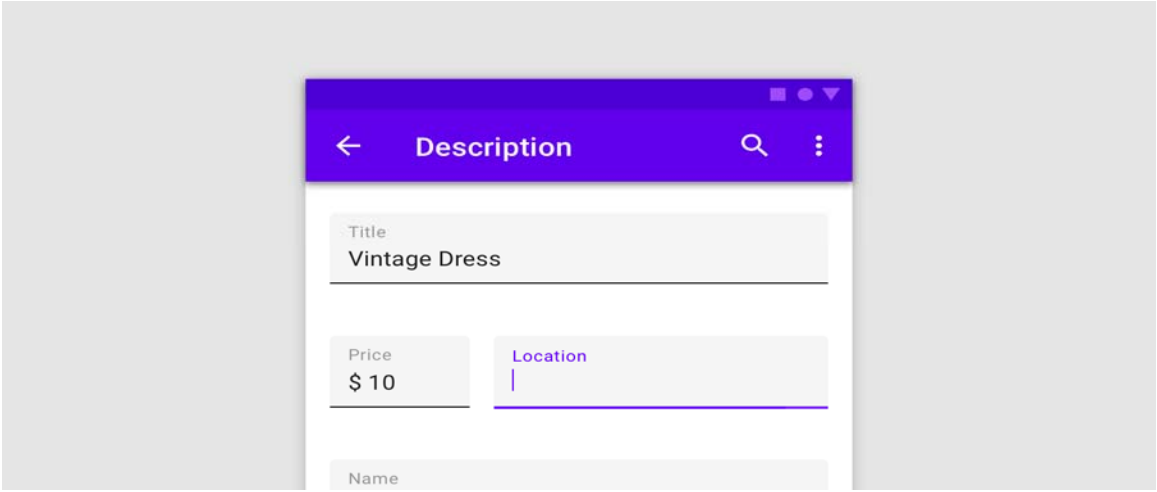

Рис. 8. EditText: активный и неактивный

- TextView – это текстовое поле, которое может содержать текст и иконку. В зависимости от параметра в атрибутах содержимое поля может находиться в разных положениях, например, в центре, справа, сверху;
- ImageView – контейнер для изображений.

Далее были сформулированы требования к наименованию элементов в sketch-проекте. Для этого было решено использовать обобщенные названия, так как зачастую дизайн пользовательских интерфейсов для приложений различных операционных систем делается вместе, а названия для одинаковых элементов могут сильно различаться в зависимости от требуемой платформы [6].

При составлении требований к структуре учитывался тот факт, что дизайнеры зачастую не знают всех тонкостей в вёрстке макетов экранов под разные платформы, поэтому в первую очередь преследовались следующие цели:

- избежать особенностей расположения элементов для разных платформ;
- не перегрузить проект большим количеством различных папок, но сделать структуру интуитивно понятной для программиста.

Таким образом, были приняты следующие варианты:

- Toolbar – папка с названием header. В ней может располагаться текст с заголовком экрана;
- TabLayout – папка с названием tabs;
- AppBarLayout – папка с названием sheader. В ней должны располагаться две папки: header и tabs, то есть Toolbar и TabLayout соответственно;
- ScrollView – папка с названием slayout. Может содержать любые элементы из списка перечисленных в технических требованиях;
- LinearLayout – папка с названием vlayout либо hlayout, что соответствует вертикальной либо горизонтальной компоновке дочерних элементов соответственно. Может содержать любые элементы из списка перечисленных в технических требованиях;
- CardView – папка с названием card. Может содержать один дочерний элемент;
- RecyclerView – папка с названием list. Может содержать один любой элемент из списка, указанного в технических требованиях. Если у папки указать имя rlist, то сгенерируется RecyclerVIew, обернутый в SwipeRefreshLayout. Такая папка аналогично может содержать любой элемент из списка, указанного в технических требованиях;
- ViewPager – папка с названием pager. Может содержать любой элемент из списка, указанного в технических требованиях.
- Button – папка с названием button. Может содержать текст с названием кнопки и стиль;
- EditText – папка с названием input. Может содержать текст с подсказкой, иконку и стиль;

- TextView – текст, в качестве имени которого указано содержимое текстового поля;
- ImageView – изображение с произвольным именем;
- Background - стиль для элементов EditText, Button и LinearLayout. Должен располагаться в корне элемента, к которому относится.

