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

Отчёт по лабораторной работе № 1

Дисциплина: Проектирование мобильных приложений

Тема: Layouts

Выполнил студент гр. 3530901/90201 _____ Д.Е. Бакин
(подпись)

Принял преподаватель _____ А.Н. Кузнецов
(подпись)

“____” _____ 2021 г.

Санкт-Петербург

2021

Репозиторий GitHub:

https://github.com/donebd/Android_spbstu2021/tree/main/labs/lab1

1. 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*

2. Задачи

Задача 1. LinearLayout

Создайте layout ресурсы для следующих макетов экрана с использованием LinearLayout. Вариант 3 - изображения 3 и 12 (Рис. 1-2).

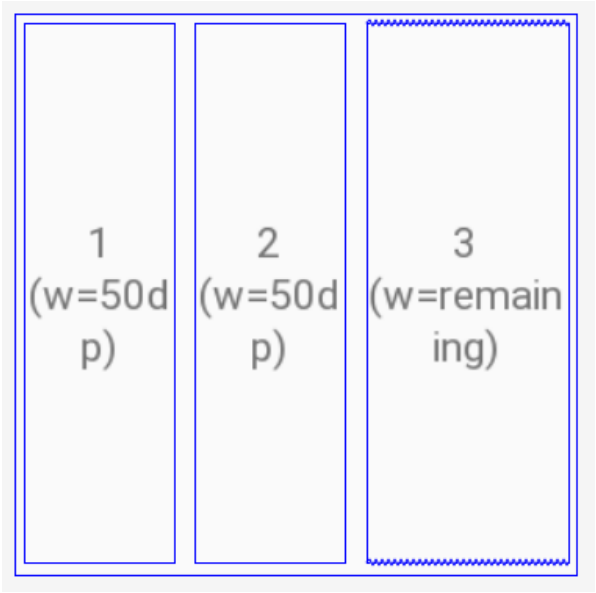

Рис. 1

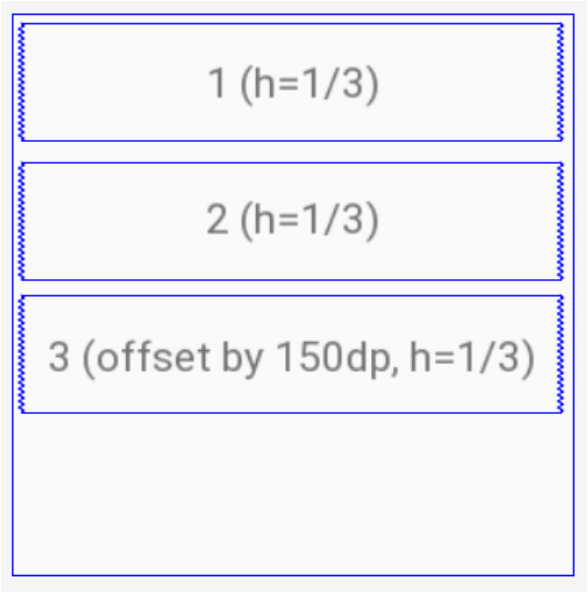

Рис. 2

Задача 2. ConstraintLayout

Решите задачу 1 (обе подзадачи) с использованием ConstraintLayout.

Задача 3. ConstraintLayout

Создайте layout ресурс для следующего макета экрана с использованием ConstraintLayout. Согласно варианту, это изображение 3 (Рис. 3).

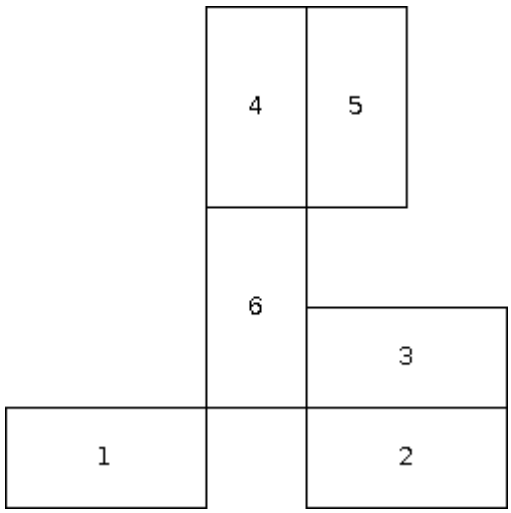

Рис. 3

3. Ход работы

3.1. Задача 1. LinearLayout

Для выполнения задачи была изучена документация о LinearLayout.

