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

**Лабораторная работа №8**

**по дисциплине**

**«Технологии разработки мобильных приложений»**

**Работа с камерой на операционной системе Android**

**Выполнил**:

ст. гр. ПРИ-120

Д. А. Грачев

**Принял**:

Петрова А. И.

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.

ВЫПОЛНЕНИЕ РАБОТЫ

1. По примеру из методички было создано приложение, которое отображает на экране картинку с камеры

public class MainActivity extends AppCompatActivity {  
  
 final int CAMERA\_ID = 0;  
 final boolean FULL\_SCREEN = true;  
 SurfaceView sv;  
 SurfaceHolder holder;  
 HolderCallback holderCallback;  
 Camera camera;   
  
 @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 = findViewById(R.id.*surfaceView*);   
 holder = sv.getHolder();  
 holder.setType(SurfaceHolder.*SURFACE\_TYPE\_PUSH\_BUFFERS*);  
  
 holderCallback = new HolderCallback();  
 holder.addCallback(holderCallback);  
 }  
  
 @Override  
 public 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;  
 }  
  
 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 {  
 result = (360 - degrees) - info.orientation;  
 result += 300;  
 }  
 result = result % 360;  
 camera.setDisplayOrientation(result);  
  
 }  
  
 public class HolderCallback implements SurfaceHolder.Callback {  
  
 public void surfaceCreated(SurfaceHolder holder) {  
 try {  
 camera.setPreviewDisplay(holder);  
 camera.startPreview();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 @Override  
 public void surfaceChanged(@NonNull SurfaceHolder holder, int format, int width, int height) {  
 camera.stopPreview();  
 setCameraDisplayOrientation(CAMERA\_ID);  
 try {  
 camera.setPreviewDisplay(holder);  
 camera.startPreview();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
}

1. На экран была добавлена кнопка сохранения фотографии

public void takePhoto(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();  
 }  
 };  
 });  
};

button = findViewById(R.id.*button*);  
button.setOnClickListener(v-> takePhoto(v));

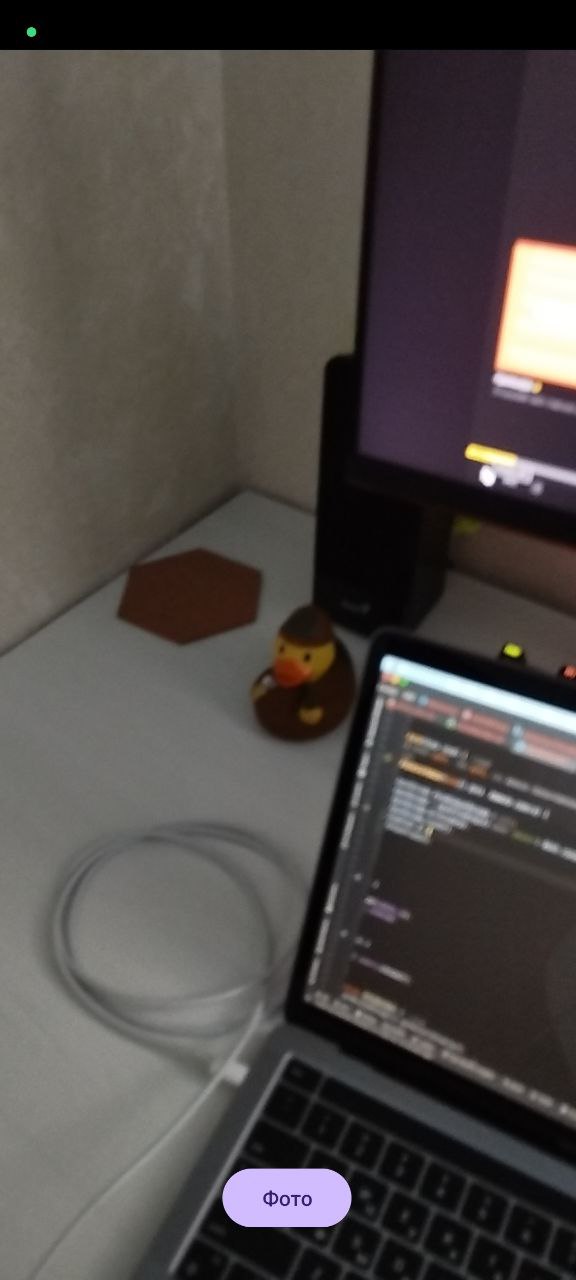


Рисунок 1. Экран приложения

1. С помощью кнопки «Фото» можно сделать фотографию, результат представлен на рисунке 2



Рисунок 2. Сделанная фотография

ВЫВОД

В ходе выполнения лабораторной работы было разработано мобильное приложение, которое будет использовать камеру устройства для вывода изображения на экран.