Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Лабораторная работа

Дисциплина: Проектирование мобильных приложений

Тема: Layouts

Выполнил студент гр. 3530901/90201 _____Е. В. Бурков
(подпись)

Принял старший преподаватель _____ А. Н. Кузнецов
(подпись)

“_____” _____ 2021 г.

Санкт-Петербург
2021

Содержание

Цели

- Познакомиться со средой разработки Android Studio
- Изучить основные принципы верстки layout с использованием XML
- Изучить основные возможности и свойства LinearLayout
- Изучить основные возможности и свойства ConstraintLayout

Задачи

- Изучить layout ресурсы (LinearLayout и ConstraintLayout) по документации на сайте: https://developer.android.com
- Согласно варианту создать необходимые layout ресурсы.
- Ответить на вопрос: В каких случаях целесообразно использовать LinearLayout, в каких ConstraintLayout?

Введение

Ресурсы очень важная часть android приложения. При программировании принято держать некоторые объекты, такие как изображения, константы, цвета, стили, анимации, интерфейс приложения и так далее. Во многом это используется для избегания хардкодинга. Все ресурсы хранятся в папке res.

К ресурсам можно обращаться из других ресурсов или из кода (статический класс R).

Layout (компоновка) – разметка или макет для визуальной части нашего приложения. Все компоновки наследуются от ViewGroup, а ViewGroup наследуется от View.

View – класс, который представляет собой базовый контейнер для компонентов пользовательского интерфейса. Потомками данного класса являются элементы для графического интерфейса, такие как , например, виджеты или компоновки.

ViewGroup – класс, который является основой для layout’ов (невидимые контейнеры для других виджетов или компоновок) и определяет их свойства.

Со всеми функциями и свойствами данных классов можно ознакомиться на официальном сайте документации. Далее будут описаны только использовавшиеся атрибуты.

LinearLayout

Данная компановка позволяет располагать остальные компоненты горизонтально(или вертикально) в одну колону(или ряд). Для лучшего пояснения перейдём к первому заданию.

Задача 1_6

Необходимо реализовать следующий макет экрана с использованием LinearLayout:

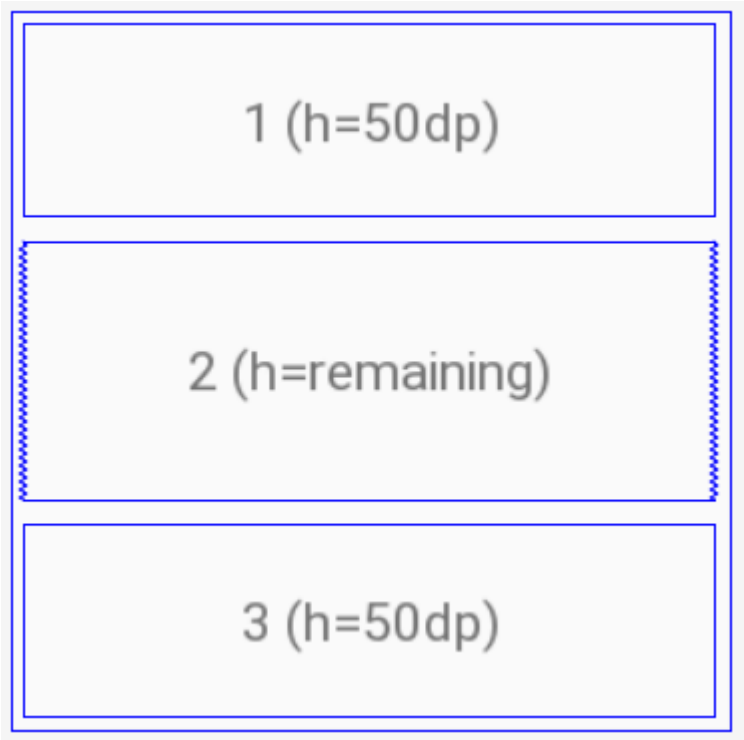

Рис. 1 Макет 1_6

Ресурс нашей компоновки предоставлен ниже:

Листинг 1 Задание 1_6

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <!--I think name of dim in okay_case -->
    <!--Added img for playing with 9.png -->
    <ImageView
        android:layout_width="match_parent"
        android:layout_height="@dimen/dim_50"

        android:contentDescription="@string/haskell_logo"
```

```
        android:src="@drawable/haskell" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"

        android:gravity="start"
        android:hint="@string/haskell_question"
        android:autofillHints="search info"
        android:inputType="text" />
    <Button
        android:layout_width="match_parent"
        android:layout_height="@dimen/dim_50"

        android:gravity="start"
        android:text="@string/haskell_button"
    />
</LinearLayout>
```

Данный ресурс представляет собой XML файл. Так же возможно задавать компоновки при помощи кода на языке Kotlin (Compose).

Корневой элемент – Linear Layout. Имеет следующие атрибуты:

- android:layout_width – параметр отвечающий за ширину компонента. Можно задавать величины в разных единицах измерения. Есть 2 особых значения:
  match_parent – занять ширину как у родительного элемента
  wrap_content – занять столько места, сколько хватит для отображения всего контента
  В нашем случае эти атрибуты равняются "match_parent" и занимают весь экран нашего устройства.
- android:layout_height – как layout_width, только отвечает за ширину
- android:orientation – атрибут для LinearLayout. Устанавливает, как будут располагаться элементы, горизонтально или вертикально.

Далее идут дочерние элементы:

ImageView – виджет отображающий ресурс, например картинку.

- android:contentDescription – описание для картинки
- android:src – путь к ресурсу контента

При написании когда оказалось, что атрибут contentDescription должен быть определён (выдаёт варнинг). Так же использовалась высота компонента из соответствующего ресурса.

EditText – виджет для ввода или модификации текста.

- android:gravity – отвечает за расположение контента внутри компонента, имеет большое количество вариаций
- android:hint – то что будет находиться в поле, при отсутствии ввода
- android:autofillHints - описывает содержимое представления, чтобы служба автозаполнения могла заполнить соответствующие данные
- android:inputType – вид ввода

Button – кнопка

Для соответствия заданию в первый и последний элемент была прописана высота 50dp, а для поля ввода использован атрибут веса, для заполнения всего свободного пространства.

Рис. 2 Получившийся layout

Задача 1_15

Необходимо реализовать следующий макет экрана с использованием LinearLayout:

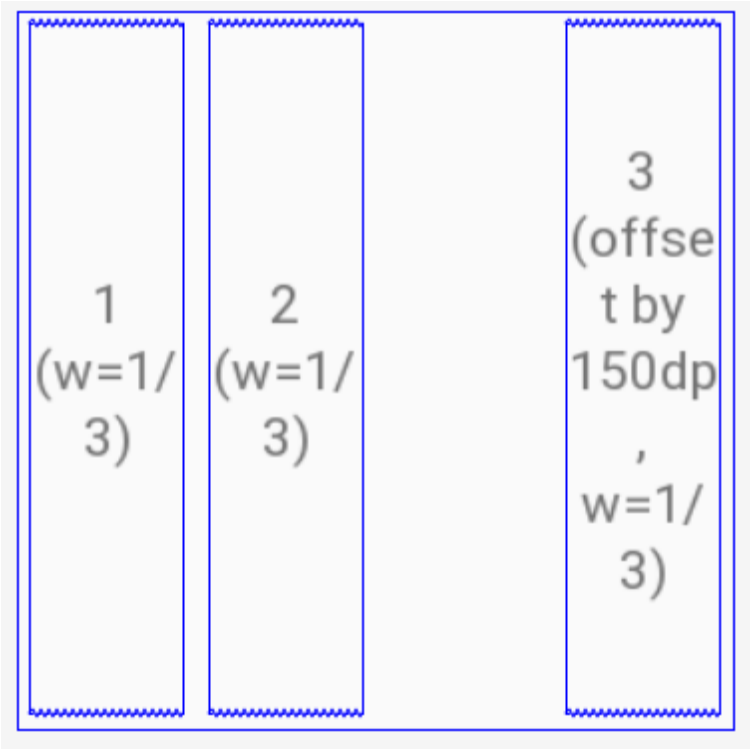

Рис. 3 Макет 1_15

Ресурс нашей компоновки предоставлен ниже:

Листинг 2 Задание 1_15

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    >

    <ImageView
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"

        android:contentDescription="@string/haskell_logo"
        android:scaleType="centerCrop"
        android:src="@drawable/haskell_logos" />

    <ImageButton
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
```

```
        android:background="@drawable/ic_launcher_background"
        android:contentDescription="@string/android_logo_button"
        android:src="@drawable/ic_launcher_foreground" />

    <RadioGroup
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_marginStart="@dimen/dim_150"
        android:layout_weight="1"
        android:orientation="vertical">

        <RadioButton
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/mushroom_soup"
            android:textSize="@dimen/sp_11" />

        <RadioButton
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/funny_kebab"
            android:textSize="@dimen/sp_11" />

        <RadioButton
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/cutlets_with_mashed_potatoes"
            android:textSize="@dimen/sp_11" />

    </RadioGroup>


</LinearLayout>
```

