Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем

и программной инженерии

**Лабораторные работы № 8**

**по дисциплине**

**«Технологии разработки мобильных приложений»**

Работа с камерой на операционной системе Android

Выполнил:

ст. гр. ПРИ-120

К. А. Борисова

Принял:

А. И. Рощина

Владимир, 2023

**Цель работы**

Разработать мобильное приложение, которое будет использовать камеру устройства для вывода изображения на экран. Научиться обрабатывать событие поворота устройства и подгонять изображение под экран.

**Ход работы**

Повторим код из методических указаний и добавим возможность делать фотографию по нажатию на кнопку. Листинг программы:

package com.example.lab8;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.graphics.Matrix;  
import android.graphics.RectF;  
import android.os.Bundle;  
import android.view.Display;  
import android.view.SurfaceHolder;  
import android.view.SurfaceView;  
import android.view.View;  
import android.view.Window;  
import android.view.WindowManager;  
import android.hardware.Camera;  
import android.view.Surface;  
import android.widget.Button;  
  
import java.io.File;  
import java.io.FileOutputStream;  
import java.io.IOException;  
  
public class MainActivity extends AppCompatActivity {  
  
 SurfaceView sv;  
 SurfaceHolder holder;  
 HolderCallback holderCallback;  
 Camera camera;  
 File photo;  
 Button buttonPhoto;  
  
  
 final int CAMERA\_ID = 0;  
 final boolean FULL\_SCREEN = true;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);  
 setContentView(R.layout.*activity\_main*);  
  
 sv = (SurfaceView) findViewById(R.id.*surfaceView*);  
 buttonPhoto = (Button) findViewById(R.id.*buttonPhoto*);  
 buttonPhoto.setOnClickListener(v->clickPhoto(v));  
 holder = sv.getHolder();  
 holder.setType(SurfaceHolder.*SURFACE\_TYPE\_PUSH\_BUFFERS*);  
 holderCallback = new HolderCallback();  
 holder.addCallback(holderCallback);  
  
 }  
  
 @Override  
 protected void onResume(){  
 super.onResume();  
 camera = camera.*open*(CAMERA\_ID);  
 setPreviewSize(FULL\_SCREEN);  
 }  
  
 @Override  
 protected void onPause(){  
 super.onPause();  
 if(camera!=null)  
 camera.release();  
 camera = null;  
 }  
 class HolderCallback implements SurfaceHolder.Callback {  
 @Override  
 public void surfaceCreated(@NonNull SurfaceHolder surfaceHolder) {  
 try {  
 camera.setPreviewDisplay(holder);  
 camera.startPreview();  
 }catch (IOException e){  
 e.printStackTrace();  
 }  
 }  
  
 @Override  
 public void surfaceChanged(@NonNull SurfaceHolder surfaceHolder, int i, int i1, int i2) {  
 camera.stopPreview();  
 setCameraDisplayOrientation(CAMERA\_ID);  
 try{  
 camera.setPreviewDisplay((holder));  
 camera.startPreview();  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
  
 @Override  
 public void surfaceDestroyed(@NonNull SurfaceHolder surfaceHolder) {  
  
 }  
 }  
  
 public void clickPhoto(View view) {  
 camera.takePicture(null, null, new Camera.PictureCallback() {  
 @Override  
 public void onPictureTaken(byte[] data, Camera camera) {  
 try {  
 FileOutputStream fileOutputStream = null;  
 fileOutputStream = openFileOutput("photo"+ Math.*random*() +".jpg", *MODE\_PRIVATE*);  
 fileOutputStream.write(data);  
 fileOutputStream.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 };  
 });  
 };  
  
  
 public void setPreviewSize(boolean fullScreen){  
 Display display = getWindowManager().getDefaultDisplay();  
 boolean widthIsMax = display.getWidth() > display.getHeight();  
  
 Camera.Size size = camera.getParameters().getPreviewSize();  
  
 RectF rectDisplay = new RectF();  
 RectF rectPreview = new RectF();  
  
 rectDisplay.set(0,0,display.getWidth(),display.getHeight());  
  
 if(widthIsMax){  
 rectPreview.set(0,0, size.width,size.height);  
 }else{  
 rectPreview.set(0,0,size.height,size.width);  
 }  
  