Layout_1_3

В макете содержатся 3 виджета, которые нужно разместить в особом порядке. Для LinerarLayout атрибут *android:orientation* по умолчанию задан как *horizontal.* Для размещения виджетов использовался атрибут View *android:layout_width*, который отвечает за ширину view и может задаваться разными значениями, как конкретными (x dp), так и константами (match_parent, wrap_content).

Листинг 1. Содержание файла layout_1_3.xml:

```
<?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">

    <View
        android:id="@+id/view"
        android:layout_width="50dp"
        android:layout_height="match_parent"
        android:background="@color/purple_200" />

    <Button
        android:id="@+id/button"
        android:layout_width="50dp"
        android:layout_height="match_parent"
        android:text="@string/button" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="@string/textview" />

</LinearLayout>
```

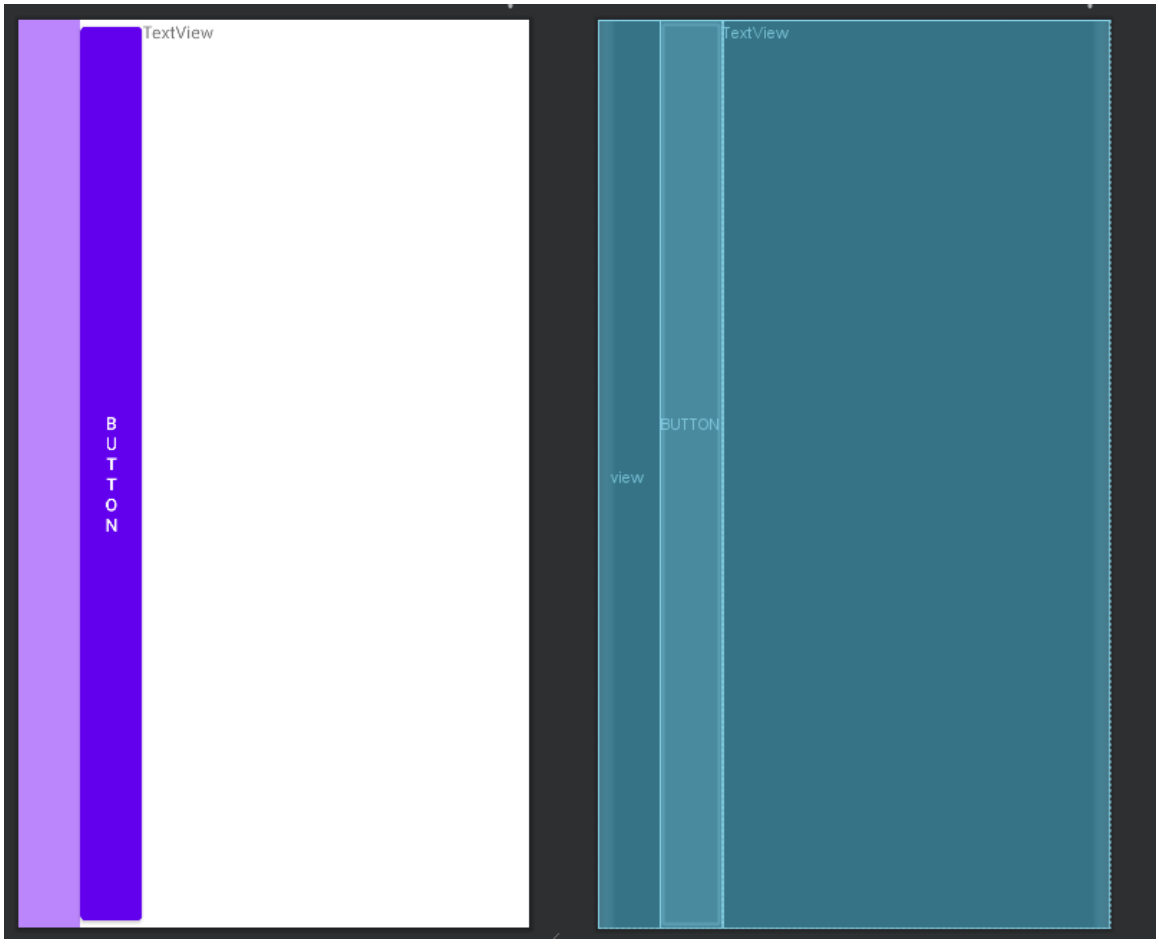

Рис. 4 Визуальное представление layout_1_3.