Для соответствия заданию была выбрана вертикальная ориентация, для всех дочерних элементов установлен равный вес, а для последнего использовался атрибут для отступа от предыдущего элемента.

Листинг 3 haskell_logos.xml

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:drawable="@drawable/haskell1"
        />
    <item
        android:drawable="@drawable/haskell2"
        android:top="@dimen/dim_50"
        android:left="@dimen/dim_50"
        />
    <item
        android:drawable="@drawable/haskell3"
        android:top="@dimen/dim_150"
        android:left="@dimen/dim_150"
        />
</layer-list>
```

Для первого элемента в ряду использовался ImageView с layer-list из логотипов Haskell. Layer-list позволяет “наслаивать” изображения друг на друга, получая в итоге один View.

Атрибут scaleType использовался для задания растягивания картинки.

ImageButton – Button совмещенная с ImageView.

RadioGroup – список в котором можно выбрать только один пункт. Наследуется от LinearLayout. Внутри должен содержать кнопки RadioButton.

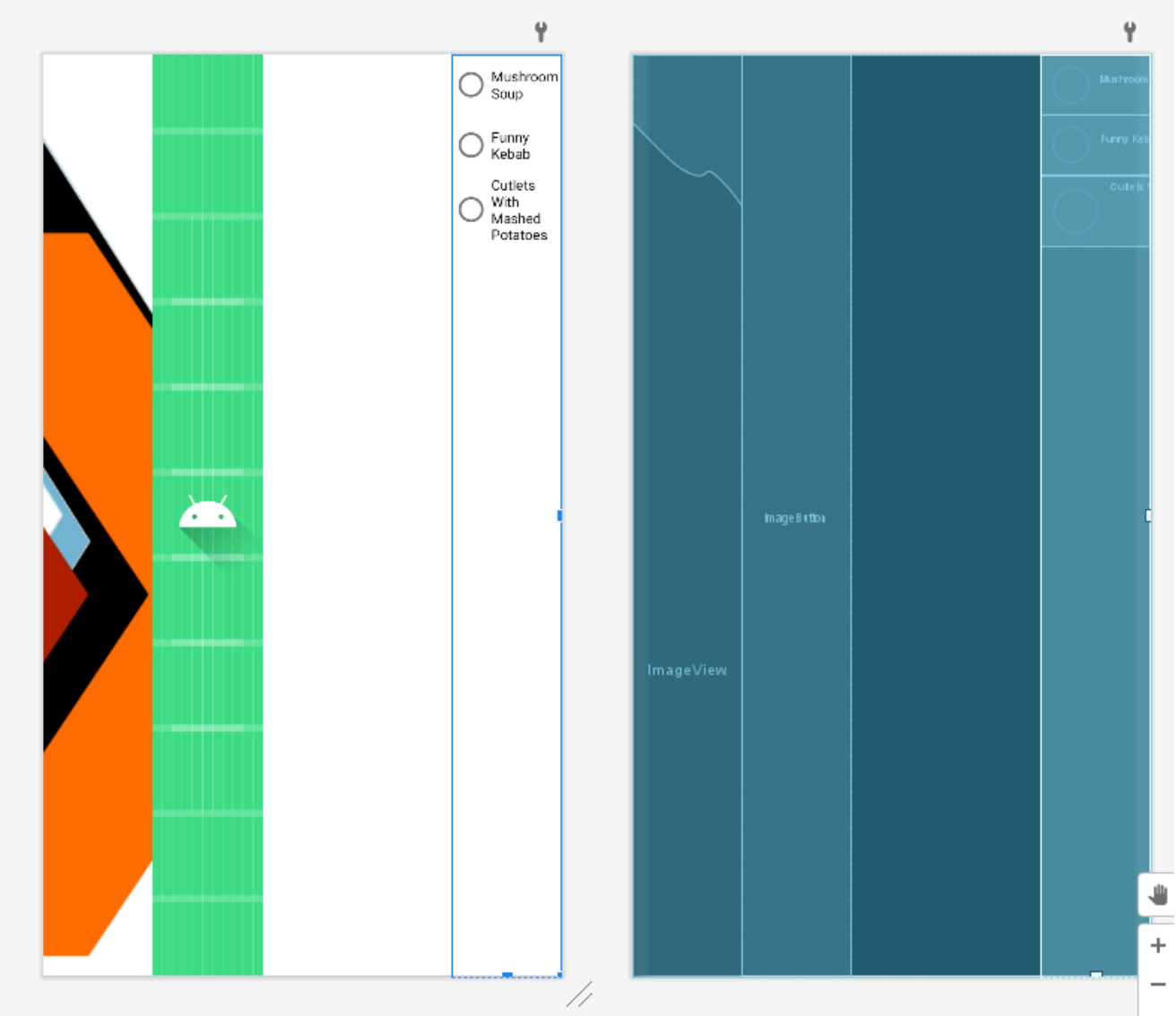

Рис. 4 Получившийся layout

Задача 1_15_alt

Можно заменить атрибут marginStart на виджет Space. Его предназначение – создание отступов между элементами.

Листинг 4 Задание 1_15_alt

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <ImageView
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:contentDescription="@string/haskell_logo"
        android:src="@drawable/haskell_logos"
        android:scaleType="centerCrop"
        />

    <ImageButton
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="@drawable/ic_launcher_background"
        android:contentDescription="@string/android_logo_button"
        android:src="@drawable/ic_launcher_foreground" />

    <Space
        android:layout_width="@dimen/dim_150"
        android:layout_height="match_parent" />

    <RadioGroup
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:orientation="vertical">

        <RadioButton
            android:textSize="@dimen/sp_11"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/mushroom_soup" />