ПРИНЦИП РАБОТЫ АЛГОРИТМА

За основу алгоритма был выбран обход в глубину: из каждого json-файла, полученного из исходного sketch-проекта, строится дерево элементов, а затем обходом по данному дереву генерируются итоговые макеты экранов интерфейса. В итоге для каждого экрана на входе имеется одно дерево элементов, а на выходе – два: для экранов и для стилей, которые задают цвет, фон и форму элементов на экране (Рис. 9).

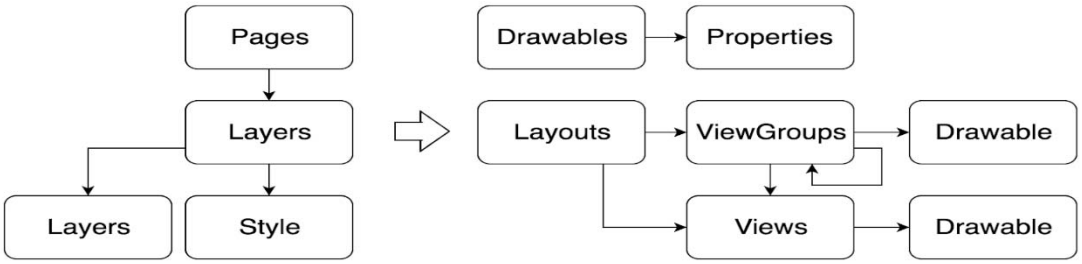

Рис. 9. Входные и выходные данные

Алгоритм генерации макетов интерфейса состоит из следующих шагов:

- распаковка файла проекта и сохранение полученных файлов в памяти устройства;
- обработка полученных json-файлов и генерация дерева всех элементов;
- генерация стилей и общих элементов для всех экранов;
- генерация макетов всех экранов;
- генерация всех итоговых файлов в памяти устройства;
- удаление вспомогательных файлов, созданных при работе программы, и перевод алгоритма в начальное состояние.

Упрощённая блок-схема данного алгоритма представлена на рис. 10 и 11.