Layout_1_12

Для данного макета мы указываем атрибут *android:orientation* как *vertical*. Здесь нам также нужно использовать атрибут *android:layout_weight*, атрибут "важности" представления. Он позволяет элементу расширяться, чтобы заполнить любое оставшееся пространство в родительском представлении. Конкретно с помощью атрибута *android:layout_weight* можно задать структуру в котором элементы в пропорциональном отношении занимают место в исходном linear layout, пропорции зависят от значения этого атрибута у каждого элемента. И также атрибут *android:layout_marginBottom*, который задает отступ элемента снизу.

Листинг 2. Содержание файла layout_1_12.xml:

```
<?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">

    <View
        android:id="@+id/view"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:background="@color/purple_200" />

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:text="@string/button" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginBottom ="150dp"
        android:layout_weight="1"
        android:text="@string/textview" />

</LinearLayout>
```

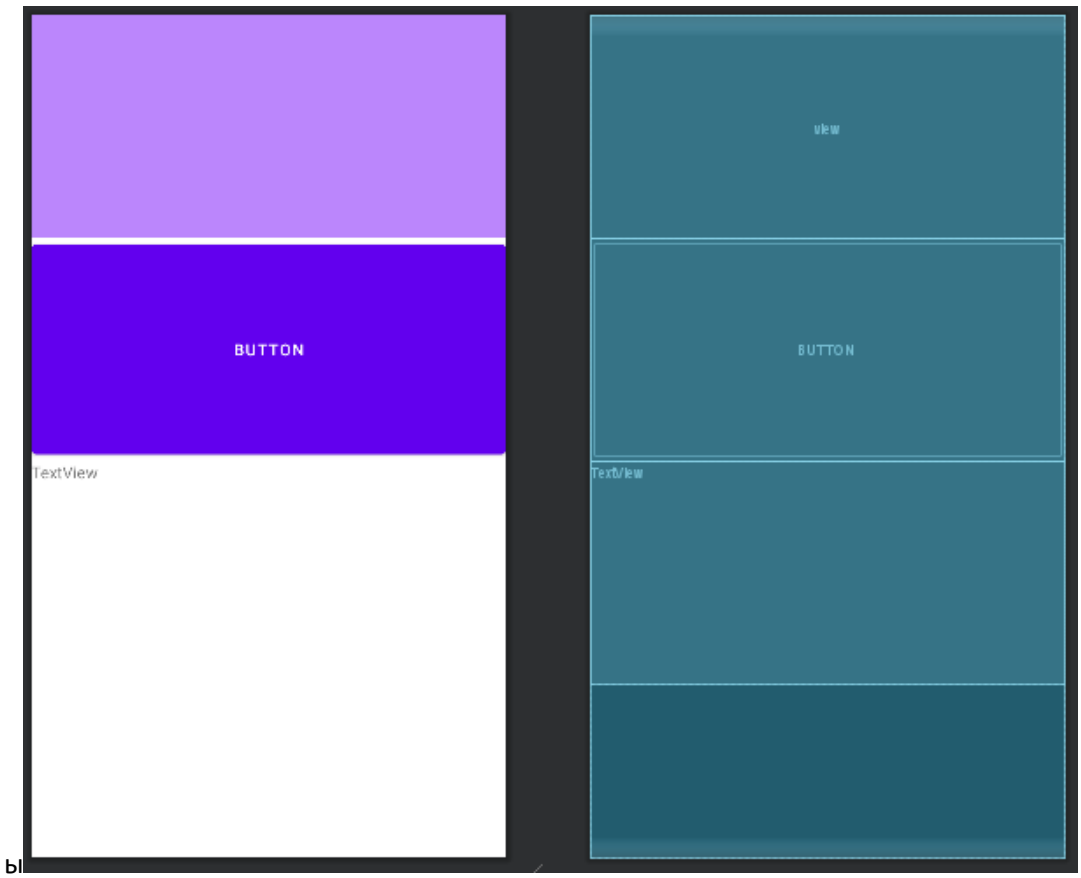

Рис. 5 Визуальное представление layout_1_12

Layout_1_12_alt

Мы можем построить такой же макет аналогичным способом, например вместо атрибута *android:layout_marginBottom*, будем использовать любой пустой ViewGroup с заданным атрибутом *android:layout_height* в 150dp.