        <RadioButton
            android:textSize="@dimen/sp_11"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/funny_kebab" />

        <RadioButton
            android:textSize="@dimen/sp_11"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/cutlets_with_mashed_potatoes" />

    </RadioGroup>
</LinearLayout>
```

В итоге получаем такую же компоновку.

ConstraintLayout

Вид компановки, который позволяет более гибко позиционировать компоненты между собой. Основные атрибуты, которые мы будет использоваться связаны с взаимным расположением концов элемента (Top, Bottom,End,Start) и другого элемента. Перейдём к примеру использования. Так же для использования необходимо создавать идентификаторы компонентов.

Задание 2_6

Необходимо выполнить задание 1_6 при помощи ConstraintLayout. Ресурс предоставлен ниже:

Листинг 5 Задание 2_6

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="@dimen/dim_50"

        android:contentDescription="@string/haskell_logo"
        android:src="@drawable/haskell"


        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@id/editText"/>

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:autofillHints="search info"
        android:gravity="start"

        android:hint="@string/haskell_question"
        android:inputType="text"

        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toTopOf="@id/button"
        app:layout_constraintTop_toBottomOf="@id/imageView" />

    <Button
        android:id="@+id/button"

        android:layout_width="match_parent"
```

```
        android:layout_height="@dimen/dim_50"
        android:gravity="start"
        android:text="@string/haskell_button"

        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toBottomOf="@id/editText" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Для каждого дочернего элемента указано расположение относительно другого элемента, например самый верхний элемент соединён верхом с родителем, а низом с верхом следующего элемента.

