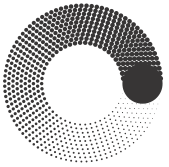
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

***Факультет информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.04.02 «Информационные системы и технологии»,**

**ЛАБОРАТОРНАЯ РАБОТА № 6**

**Дисциплина: Разработка мобильных приложений для Android**

**Тема: Работа с 2D-графикой**

**Выполнил(а): студент(ка) группы 224-371** Лейн Ф.Е.

(Фамилия И.О.)

**Проверил(а):** Якубовский К.И.

(Фамилия И.О. )

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**ЦЕЛЬ РАБОТЫ**

Разработать приложение, содержащие различные объекты ShapeDrawable, анимированные с использованием анимации преобразованием и кадровой анимации.

**ХОД РАБОТЫ**

В начале была выполнена разработка макета (формы) приложения, результат показан на рисунке 1. Весь экран приложения занимает объект ImageView. Исходный код макета находится в приложении В.

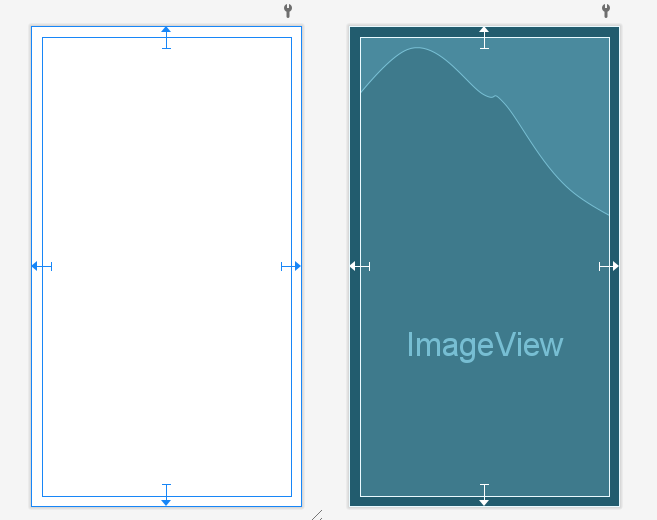


Рисунок 1 – Разработанный макет приложения

На рисунке 2 показаны разработанный файл меню, реализующее выбор фигуры для отрисовки и анимации. Исходный код файла menu.xml находится в приложении Б.