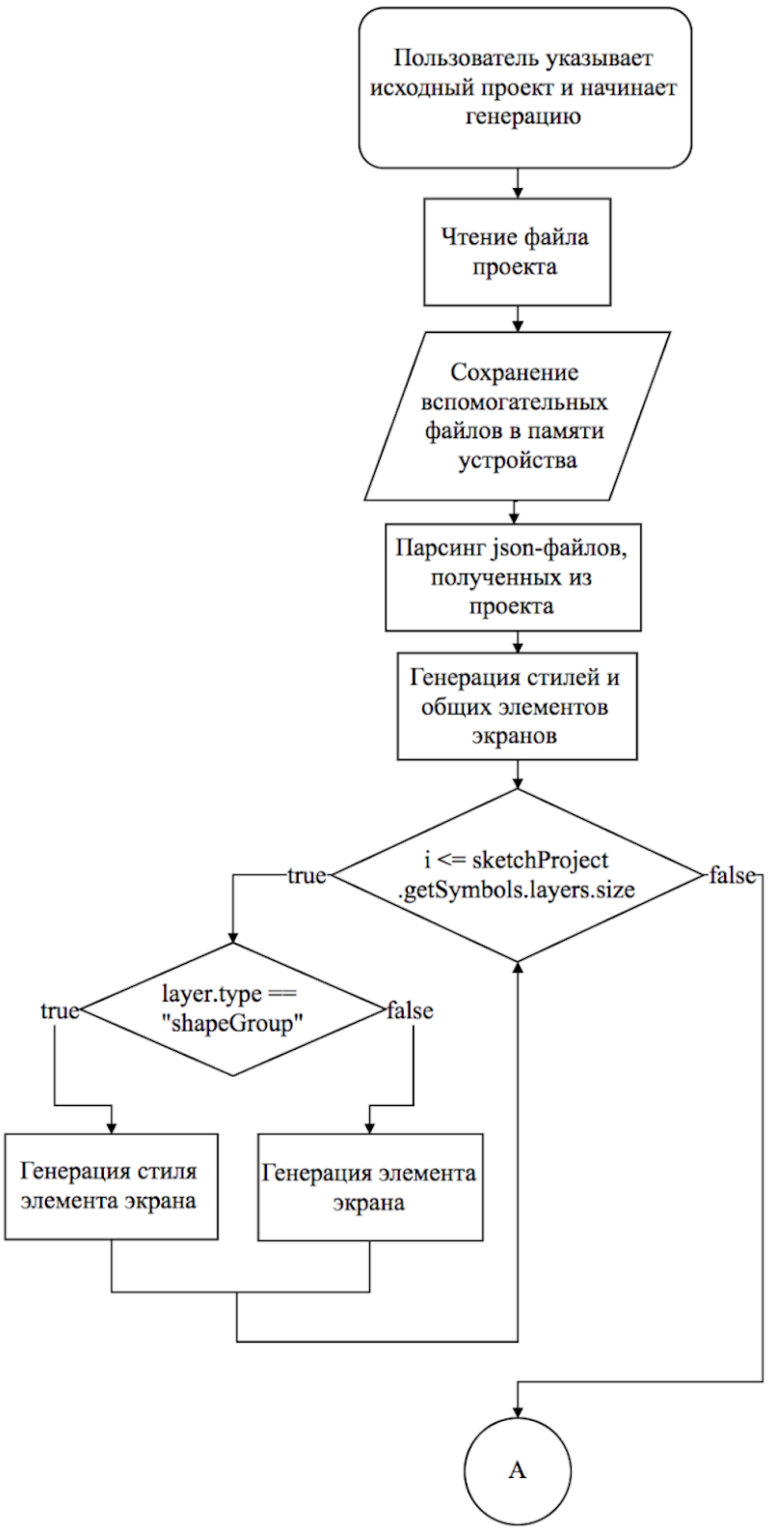

Рис. 10. Упрощённая блок-схема алгоритма (часть 1)