Так же для заполнения оставшегося места нет необходимости в атрибуте weight. При правильном соединении элементы сами займут всё оставшееся место.

По итогу компоновка эквивалента необходимому макету

Задача 2_15

Необходимо выполнить задание 1_15 с использованием ConstraintLayout.

Листинг 6 Задание 2_15

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/haskellLogo"
        android:layout_width="0dp"
        android:layout_height="match_parent"


        android:contentDescription="@string/haskell_logo"
        android:scaleType="centerCrop"
        android:src="@drawable/haskell_logos"

        app:layout_constraintEnd_toStartOf="@id/androidButton"
        app:layout_constraintStart_toStartOf="parent" />

    <ImageButton
        android:id="@+id/androidButton"
        android:layout_width="0dp"
        android:layout_height="match_parent"
```

```
        android:background="@drawable/ic_launcher_background"
        android:contentDescription="@string/android_logo_button"
        android:src="@drawable/ic_launcher_foreground"

        app:layout_constraintEnd_toStartOf="@id/radioGroup"
        app:layout_constraintStart_toEndOf="@id/haskellLogo"

        />

    <RadioGroup
        android:id="@+id/radioGroup"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_marginStart="@dimen/dim_150"
        android:orientation="vertical"

        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@id/androidButton">


        <RadioButton
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/mushroom_soup"
            android:textSize="@dimen/sp_11" />

        <RadioButton
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/funny_kebab"
            android:textSize="@dimen/sp_11" />

        <RadioButton
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/cutlets_with_mashed_potatoes"
            android:textSize="@dimen/sp_11" />