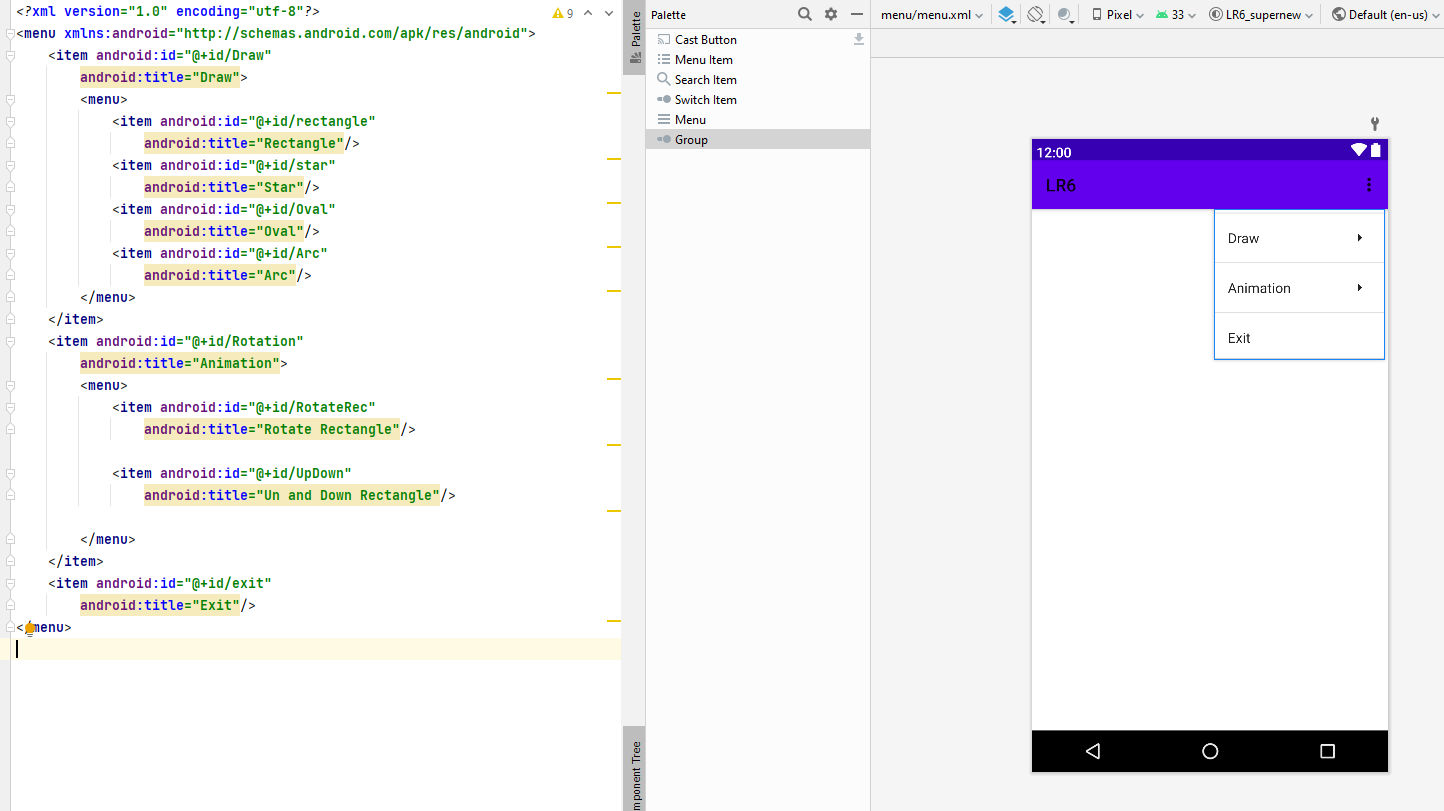


Рисунок 2 – Меню для выбора объекта отрисовки и анимации

Для реализации поставленных задач необходимо иметь изображение в папке drawable, что показано на рисунке 3.

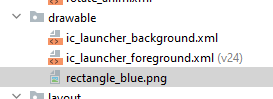


Рисунок 3 – Файл синего прямоугольника

В листинге 1 показан процесс создания прямоугольника, используя класс RectShape и произвольный размер (200x200)

Листинг 1 – Отрисовка прямоугольника

*// Draw rectangle***case** R.id.***rectangle***:  
 shapeDrawable = **new** ShapeDrawable(**new** RectShape());  
 shapeDrawable.setIntrinsicHeight(200);  
 shapeDrawable.setIntrinsicWidth(200);  
 shapeDrawable.getPaint().setColor(Color.***GREEN***);  
 **break**;

В листинге 2 показан процесс создания звезды, используя класс Path и заданные точки.

Листинг 2 – Отрисовка звезды

*// Draw star***case** R.id.***star***:  
 Path path = **new** Path();  
 path.moveTo(50, 0);  
 path.lineTo(25, 100);  
 path.lineTo(100, 50);  
 path.lineTo(0, 50);  
 path.lineTo(75, 100);  
 path.lineTo(50, 0);  
  
 shapeDrawable = **new** ShapeDrawable(**new** PathShape(path, 100, 100));  
 shapeDrawable.setIntrinsicHeight(300);  
 shapeDrawable.setIntrinsicWidth(300);  
 shapeDrawable.getPaint().setColor(Color.***BLUE***);  
 shapeDrawable.getPaint().setStyle(Paint.Style.***STROKE***);  
 **break**;

В листинге 3 показан процесс создания овала, используя класс RoundRectShape.