 Matrix matrix = new Matrix();  
  
 if(!fullScreen){  
 matrix.setRectToRect(rectPreview,rectDisplay,Matrix.ScaleToFit.*START*);  
 }else{  
 matrix.setRectToRect(rectDisplay,rectPreview,Matrix.ScaleToFit.*START*);  
 matrix.invert(matrix);  
 }  
  
 matrix.mapRect(rectPreview);  
  
 sv.getLayoutParams().height = (int) (rectPreview.bottom);  
 sv.getLayoutParams().width = (int) (rectPreview.right);  
 }  
  
 void setCameraDisplayOrientation(int cameraId){  
 int rotation = getWindowManager().getDefaultDisplay().getRotation();  
 int degrees = 0;  
  
 switch (rotation) {  
 case Surface.*ROTATION\_0*:  
 degrees = 0;  
 break;  
 case Surface.*ROTATION\_90*:  
 degrees = 90;  
 break;  
 case Surface.*ROTATION\_180*:  
 degrees = 180;  
 break;  
 case Surface.*ROTATION\_270*:  
 degrees = 270;  
 break;  
 }  
  
 int result = 0;  
  
 Camera.CameraInfo info = new Camera.CameraInfo();  
 Camera.*getCameraInfo*(cameraId,info);  
  
 if(info.facing == Camera.CameraInfo.*CAMERA\_FACING\_BACK*){  
 result = ((360 - degrees) + info.orientation);  
 }else if(info.facing == Camera.CameraInfo.*CAMERA\_FACING\_FRONT*){  
 result = ((360 - degrees) - info.orientation);  
 result += 360;  
 }  
  
 result = result % 360;  
 camera.setDisplayOrientation(result);  
 }  
}

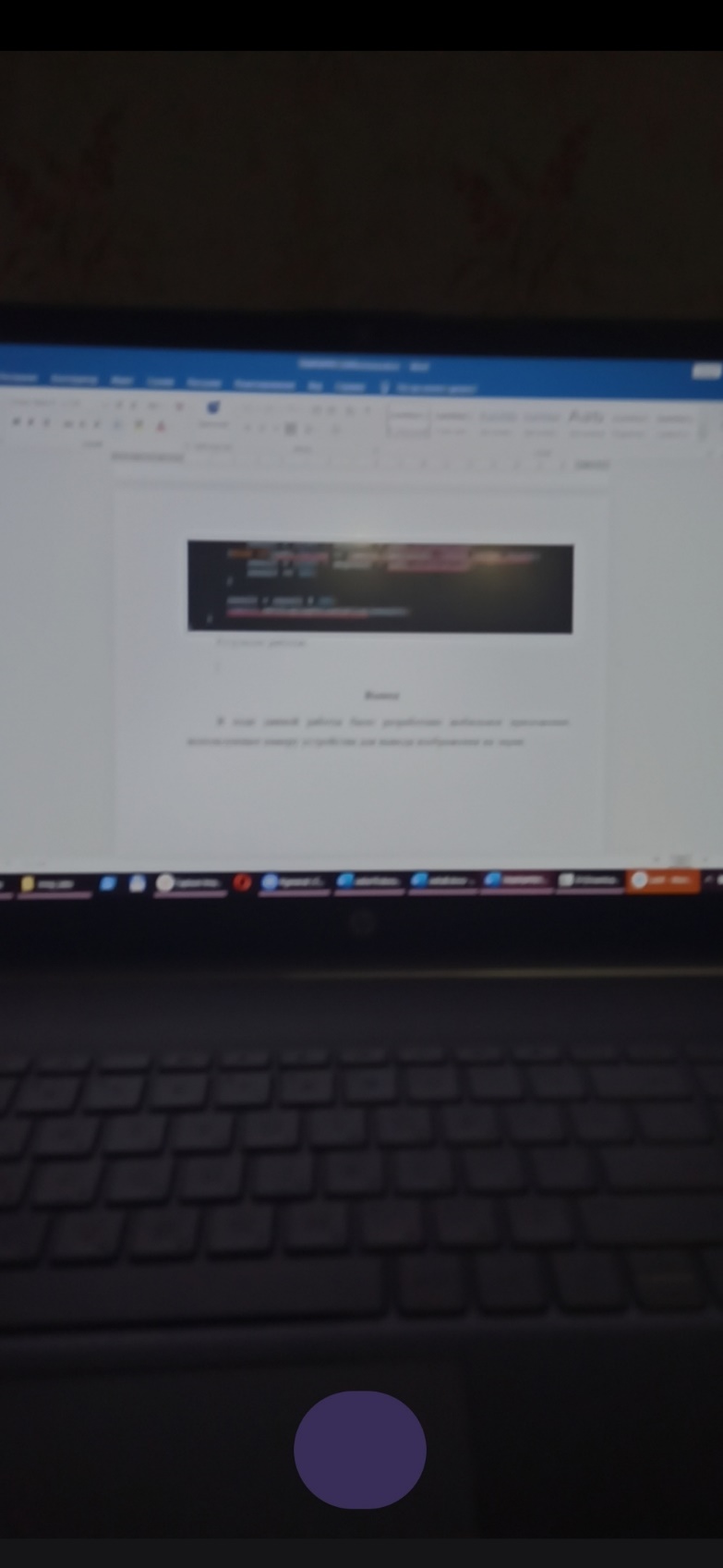
Результат работы:

Рисунок . Приложение

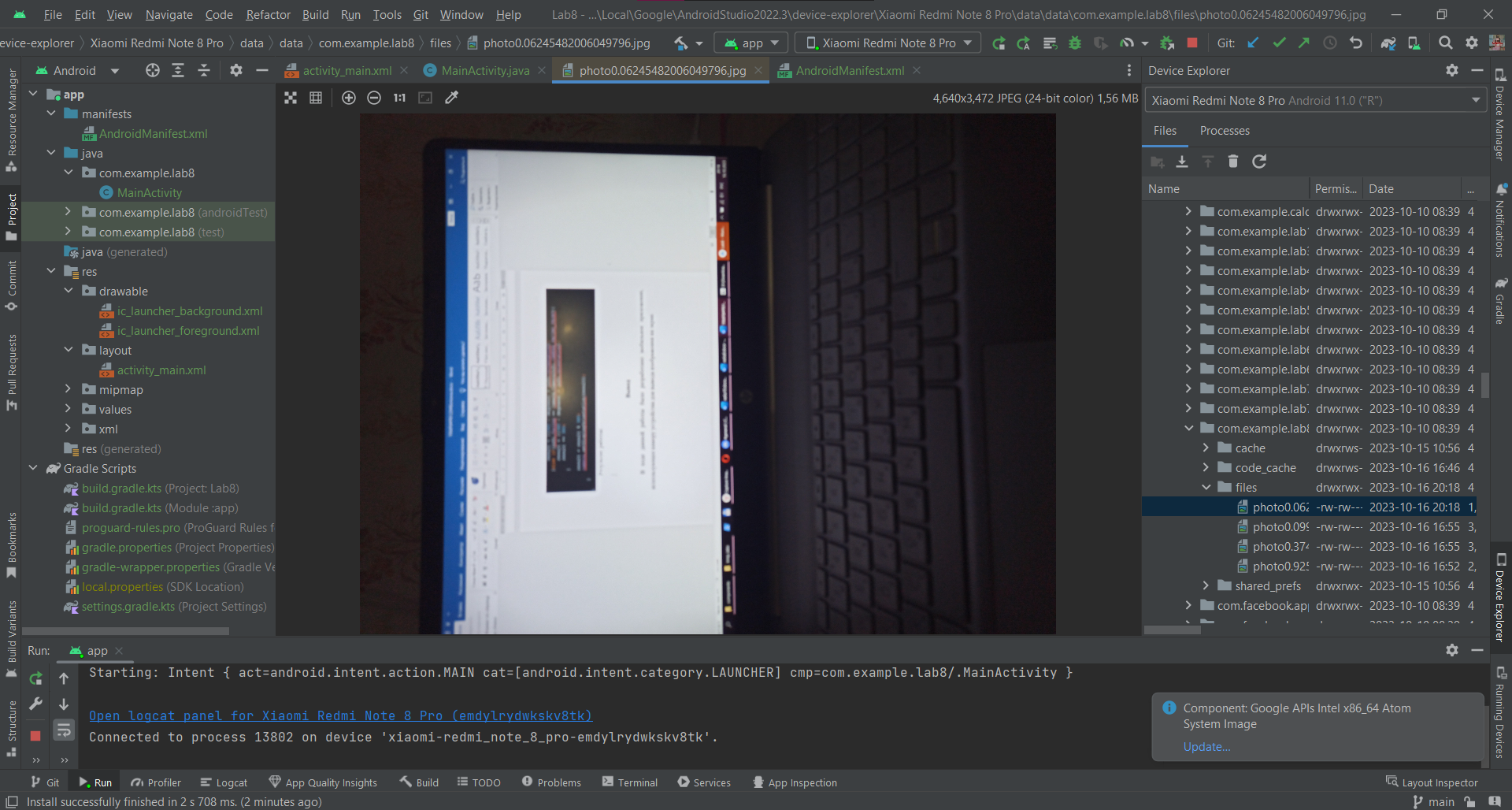


Рисунок . Сохраненное изображение

**Вывод**

В ходе данной работы было разработано мобильное приложение, использующее камеру устройства для вывода изображения на экран.