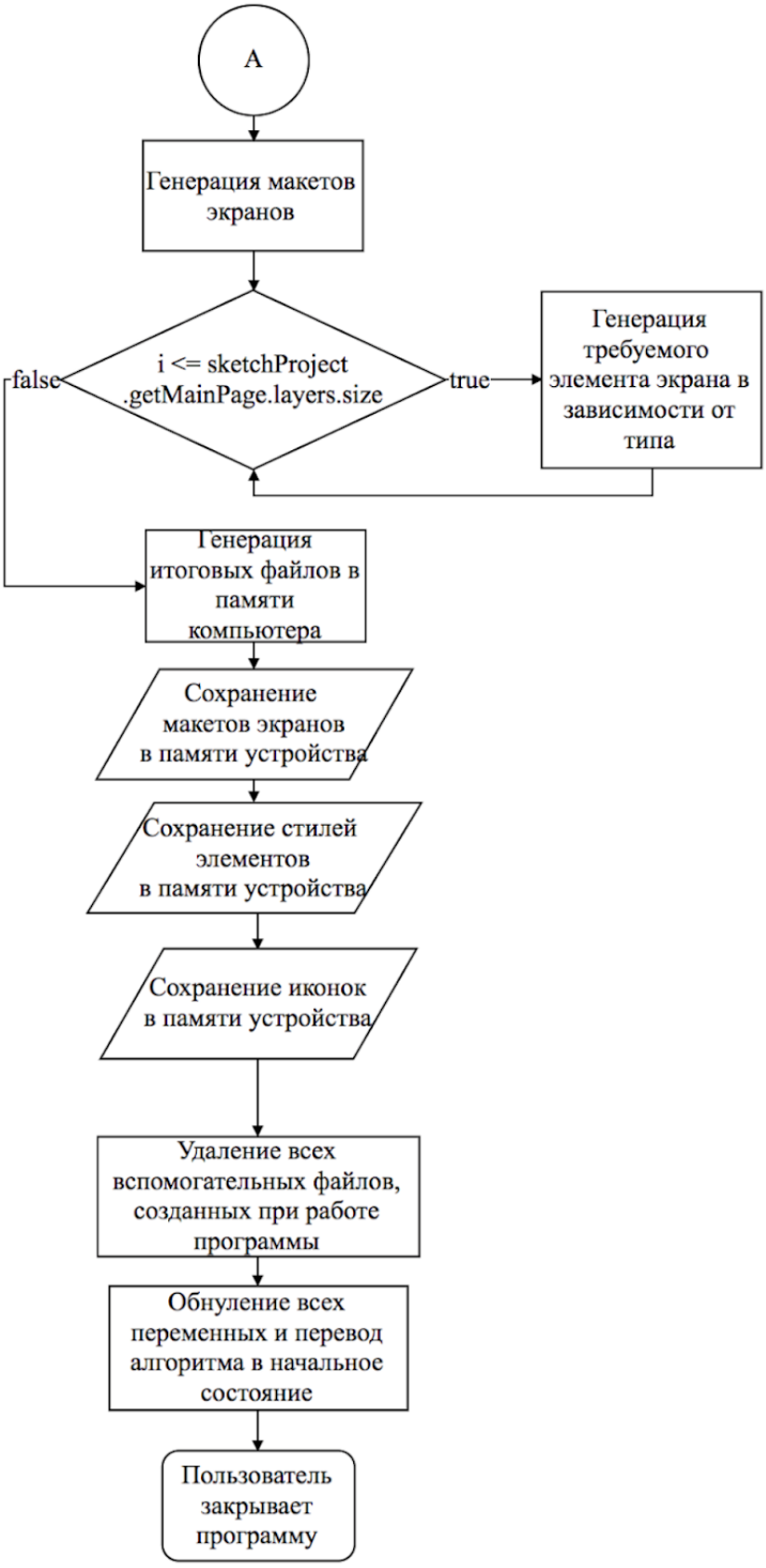

Рис. 11. Упрощённая блок-схема алгоритма (часть 2)

ПРИНЦИП РАБОТЫ ПРОГРАММНОГО РЕШЕНИЯ

Пользователь взаимодействует с программой посредством графического интерфейса (Рис. 12).

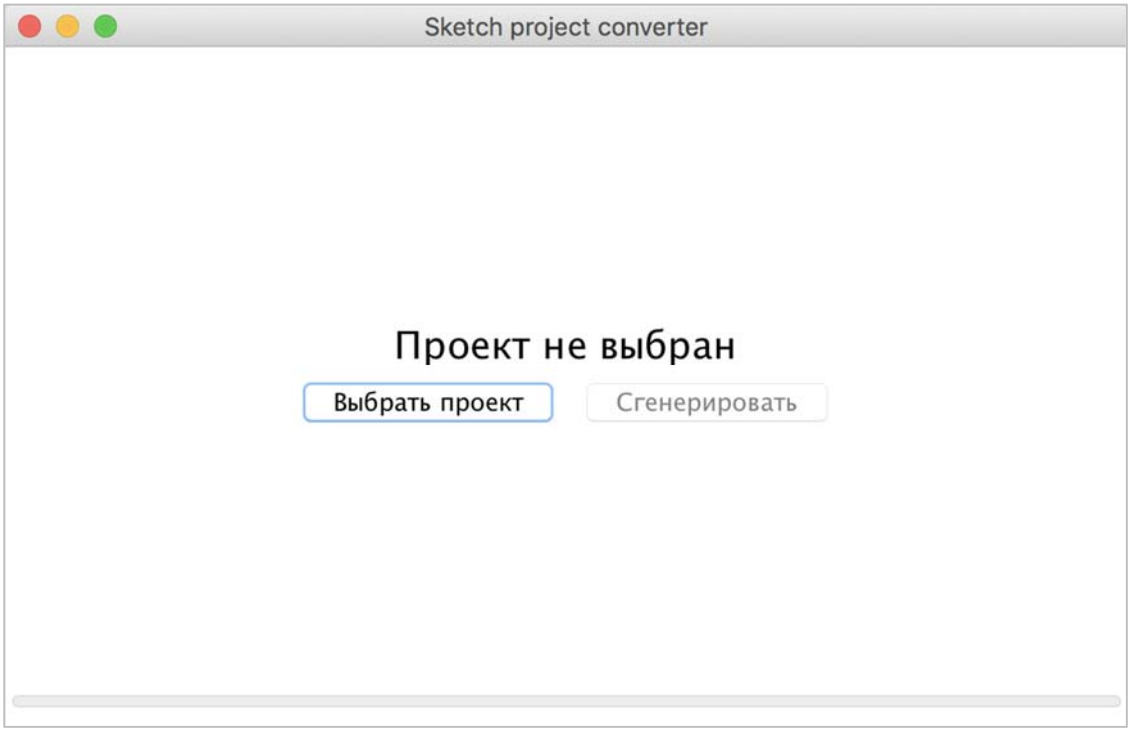

Рис. 12. Пользовательский интерфейс программы

После выбора проекта сверху над кнопкой отобразится название выбранного пользователем проекта. Если нажать на кнопку «Сгенерировать», то начнётся работа алгоритма. Когда все требуемые макеты будут сгенерированы, полоса прогресса заполнится на сто процентов, а под названием проекта отобразится количество сгенерированных макетов экранов и количество сгенерированных дополнительных макетов (Рис. 13), например, дочерние элементы для RecyclerView, страницы для ViewPager. Сгенерированные макеты экранов сохранятся в папке рядом с выбранным sketch-проектом.