Листинг 3 – Отрисовка овала

*// Draw oval***case** R.id.***Oval***:  
 **float**[] outR = **new float**[]{10, 10, 50, 50, 20, 20, 20, 20};  
 RectF rectF = **new** RectF(10, 10, 10, 10);  
 **float**[] inR = **new float**[]{40, 40, 40, 40, 40, 40, 40, 40};  
 shapeDrawable = **new** ShapeDrawable(**new** RoundRectShape(outR, rectF, inR));  
 shapeDrawable.setIntrinsicHeight(300);  
 shapeDrawable.setIntrinsicWidth(300);  
 shapeDrawable.getPaint().setColor(Color.***BLUE***);  
 **break**;

В листинге 4 показан процесс создания дуги, используя класс ArcShape.

Листинг 4 – Отрисовка дуги

*// Draw arc***case** R.id.***Arc***:  
 shapeDrawable = **new** ShapeDrawable(**new** ArcShape(0, 255));  
 shapeDrawable.setIntrinsicHeight(300);  
 shapeDrawable.setIntrinsicWidth(300);  
 shapeDrawable.getPaint().setColor(Color.***RED***);  
 **break**;

В листинге 5 показан процесс создания прямоугольника, используя класс RectShape для последующей анимации произвольного размера (400x600).

Листинг 5 – Отрисовка прямоугольника для анимаций

*// Animated rectangle***case** R.id.***RotateRec***:  
**case** R.id.***UpDown***:  
 Drawable drawable = ContextCompat.*getDrawable*(**this**, R.drawable.***rectangle\_blue***);  
 BitmapDrawable bitmapDrawable = (BitmapDrawable) drawable;  
 shapeDrawable = **new** ShapeDrawable();  
 shapeDrawable.setShape(**new** RectShape());  
 shapeDrawable.setIntrinsicHeight(400);  
 shapeDrawable.setIntrinsicWidth(600);  
 shapeDrawable.getPaint().setShader(**new** BitmapShader(bitmapDrawable.getBitmap(),  
 Shader.TileMode.***REPEAT***, Shader.TileMode.***REPEAT***));  
 **break**;

В листинге 6 показан алгоритм работы анимаций, которые выполняются при помощи класса ObjectAnimator. В начале, для избегания конфликта между анимациями, все текущие анимации останавливаются, затем создаются анимации при помощи ObjectAnimator. Также, создаётся слушатель анимации, реализующий методы onAnimationStart(), onAnimationEnd(). Также, предусмотрен возврат в исходное положение при завершении анимации, т.к. анимации будут проигрываться бесконечно.

Листинг 6 – создание и обработка анимаций

*/\*\*  
 \* Starts rotation animation  
 \*/***private void** startRotationAnimation() {  
 *// Stop all animations* stopAnimations();  
  
 *// Create 1s rotation* **rotationAnimator** = ObjectAnimator.*ofFloat*(**imageView**, **"rotation"**, 0f, 360f);  
 **rotationAnimator**.setDuration(1000);  
 **rotationAnimator**.setRepeatCount(ObjectAnimator.***INFINITE***);  
 **rotationAnimator**.addListener(**new** Animator.AnimatorListener() {  
 @Override  
 **public void** onAnimationStart(Animator animator) {}  
  
 @Override  
 **public void** onAnimationEnd(Animator animator) {  
 *// Repeat from 0 after finish* **imageView**.setRotation(0f);  
 }  
  
 @Override  
 **public void** onAnimationCancel(Animator animator) {}  
  
 @Override  
 **public void** onAnimationRepeat(Animator animator) {}  
 });  
  
 *// Start animation* **animatorSet** = **new** AnimatorSet();  
 **animatorSet**.play(**rotationAnimator**);  
 **animatorSet**.start();  
}  
  
*/\*\*  
 \* Starts jumping animation (Y)  
 \*/***private void** startJumpAnimation() {  
 *// Stop all animations* stopAnimations();  
  
 *// Set start and stop Y positions* **float** startY = **imageView**.getY();  
 **float** endY = startY - 100;  
  