Листинг 3. Содержание файла layout_1_12.xml:

```
<?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">

    <View
        android:id="@+id/view"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:background="@color/purple_200" />

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:text="@string/button" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:text="@string/textview" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="150dp" />

</LinearLayout>
```

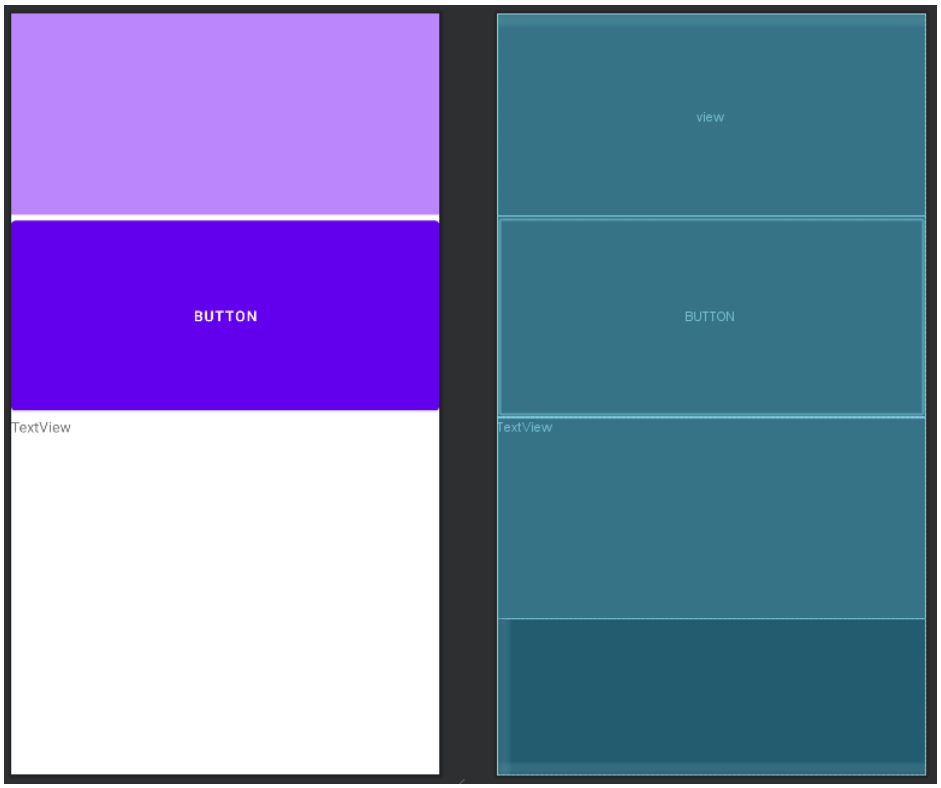

Рис. 6 Визуальное представление layout_1_12_alt

3.2. Задача 2. ConstraintLayout

Для решения этой задачи использовался Constraint Layout. Он позволяет «привязывать» края объекта к другим краям, фиксированные отступы.

Layout_2_3

Для каждого виджета мы должны как минимум привязать один край по вертикали и один по горизонтали. Для того чтобы виджет занимал все доступное ему пространство в атрибутах width и height, мы должны использовать константу 0dp она же match_constraints. (*You should not use match_parent for any view in a ConstraintLayout. Instead use "match constraints" (0dp)*) match_constraints View займет пространство, доступное между объектами, к которым он привязан. Если match_parent растягивает View по размеру родителя, то с match_constraints View займет пространство, доступное между объектами, к которым он привязан. Здесь мы также используем различные атрибуты привязки.