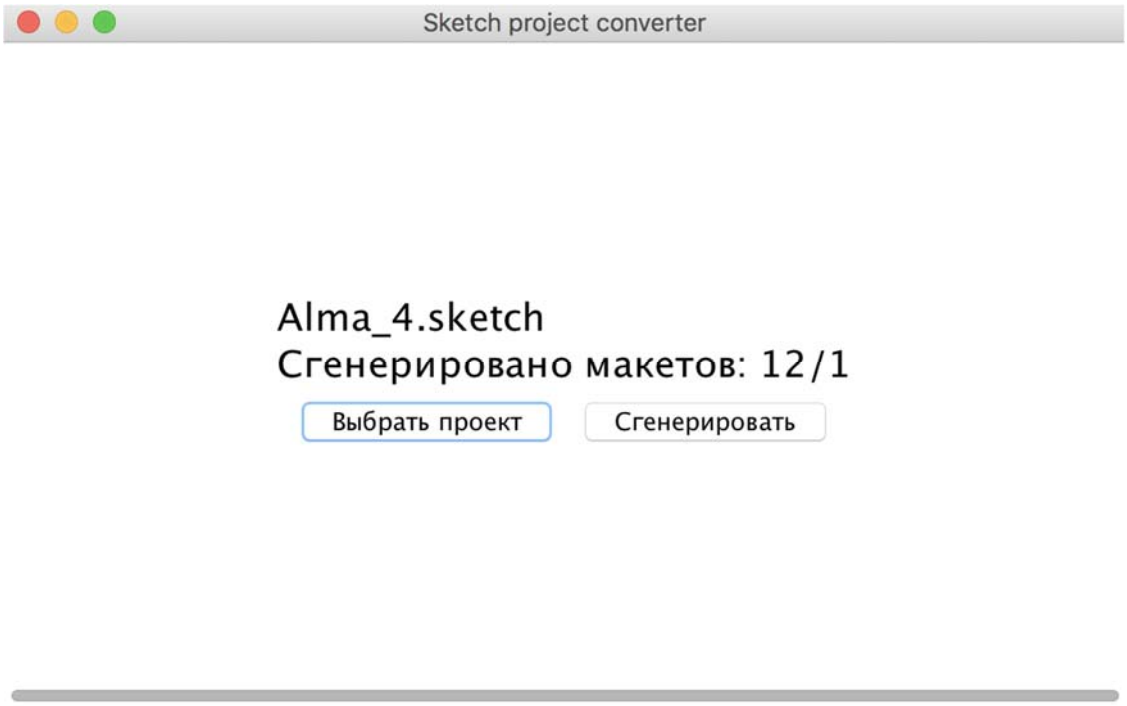

Рис. 13. Иерархия классов моделей

По исходному JSON-файлу будет сгенерирован следующий код:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  android:background="@drawable/bg"
  android:gravity="center_horizontal"
  android:layout_height="match_parent"
  android:layout_width="match_parent"
  android:orientation="vertical">
  <LinearLayout
    android:gravity="center_horizontal"
    android:layout_height="0dp"
    android:layout_weight="538"
    android:layout_width="321dp"
    android:orientation="vertical">
    <ImageView
```

```
            android:layout_height="0dp"
            android:layout_weight="254"
            android:layout_width="299dp"
            android:scaleType="centerCrop"
            app:srcCompat="@drawable/_logo" />
        <TextView
            android:gravity="center_horizontal"
            android:layout_height="0dp"
            android:layout_marginBottom="28dp"
            android:layout_weight="30"
            android:layout_width="match_parent"
            android:text="ALMA"
            android:textColor="#ff5c5c5c" />
        <LinearLayout
            android:gravity="center"
            android:layout_height="0dp"
            android:layout_marginBottom="56dp"
            android:layout_weight="89"
            android:layout_width="281dp"
            android:orientation="vertical">
    <EditText                                                    an-
droid:background="@drawable/inputc2593e28_62f5_4107_bb62"
            android:drawablePadding="4dp"
            android:drawableStart="@drawable/ic_phone"
            android:hint="Номер телефона"
            android:layout_height="0dp"
            android:layout_weight="45"
            android:layout_width="281dp"
            android:padding="8dp"
            android:textColor="#ff6bcbd9"
            android:textColorHint="#ff6bcbd9" />
    <EditText  android:background="@drawable/input1b1d12af_b927_49f3_8bc1"
            android:drawablePadding="4dp"
```