 *// Create 0.5s jumping animation* **jumpAnimator** = ObjectAnimator.*ofFloat*(**imageView**, **"y"**, startY, endY);  
 **jumpAnimator**.setDuration(500);  
 **jumpAnimator**.setInterpolator(**new** AccelerateDecelerateInterpolator());  
 **jumpAnimator**.setRepeatMode(ObjectAnimator.***REVERSE***);  
 **jumpAnimator**.setRepeatCount(ObjectAnimator.***INFINITE***);  
 **jumpAnimator**.addListener(**new** Animator.AnimatorListener() {  
 @Override  
 **public void** onAnimationStart(Animator animator) {}  
  
 @Override  
 **public void** onAnimationEnd(Animator animator) {  
 *// Repeat from start after finish* **imageView**.setY(startY);  
 }  
  
 @Override  
 **public void** onAnimationCancel(Animator animator) {}  
  
 @Override  
 **public void** onAnimationRepeat(Animator animator) {}  
 });  
  
 *// Start animation* **animatorSet** = **new** AnimatorSet();  
 **animatorSet**.play(**jumpAnimator**);  
 **animatorSet**.start();  
}

На рисунке 4 показан скриншот примера работы приложения в режиме анимации-вращения.

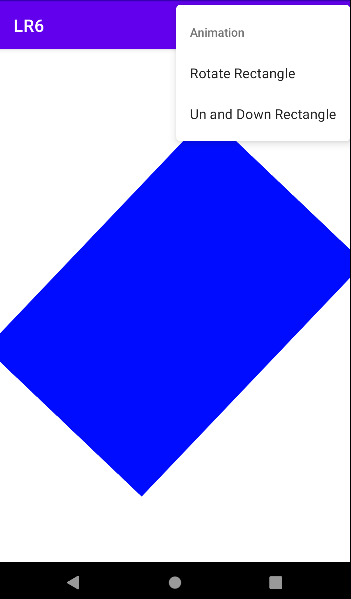


Рисунок 4 – Пример работы приложения (анимация вращающегося прямоугольника)

**ВЫВОД**

В результате выполнения данной лабораторной работы были изучены навыки отрисовки объектов в классе ShapeDrawable, а также навыки создания анимаций.

**Приложение А. Листинг кода основного класса приложения**

**package** com.labs.lr6;  
  
**import** android.animation.Animator;  
**import** android.animation.AnimatorSet;  
**import** android.animation.ObjectAnimator;  
**import** android.graphics.BitmapShader;  
**import** android.graphics.Color;  
**import** android.graphics.Paint;  
**import** android.graphics.Path;  
**import** android.graphics.RectF;  
**import** android.graphics.Shader;  
**import** android.graphics.drawable.BitmapDrawable;  
**import** android.graphics.drawable.Drawable;  
**import** android.graphics.drawable.ShapeDrawable;  
**import** android.graphics.drawable.shapes.ArcShape;  
**import** android.graphics.drawable.shapes.PathShape;  
**import** android.graphics.drawable.shapes.RectShape;  
**import** android.graphics.drawable.shapes.RoundRectShape;  
**import** android.os.Bundle;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
**import** android.view.animation.AccelerateDecelerateInterpolator;  
**import** android.widget.ImageView;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
**import** androidx.core.content.ContextCompat;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 **private** ImageView **imageView**;  
 **private** ObjectAnimator **rotationAnimator**;  
 **private** ObjectAnimator **jumpAnimator**;  
 **private** AnimatorSet **animatorSet**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// Initialize ImageView* **imageView** = findViewById(R.id.***imageView***);  
 }  
  
 */\*\*  
 \* Generates menu from menu.xml  
 \** ***@param menu*** *Menu object  
 \** ***@return*** *true  
 \*/* @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu***, menu);  
 **return true**;  
 }  
  
 */\*\*  
 \* Handles selected action from menu  
 \** ***@param item*** *MenuItem  
 \** ***@return*** *\*/* @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 *// Create new shape object* ShapeDrawable shapeDrawable;  
  
