Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №8

по дисциплине "Технологии разработки мобильных приложений"

ТЕМА РАБОТЫ:

Вывод изображения с камеры

Выполнил:

студент гр. ПРИ-120

Парахин К.В.

Принял:

преподаватель кафедры ИСПИ

Рощина А.И.

Владимир 2023 г.

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.

Выполнение работы

Выполнение примера из методички

package com.example.lab6\_parakhin;  
  
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.Window;  
import android.view.WindowManager;  
import android.hardware.Camera;  
import android.view.Surface;  
import java.io.IOException;  
  
public class MainActivity extends AppCompatActivity {  
  
 SurfaceView sv;  
 SurfaceHolder holder;  
 HolderCallback holderCallback;  
 Camera camera;  
  
 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*);  
 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 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);  
 }  
}

Задание (Вариант 1)

1. Выполнить одно из следующих двух заданий:

1.1. Сделать снимок

1.2. Записать видео

Попробуем реализовать получение снимка с камеры мобильного устройства (и вывод полученного файла на экран).

Сначала требуется добавить разрешения на использование камеры в файле AndroidManifest.xml:

<uses-permission android:name="android.permission.CAMERA"/>  
<uses-feature android:name="android.hardware.camera" android:required="false"/>  
<uses-feature android:name="android.hardware.camera.autofocus"/>

Добавим объекты SurfaceView и Button на активность activity\_main.xml:

<SurfaceView  
 android:id="@+id/surfaceView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 android:layout\_marginStart="20dp"  
 android:layout\_marginTop="20dp"  
 android:layout\_marginEnd="20dp"  
 android:layout\_marginBottom="20dp"/>  
  
<Button  
 android:id="@+id/button"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:text="@string/photo"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@id/surfaceView"  
 android:layout\_marginStart="50dp"  
 android:layout\_marginEnd="50dp"  
 android:layout\_marginTop="50dp"/>

Код приложения-камеры:

package com.example.lab8\_camera\_parakhin;  
  
import static android.Manifest.permission\_group.*CAMERA*;  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
import androidx.core.content.ContextCompat;  
  
import android.content.pm.PackageManager;  
import android.graphics.Matrix;  
import android.graphics.RectF;  
import android.os.Bundle;  
import android.os.Environment;  
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 surfaceView;  
 SurfaceHolder holder;  
 HolderCallback holderCallback;  
 Camera camera;  
 File photoFile;  
 Button buttonTakePhoto;  
  
 final int CAMERA\_ID = 0;  
 final boolean FULL\_SCREEN = true;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
 File pictures = Environment  
 .*getExternalStoragePublicDirectory*(Environment.*DIRECTORY\_PICTURES*);  
 photoFile = new File(pictures, "photo.jpg");  
  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);  
 setContentView(R.layout.*activity\_main*);  
  
 surfaceView = (SurfaceView) findViewById(R.id.*surfaceView*);  
 buttonTakePhoto = (Button) findViewById(R.id.*button*);  
 buttonTakePhoto.setOnClickListener(v->onClickPicture(v));  
 holder = surfaceView.getHolder();  
 holder.setType(SurfaceHolder.*SURFACE\_TYPE\_PUSH\_BUFFERS*);  
 holderCallback = new HolderCallback();  
 holder.addCallback(holderCallback);  
  
 if (!checkPermission()){  
 requestPermission();  
 }  
 }  
  
 private boolean checkPermission(){  
 return ContextCompat.*checkSelfPermission*(getApplicationContext(), *CAMERA*)  
 == PackageManager.*PERMISSION\_GRANTED*;  
 }  
  
 private void requestPermission(){  
 ActivityCompat.*requestPermissions*(this, new String[] {*CAMERA*}, 200);  
 }  
  
 @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 onClickPicture(View view) {  
 camera.takePicture(null, null, new Camera.PictureCallback() {  
 @Override  
 public void onPictureTaken(byte[] data, Camera camera) {  
 try {  
 FileOutputStream fos = new FileOutputStream(photoFile);  
 fos.write(data);  
 fos.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);  
  
 surfaceView.getLayoutParams().height = (int) (rectPreview.bottom);  
 surfaceView.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);  
 }  
}

Тестирование приложения

При тестировании не удалось пофиксить ошибку, указанную ниже в логах.

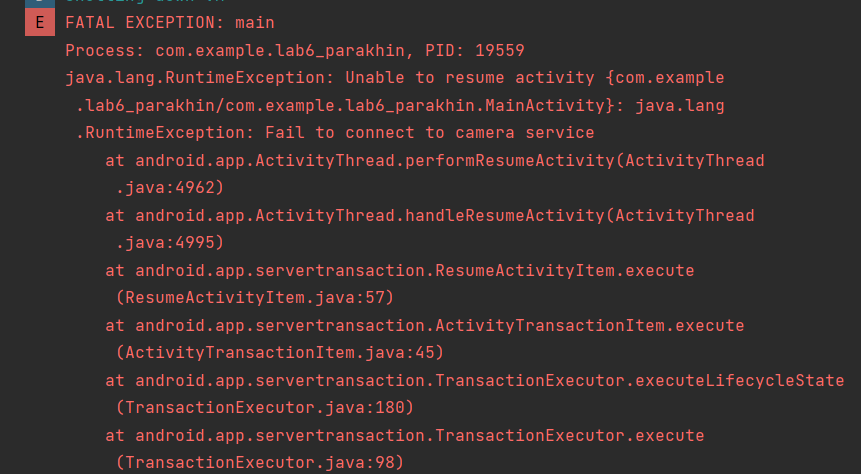


Рисунок 1. Ошибка при попытке запустить приложение

Вывод

В результате выполнения работы я попытался использовать камеру мобильного устройства для вывода изображений на экран.