Листинг 4. Содержание файла layout_2_3.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <View
        android:id="@+id/view"
        android:layout_width="50dp"
        android:layout_height="0dp"
        android:background="@color/purple_200"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintBottom_toBottomOf="parent" />
    <Button
        android:id="@+id/button"
        android:layout_width="50dp"
        android:layout_height="0dp"
        android:text="@string/button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/view"
        app:layout_constraintTop_toTopOf="parent" />
    <TextView
        android:id="@+id/textView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:text="@string/textview"
        app:layout_constraintStart_toEndOf="@id/button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"/>
</androidx.constraintlayout.widget.ConstraintLayout>
```

В макете для каждого элемента используется разное количество привязок. Для корректного отображения для каждого элемента нужно как минимум 2 привязки (вертикальная и горизонтальная). Так в элементе button мы можем убрать одну из вертикальных привязок, при этом элемент вдруг станет растянут не на весь экран, хотя у соседнего View такого поведения не наблюдается, но при этом мы можем решить эту проблему поменяв атрибут *android:layout_height* на не нежелательный в constarint layout *"match_parent"*. Аналогично мы можем поступить и с элементом TextView, убрать одну вертикальную привязку(любую), и сделать *android:layout_height= ”match_parent”*. Но убрать уже одну горизонтальную привязку (любую) мы не можем, так как *match_parent* здесь уже займет все исходное горизонтальное пространство constarint layout.

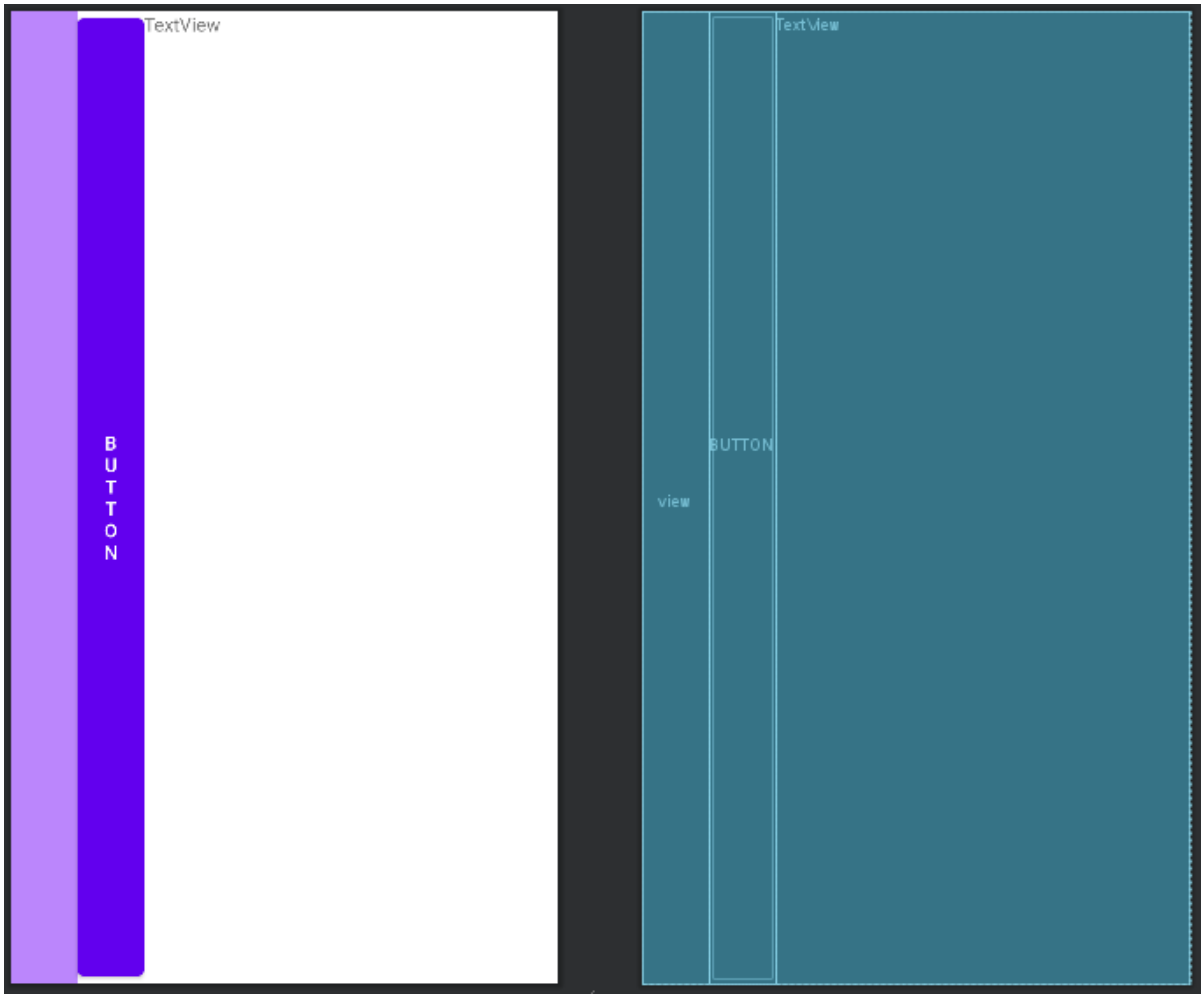

Рис. 7 Визуальное представление layout_2_3

Layout_2_12

Аналогичным образом верстаем наш второй макет, используя при этом атрибут отступа.

Листинг 5. Содержание файла layout_2_12.xml:

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

    <View
        android:id="@+id/view"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="@color/purple_200"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@+id/button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

    <Button
        android:id="@+id/button"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:text="@string/button"