 **switch** (item.getItemId()) {  
 *// Draw rectangle* **case** R.id.***rectangle***:  
 shapeDrawable = **new** ShapeDrawable(**new** RectShape());  
 shapeDrawable.setIntrinsicHeight(200);  
 shapeDrawable.setIntrinsicWidth(200);  
 shapeDrawable.getPaint().setColor(Color.***GREEN***);  
 **break**;  
  
 *// Draw star* **case** R.id.***star***:  
 Path path = **new** Path();  
 path.moveTo(50, 0);  
 path.lineTo(25, 100);  
 path.lineTo(100, 50);  
 path.lineTo(0, 50);  
 path.lineTo(75, 100);  
 path.lineTo(50, 0);  
  
 shapeDrawable = **new** ShapeDrawable(**new** PathShape(path, 100, 100));  
 shapeDrawable.setIntrinsicHeight(300);  
 shapeDrawable.setIntrinsicWidth(300);  
 shapeDrawable.getPaint().setColor(Color.***BLUE***);  
 shapeDrawable.getPaint().setStyle(Paint.Style.***STROKE***);  
 **break**;  
  
 *// Draw oval* **case** R.id.***Oval***:  
 **float**[] outR = **new float**[]{10, 10, 50, 50, 20, 20, 20, 20};  
 RectF rectF = **new** RectF(10, 10, 10, 10);  
 **float**[] inR = **new float**[]{40, 40, 40, 40, 40, 40, 40, 40};  
 shapeDrawable = **new** ShapeDrawable(**new** RoundRectShape(outR, rectF, inR));  
 shapeDrawable.setIntrinsicHeight(300);  
 shapeDrawable.setIntrinsicWidth(300);  
 shapeDrawable.getPaint().setColor(Color.***BLUE***);  
 **break**;  
  
 *// Draw arc* **case** R.id.***Arc***:  
 shapeDrawable = **new** ShapeDrawable(**new** ArcShape(0, 255));  
 shapeDrawable.setIntrinsicHeight(300);  
 shapeDrawable.setIntrinsicWidth(300);  
 shapeDrawable.getPaint().setColor(Color.***RED***);  
 **break**;  
  
 *// Animated rectangle* **case** R.id.***RotateRec***:  
 **case** R.id.***UpDown***:  
 Drawable drawable = ContextCompat.*getDrawable*(**this**, R.drawable.***rectangle\_blue***);  
 BitmapDrawable bitmapDrawable = (BitmapDrawable) drawable;  
 shapeDrawable = **new** ShapeDrawable();  
 shapeDrawable.setShape(**new** RectShape());  
 shapeDrawable.setIntrinsicHeight(400);  
 shapeDrawable.setIntrinsicWidth(600);  
 shapeDrawable.getPaint().setShader(**new** BitmapShader(bitmapDrawable.getBitmap(),  
 Shader.TileMode.***REPEAT***, Shader.TileMode.***REPEAT***));  
 **break**;  
  
 *// Close app* **case** R.id.***exit***:  
 finish();  
 **return true**;  
  
 **default**:  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 *// If we have anything to draw  
 // Start or stop animations* **if** (item.getItemId() == R.id.***RotateRec***)  
 startRotationAnimation();  
 **else if** (item.getItemId() == R.id.***UpDown***)  
 startJumpAnimation();  
 **else** stopAnimations();  
  
 *// Set shape to imageView* **imageView**.setImageDrawable(shapeDrawable);  
 **return true**;  
 }  
  
 */\*\*  
 \* Starts rotation animation  
 \*/* **private void** startRotationAnimation() {  
 *// Stop all animations* stopAnimations();  
  
 *// Create 1s rotation* **rotationAnimator** = ObjectAnimator.*ofFloat*(**imageView**, **"rotation"**, 0f, 360f);  
 **rotationAnimator**.setDuration(1000);  
 **rotationAnimator**.setRepeatCount(ObjectAnimator.***INFINITE***);  
 **rotationAnimator**.addListener(**new** Animator.AnimatorListener() {  
 @Override  
 **public void** onAnimationStart(Animator animator) {}  
  