```
            android:drawableStart="@drawable/ic_password"
            android:hint="Пароль"
            android:layout_height="0dp"
            android:layout_weight="45"
            android:layout_width="281dp"
            android:padding="8dp"
            android:textColor="#ff6bcbd9"
            android:textColorHint="#ff6bcbd9" />
        </LinearLayout>
    <Button                                                    an-
droid:background="@drawable/buttonb04c99c4_f455_4cb5_9da9"
            android:layout_height="0dp"
            android:layout_marginBottom="19dp"
            android:layout_weight="48"
            android:layout_width="281dp"
            android:text="ВОЙТИ"
            android:textColor="#ffffffff" />
        <TextView
            android:gravity="center_horizontal"
            android:layout_height="wrap_content"
            android:layout_marginBottom="15dp"
            android:layout_width="match_parent"
            android:text="Создать аккаунт"
            android:textColor="#ff6e6e6e" />
      </LinearLayout>
    </LinearLayout>
```

Пример сгенерированного макета представлен на Рис. 14. Можно заметить, что макет экрана генерируется достаточно точно, но есть небольшие отличия в тексте, которые связаны с техническими особенностями хранения информации о нём в исходных данных sketch-проекта. Также вместо картинки отображается закрашенный квадрат, что связано с тем, что в исходном проекте хранится много изображений, среди которых присутствуют изображения по умолча-

нию, отделение которых от основных графических файлов накладывает дополнительный ряд ограничений к структуре sketch-проекта. Так как основная масса изображений в мобильных приложениях загружается из интернета, было решено отказаться от данной функции.

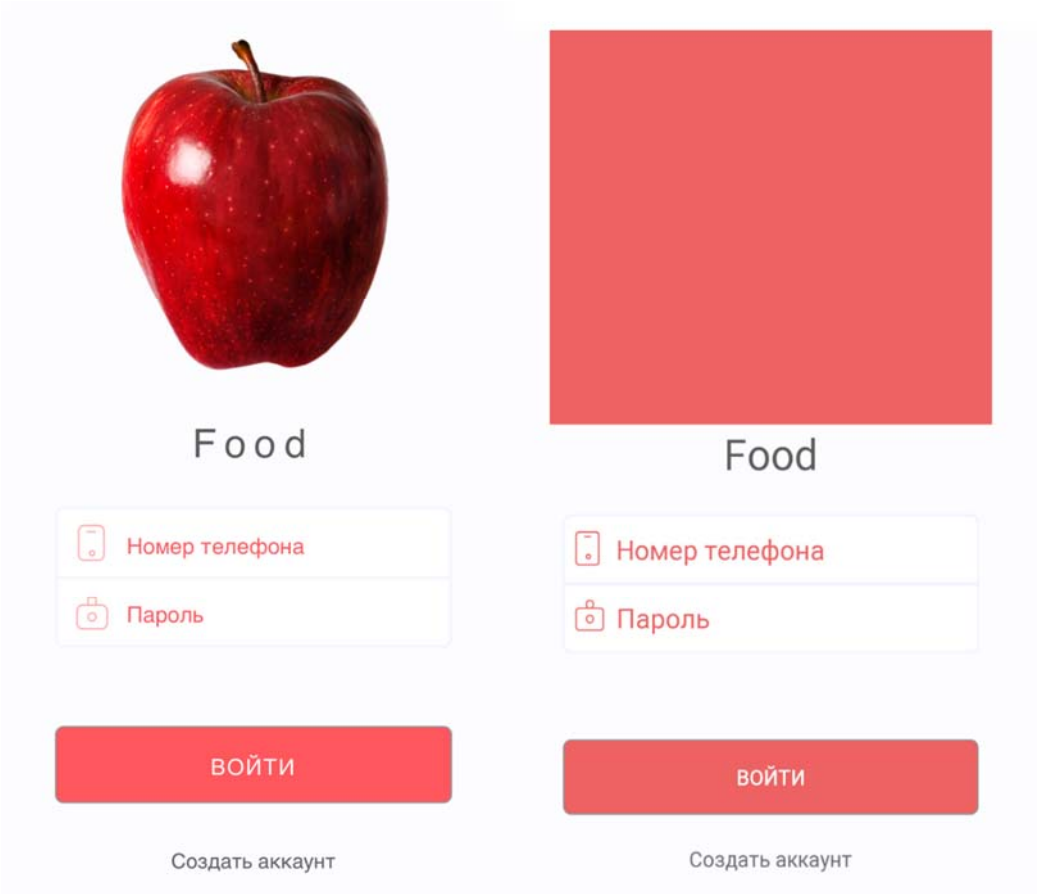

Рис. 14. Макеты: исходный и сгенерированный программой

ЗАКЛЮЧЕНИЕ

Разработано программное решение для генерации кода пользовательского интерфейса нативных мобильных приложений для операционной системы Android. Разработанный программный продукт обладает возможностью генерации пользовательских интерфейсов мобильных приложений по данным графического редактора.

Задачи, выполненные в рамках работы:

- разработаны технические требования и требования к структуре sketch-проекта;
- разработан алгоритм генерации кода макетов UI по данным графического редактора;

- создан программный инструмент, реализующий генерацию файлов пользовательских интерфейсов для дальнейшего использования в Android-проектах.

Разработанный программный инструмент позволит Android-разработчикам сократить время на создание пользовательских интерфейсов по макетам графического редактора, что, в свою очередь, сократит общее время разработки программного продукта для операционной системы Android.