    </RadioGroup>


</androidx.constraintlayout.widget.ConstraintLayout>
```

Здесь для отступа оставили атрибут marginStart.

По итогу компоновка эквивалента необходимому макету

Задание 3_6

Необходимо создать layout ресурс для следующего макета:

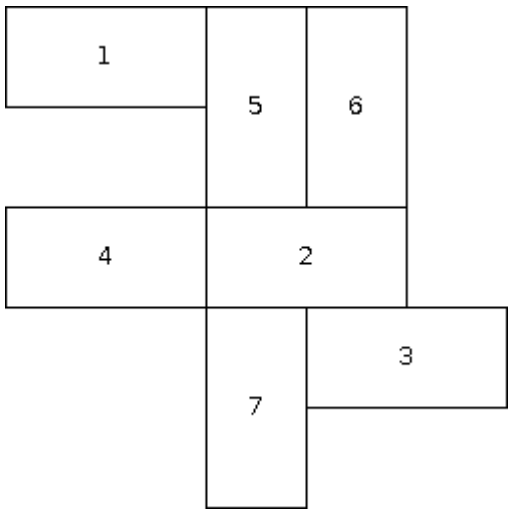

Рис. 5 Макет 3_6

Данный макет является очень сложным. Для упрощения процесса создания ресурса были использованы GuideLines. Данный класс является помощником при создании ConstraintLayout. Были заданы 3 таких линии и при их помощи был создан следующий ресурс:

Листинг 7 Задание 3_6

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/teal_700">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="@color/teal_200"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintDimensionRatio="1:1"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <androidx.constraintlayout.widget.Guideline
            android:id="@+id/line1"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:orientation="vertical"
            app:layout_constraintGuide_percent="0.4" />

        <androidx.constraintlayout.widget.Guideline
            android:id="@+id/line2"
            android:layout_width="0dp"
```

```
            android:layout_height="0dp"
            android:orientation="vertical"
            app:layout_constraintGuide_percent="0.8" />

        <androidx.constraintlayout.widget.Guideline
            android:id="@+id/line3"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:orientation="horizontal"
            app:layout_constraintGuide_percent="0.6" />

        <ImageView
            android:id="@+id/imageView1"
            android:layout_width="0dp"
            android:layout_height="0dp"

            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"
            app:layout_constraintDimensionRatio="2:1"

            app:layout_constraintEnd_toStartOf="@id/line1"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <ImageView
            android:id="@+id/imageView2"
            android:layout_width="0dp"
            android:layout_height="0dp"

            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"

            app:layout_constraintDimensionRatio="2:1"
            app:layout_constraintEnd_toStartOf="@id/line2"
            app:layout_constraintStart_toEndOf="@id/line1"
            app:layout_constraintTop_toBottomOf="@id/imageView5" />

        <ImageView
            android:id="@+id/imageView3"
            android:layout_width="0dp"
            android:layout_height="0dp"

            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"

            app:layout_constraintDimensionRatio="2:1"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toEndOf="@id/imageView7"
            app:layout_constraintTop_toTopOf="@id/line3" />

        <ImageView
            android:id="@+id/imageView4"
            android:layout_width="0dp"
            android:layout_height="0dp"

            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"

            app:layout_constraintBottom_toBottomOf="@id/line3"
            app:layout_constraintDimensionRatio="2:1"
            app:layout_constraintEnd_toStartOf="@id/imageView2"
```

```
            app:layout_constraintStart_toStartOf="parent"

            />

        <ImageView
            android:id="@+id/imageView5"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"

            app:layout_constraintDimensionRatio="1:2"