 @Override  
 **public void** onAnimationEnd(Animator animator) {  
 *// Repeat from 0 after finish* **imageView**.setRotation(0f);  
 }  
  
 @Override  
 **public void** onAnimationCancel(Animator animator) {}  
  
 @Override  
 **public void** onAnimationRepeat(Animator animator) {}  
 });  
  
 *// Start animation* **animatorSet** = **new** AnimatorSet();  
 **animatorSet**.play(**rotationAnimator**);  
 **animatorSet**.start();  
 }  
  
 */\*\*  
 \* Starts jumping animation (Y)  
 \*/* **private void** startJumpAnimation() {  
 *// Stop all animations* stopAnimations();  
  
 *// Set start and stop Y positions* **float** startY = **imageView**.getY();  
 **float** endY = startY - 100;  
  
 *// Create 0.5s jumping animation* **jumpAnimator** = ObjectAnimator.*ofFloat*(**imageView**, **"y"**, startY, endY);  
 **jumpAnimator**.setDuration(500);  
 **jumpAnimator**.setInterpolator(**new** AccelerateDecelerateInterpolator());  
 **jumpAnimator**.setRepeatMode(ObjectAnimator.***REVERSE***);  
 **jumpAnimator**.setRepeatCount(ObjectAnimator.***INFINITE***);  
 **jumpAnimator**.addListener(**new** Animator.AnimatorListener() {  
 @Override  
 **public void** onAnimationStart(Animator animator) {}  
  
 @Override  
 **public void** onAnimationEnd(Animator animator) {  
 *// Repeat from start after finish* **imageView**.setY(startY);  
 }  
  
 @Override  
 **public void** onAnimationCancel(Animator animator) {}  
  
 @Override  
 **public void** onAnimationRepeat(Animator animator) {}  
 });  
  
 *// Start animation* **animatorSet** = **new** AnimatorSet();  
 **animatorSet**.play(**jumpAnimator**);  
 **animatorSet**.start();  
 }  
  
 **private void** stopAnimations() {  
 *// Cancel all animations* **if** (**animatorSet** != **null**) {  
 **animatorSet**.cancel();  
 **animatorSet** = **null**;  
 }  
  
 **if** (**rotationAnimator** != **null**) {  
 **rotationAnimator**.cancel();  
 **rotationAnimator** = **null**;  
 }  
  
 **if** (**jumpAnimator** != **null**) {  
 **jumpAnimator**.cancel();  
 **jumpAnimator** = **null**;  
 }  
 }  
}

**Приложение Б. Листинг кода файла menu.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**menu xmlns:android="http://schemas.android.com/apk/res/android"**>  
 <**item android:id="@+id/Draw"  
 android:title="Draw"**>  
 <**menu**>  
 <**item android:id="@+id/rectangle"  
 android:title="Rectangle"**/>  
 <**item android:id="@+id/star"  
 android:title="Star"**/>  
 <**item android:id="@+id/Oval"  
 android:title="Oval"**/>  
 <**item android:id="@+id/Arc"  
 android:title="Arc"**/>  
 </**menu**>  
 </**item**>  
 <**item android:id="@+id/Rotation"  
 android:title="Animation"**>  
 <**menu**>  
 <**item android:id="@+id/RotateRec"  
 android:title="Rotate Rectangle"**/>  
  
 <**item android:id="@+id/UpDown"  
 android:title="Un and Down Rectangle"**/>  
  
 </**menu**>  
 </**item**>  
 <**item android:id="@+id/exit"  
 android:title="Exit"**/>  
</**menu**>

**Приложение В. Листинг кода разметки Activity приложения**

*<?***xml version="1.0" encoding="utf-8"***?>*<**androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/root"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity"**>  
  
  
 <**ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="16dp"  
 android:layout\_marginTop="16dp"  
 android:layout\_marginEnd="16dp"  
 android:layout\_marginBottom="16dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:ignore="ContentDescription"** />  
</**androidx.constraintlayout.widget.ConstraintLayout**>