        app:layout_constraintTop_toBottomOf="@id/view"
        app:layout_constraintBottom_toTopOf="@+id/textView"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />


    <TextView
        android:id="@+id/textView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:text="@string/textview"
        android:layout_marginBottom="150dp"
        app:layout_constraintTop_toBottomOf="@id/button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

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

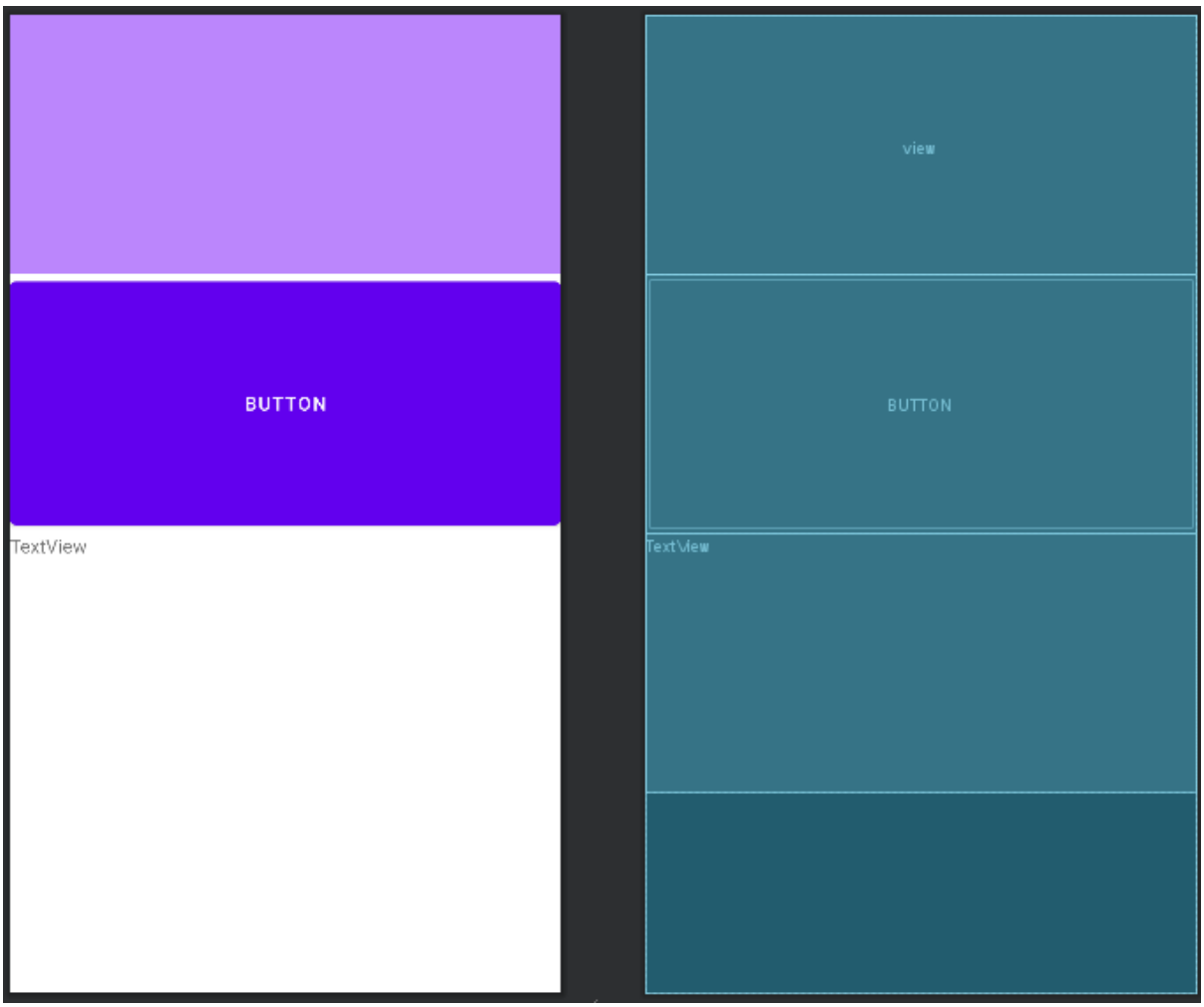

Рис. 8 Визуальное представление layout_2_12

3.3. Задача 3. ConstraintLayout

Layout_3_3

Для задания формы квадрата создадим внутри еще один ConstraintLayout и с помощью атрибута *app:layout_constraintDimensionRatio* зададим соотношение сторон равное 1. Для удобства верстки были использованы элементы *Guideline*. Guideline используется, чтобы привязывать к нему стороны view выравнивая их тем самым по одной линии.

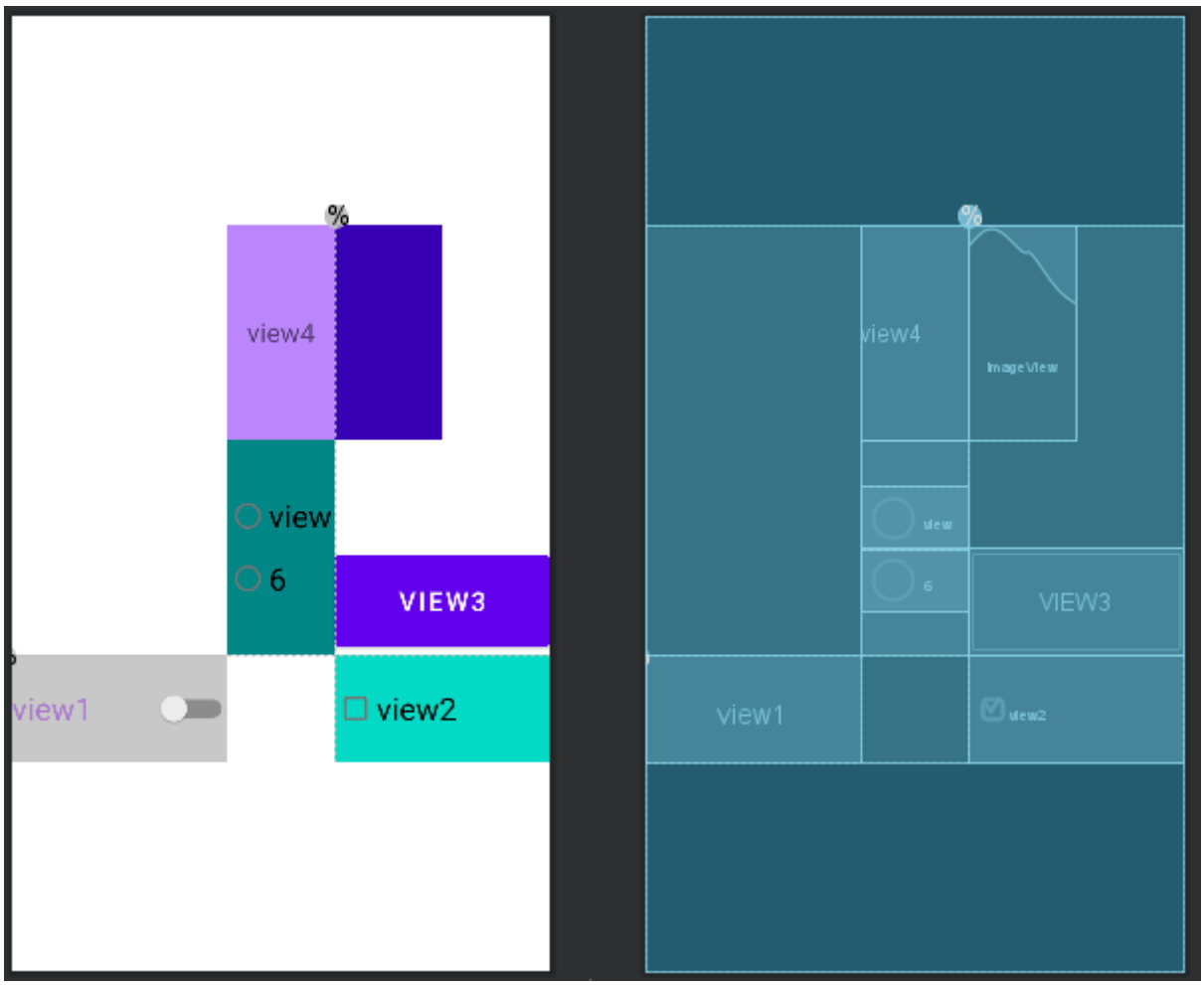

Рис. 9 Визуальное представление layout_3_3

Листинг 6. Содержание файла layout_3_3.xml:

```
<?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">

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

        <androidx.constraintlayout.widget.Guideline
            android:id="@+id/vertical_guideline"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            app:layout_constraintGuide_percent="0.6" />

        <androidx.constraintlayout.widget.Guideline
            android:id="@+id/horizontal_guideline"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            app:layout_constraintGuide_percent="0.8" />

        <androidx.appcompat.widget.SwitchCompat
            android:id="@+id/view1"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:background="@color/cardview_shadow_start_color"
            android:text="@string/view1"
            android:textColor="#A7C"
            android:textSize="24sp"

            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toStartOf="@id/view6"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/horizontal_guideline"/>

        <CheckBox
            android:id="@+id/view2"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:background="@color/teal_200"
            android:text="@string/view2"
            android:textSize="24sp"

            app:layout_constraintStart_toEndOf="@id/vertical_guideline"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintTop_toBottomOf="@id/horizontal_guideline" />

        <Button
            android:id="@+id/view3"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:text="@string/view3"
```

```
            android:textSize="20sp"

            app:layout_constraintDimensionRatio="2"
            app:layout_constraintStart_toEndOf="@id/vertical_guideline"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintBottom_toTopOf="@id/horizontal_guideline" />

        <TextView
            android:id="@+id/view4"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:background="@color/purple_200"
            android:gravity="center"
            android:text="@string/view4"
            android:textSize="20sp"

            app:layout_constraintBottom_toTopOf="@+id/view6"
            app:layout_constraintEnd_toStartOf="@id/vertical_guideline"
            app:layout_constraintStart_toStartOf="@id/view6"
            app:layout_constraintTop_toTopOf="parent" />

        <ImageView
            android:id="@+id/view5"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:background="@color/design_default_color_primary_dark"
            android:contentDescription="@string/view5"

            app:layout_constraintDimensionRatio="0.5"
            app:layout_constraintStart_toEndOf="@id/vertical_guideline"
            app:layout_constraintBottom_toTopOf="@id/view6"
            app:layout_constraintTop_toTopOf="parent" />

        <RadioGroup
            android:id="@+id/view6"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:gravity="center"
            android:background="@color/teal_700"

            app:layout_constraintDimensionRatio="0.5"
            app:layout_constraintEnd_toStartOf="@id/vertical_guideline"
            app:layout_constraintBottom_toTopOf="@id/horizontal_guideline"
            app:layout_constraintTop_toBottomOf="@id/view4" >

            <RadioButton
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="@string/view"
                android:textSize="24sp" />

            <RadioButton
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="@string/_6"
                android:textSize="24sp" />
        </RadioGroup>

    </androidx.constraintlayout.widget.ConstraintLayout>

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

4. Вывод

Была выполнена лабораторная работа. В среде разработки Android Studio были сверстаны предложенные макеты, при помощи LinearLayout и ConstraintLayout ViewGroup. Были изучены различные их атрибуты и особенности.

Так LinearLayout стоит использовать для совсем простеньких Layout’ов, где элементы расположены друг за другом. Во всех остальных случаях, лучше пользоваться ConstraintLayout, дабы избежать большой вложенной xml структуры.