            app:layout_constraintEnd_toStartOf="@id/imageView6"
            app:layout_constraintStart_toEndOf="@id/line1"
            app:layout_constraintTop_toTopOf="parent" />

        <ImageView
            android:id="@+id/imageView6"
            android:layout_width="0dp"
            android:layout_height="0dp"

            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"

            app:layout_constraintBottom_toTopOf="@id/imageView2"
            app:layout_constraintDimensionRatio="1:2"
            app:layout_constraintEnd_toStartOf="@id/line2"
            app:layout_constraintStart_toEndOf="@id/imageView5"
            app:layout_constraintTop_toTopOf="parent" />

        <ImageView
            android:id="@+id/imageView7"
            android:layout_width="0dp"
            android:layout_height="0dp"

            android:background="@drawable/ic_launcher_background"
            android:contentDescription="@string/android_logo_button"
            android:src="@drawable/ic_launcher_foreground"

            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintDimensionRatio="1:2"
            app:layout_constraintStart_toStartOf="@id/line1"
            app:layout_constraintTop_toBottomOf="@id/line3"

            />


    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Для лучшей читабельности кода были выполнены привязки по всем краям компонентов. Так же для того, чтобы всегда занимался один максимально большой квадрат использован вложенный ConstraintLayout с атрибутом, отвечающим за соотношение сторон. Данный атрибут использовался и во всех дочерних элементах.

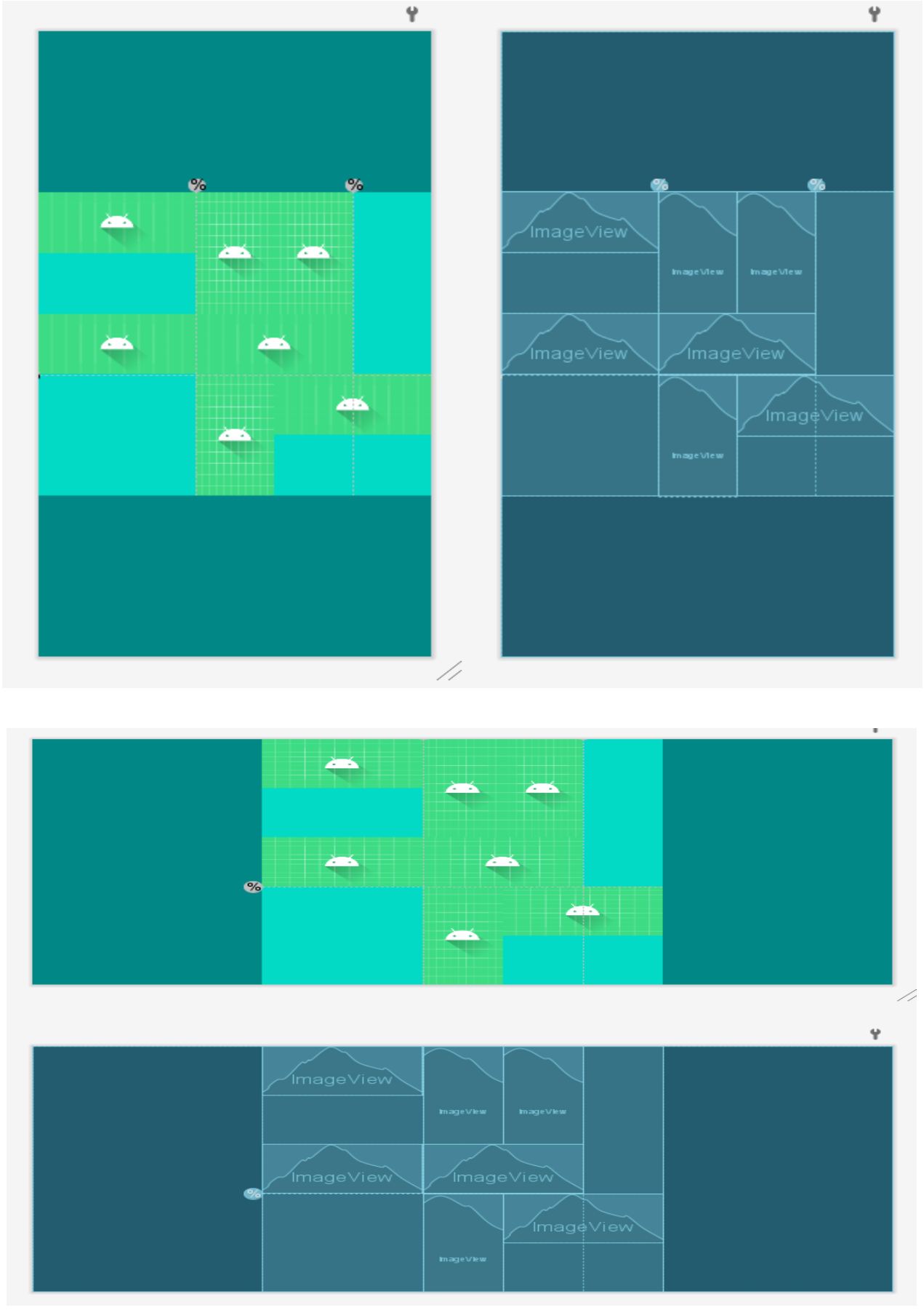

Рис. 6 Получившаяся компоновка

Выводы

В ходе данной лабораторной работы было произведено знакомство со средой разработки Android приложений Android Studio. Так как она основана на знакомой IDE от JetBrains, то большинство функций для работы с кодом и проектом были уже известны из других IDE (intellij IDEA, PyCharm, CLion). Больше внимания для меня заслужили функции связанные с Android проектированием. Удобный вынос константы в ресурс сэкономил много времени, так же удобно, что среда разработки сама понимает в какой ресурс надо записать значение (например в dimens.xml или strings.xml). Использовался редактор изображений в формате .9.png. А также редактор компоновок. Он позволяет задавать описание не xml файлом, а создавать layout интерактивно, перемещая нужные компоненты на рабочую область и работать с ними там. Несмотря на это, я писал всё через XML файл, потому что мне привычней знать каждый аспект того, как я располагаю элемент. После выполнения заданий в среде разработке был только одно предупреждение о контрасте изображения на фоне.

По лекции и документации были изучены основы создания пользовательского интерфейса для Android приложений. Были реализованы макеты согласно заданиям.

В каких случаях целесообразно использовать LinearLayout, в каких ConstraintLayout?

Я считаю, что LinearLayout следует использовать в случаях, если нам необходимо описать простой интерфейс, иначе в описании будет слишком много вложенных тэгов и файл будет намного дольше парситься. В остальных случаях лучше всего использовать ConstraintLayout

Сссылка на репозиторий: https://github.com/wooftown/spbstu-android

Список источников:

- https://developer.android.com
- https://github.com/andrei-kuznetsov/android-lectures