В дальнейшем планируется расширить список генерируемых элементов, добавить генерацию паттернов MVP/MVVM и опубликовать проект в открытом доступе, что даст возможность сторонним разработчикам кастомизировать алгоритм согласно своим требованиям.

СПИСОК ЛИТЕРАТУРЫ

1. Введение в жизненный цикл разработки мобильных приложений [Электронный ресурс]. URL: https://codedocs.ru/xamarin/vvedenie-v-zhiznennyj-tsikl-razrabotki-mobilnyh-prilozhenij.html

2. Sketch [Электронный ресурс]. URL: https://sketchapp.com

3. Zeplin [Электронный ресурс]. URL: https://zeplin.io/

4. Supernova [Электронный ресурс]. URL: https://supernova.studio/

5. Material Design [Электронный ресурс]/ URL: https://material.io/ guidelines/

6. Human Interface Guidelines [Электронный ресурс]. URL: https://developer. apple.com/ios/human-interface-guidelines/overview/themes/

7. Android Studio [Электронный ресурс]. URL: https://developer.android.com/ studio/projects/index.html

ALGORITHM FOR GENERATION OF MOBILE APPLICATIONS' UI CODE BASED ON DATA OF GRAPHIC EDITOR

A. Usachev

Higher School of Information Technologies and Intelligent Systems at Kazan Federal University

usacheow.ar@gmail.com

Abstract

The paper is devoted to the development of the algorithm for generating the code of user interfaces for native Android applications based on the data of the graphical editor. The problem of negative impact on the development time of the product for performing routine actions is considered, and a software tool for solving this problem is proposed.

Keywords: *user interface, graphic editor, generation algorithm*

REFERENCES

1. CodeDocs site. URL: https://codedocs.ru/xamarin /vvedenie-v-zhiznennyj-tsikl-razrabotki-mobilnyh-prilozhenij.html
2. Official site of the program Sketch. URL: https://sketchapp.com
3. Official site of the program Zeplin. URL: https://zeplin.io/
4. Official site of the program Supernova. URL: https://supernova.studio/
5. Official site with guidelines for Material Design. URL: https://material. io/guidelines/
6. Official site with guidelines for Human Interface. URL: https://developer.apple.com/ios/human-interface-guidelines/overview/themes/
7. Official site for Android developer. URL: https://developer. android.com/studio/projects/index.html

СВЕДЕНИЯ ОБ АВТОРЕ

УСАЧЁВ Артемий Юрьевич – студент Высшей школы информационных технологий и интеллектуальных систем Казанского федерального университета, android разработчик.

Artemy Yurievich USACHEV – student of the Higher School of Information Technologies and Intelligent Systems of Kazan Federal University, middle developer of software for Android.

email: usacheow.ar@gmail.com

Материал поступил в редакцию 10 июня 2018 года