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 № 4-5**

**по дисциплине**

**«Технологии разработки мобильных приложений»**

Выполнил:

ст. гр. ПРИ-120

К. А. Борисова

Принял:

А. И. Рощина

Владимир, 2023

**Работа с сенсорами на операционной системе Android**

**Цель работы**

Изучить работу сенсоров на примере платформы Android и разработать мобильное приложение, которое будет считывать сигналы сенсоров устройства и представлять их в удобном для восприятия виде.

**Ход работы**

Для начала создадим шаблонный проект Android приложения. Выберем шаблон с пустой активностью. Создадим приложение, используя доступные сенсоры. Доработаем приложение настроив изменения яркости экрана телефона, используя датчик освещенности. Часть листинга MainActivity.java (полный код <https://github.com/VV-I-S/trmp_labs/tree/main/Lab4Level> ):

public void onSensorChanged(SensorEvent event)  
{  
 if(event.sensor.getType()==Sensor.*TYPE\_ACCELEROMETER*){  
 mAccelerometerX.setText(Float.*toString*(event.values[0]));  
 mAccelerometerY.setText(Float.*toString*(event.values[1]));  
 mAccelerometerZ.setText(Float.*toString*(event.values[2]));  
 }  
 if(event.sensor.getType()==Sensor.*TYPE\_MAGNETIC\_FIELD*){  
 mMagneticX.setText(Float.*toString*(event.values[0]));  
 mMagneticY.setText(Float.*toString*(event.values[1]));  
 mMagneticZ.setText(Float.*toString*(event.values[2]));  
 }  
 if(event.sensor.getType()==Sensor.*TYPE\_PROXIMITY*){  
 mProximity.setText(Float.*toString*(event.values[0]));  
 }  
 if(event.sensor.getType()==Sensor.*TYPE\_LIGHT*){  
 mLight.setText(Float.*toString*(event.values[0]));  
 mValue = event.values[0];  
 WindowManager.LayoutParams layout = getWindow().getAttributes();  
 layout.screenBrightness = (int)(255f\* mValue / mMaxValue);  
 getWindow().setAttributes(layout);  
 }  
}  
@Override  
protected void onStart() {  
 super.onStart();  
 sensorManager.registerListener(this, mAccelerometerSensor, SensorManager.*SENSOR\_DELAY\_NORMAL*);  
 sensorManager.registerListener(this, mMagneticSensor, SensorManager.*SENSOR\_DELAY\_NORMAL*);  
 sensorManager.registerListener(this, mProximitySensor, SensorManager.*SENSOR\_DELAY\_NORMAL*);  
 sensorManager.registerListener(this, mLightSensor, SensorManager.*SENSOR\_DELAY\_UI*);  
}  
@Override  
protected void onStop() {  
 super.onStop();  
 sensorManager.unregisterListener(this, mAccelerometerSensor);  
 sensorManager.unregisterListener(this, mMagneticSensor);  
 sensorManager.unregisterListener(this, mProximitySensor);  
 sensorManager.unregisterListener(this, mLightSensor);  
}

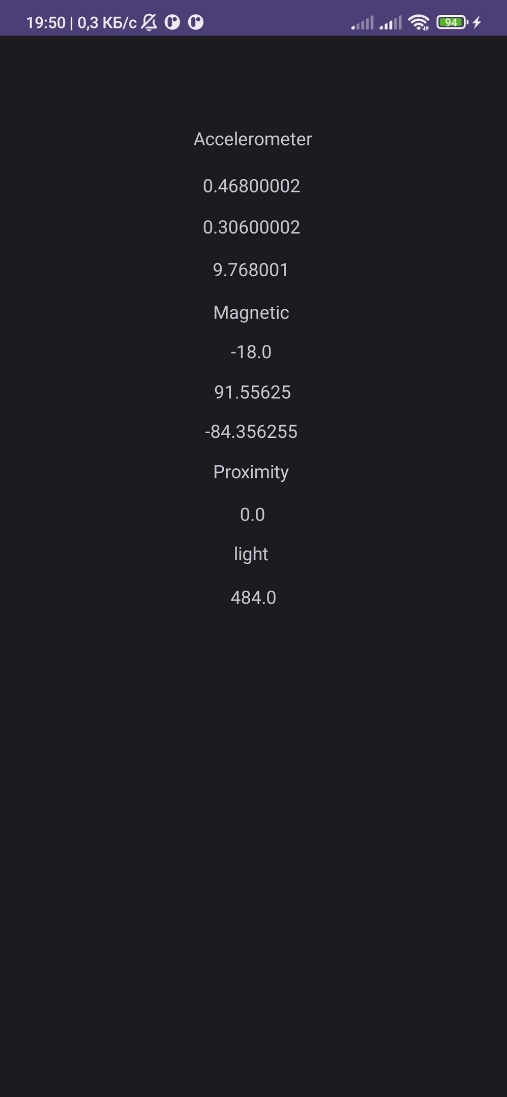
Результат работы:

Рисунок . Результат работы используя разные сенсоры

Используя тип сенсора Sensor.*TYPE\_ROTATION\_VECTOR,* создадим уровень. Часть кода (полный код по ссылке https://github.com/VV-I-S/trmp\_labs/tree/main/Lab4 ):

@Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 tv = (TextView) findViewById(R.id.*tv*);  
 im = (ImageView) findViewById(R.id.*im*);  
 sm = (SensorManager) getSystemService(Context.*SENSOR\_SERVICE*);  
 mLightSensor = sm.getDefaultSensor(Sensor.*TYPE\_LIGHT*);  
 mMaxValue = mLightSensor.getMaximumRange();  
 if(sm != null)s = sm.getDefaultSensor(Sensor.*TYPE\_ROTATION\_VECTOR*);  
 sv = new SensorEventListener() {  
 @Override  
 public void onSensorChanged(SensorEvent sensorEvent) {  
  
 float[] rotationMatrix = new float[16];  
 SensorManager.*getRotationMatrixFromVector*(rotationMatrix, sensorEvent.values);  
 float[] remmapedRotationMatrix = new float[16];  
 SensorManager.*remapCoordinateSystem*(rotationMatrix, SensorManager.*AXIS\_X*, SensorManager.*AXIS\_Z*, remmapedRotationMatrix);  
  
 float[] orientations = new float[3];  
 SensorManager.*getOrientation*(remmapedRotationMatrix, orientations);  
 for (int i=0; i<3; i++){  
 orientations[i] = (float) (Math.*toDegrees*(orientations[i]));  
 }  
 tv.setText(String.*valueOf*((int) orientations[2]));  
 im.setRotation(-orientations[2]);  
 }  
 @Override  
 public void onAccuracyChanged(Sensor sensor, int i) {  
 }  
 };  
 }  
 @Override  
 protected void onResume() {  
 super.onResume();  
 sm.registerListener(sv, s, SensorManager.*SENSOR\_DELAY\_FASTEST*);  
 }  
 @Override  
 protected void onPause() {  
 super.onPause();  
 sm.unregisterListener(sv);  
 }  
}

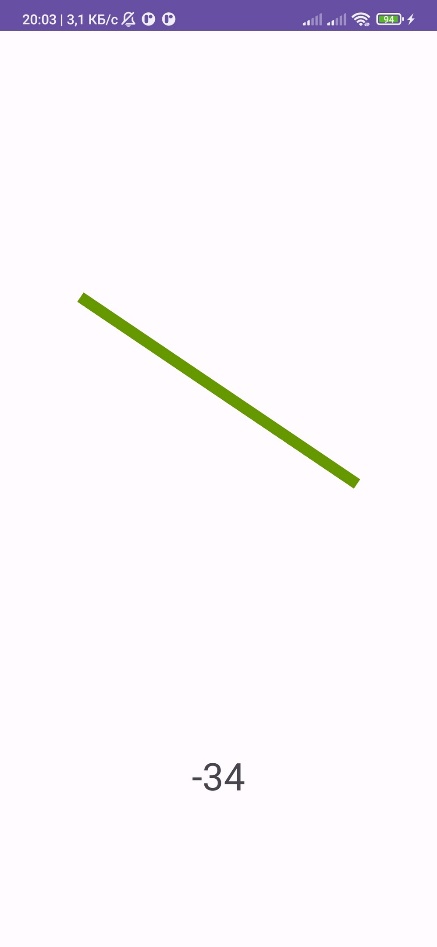
Результат работы в светлой и темной темах:

Рисунок . Результат работы

**Работа с базой данных на операционной системе Android**

**Цель работы**

Изучить работу с базой данных в ОС Android на примере SQLite и получить практические навыки разработки мобильного приложения, взаимодействующего с базой данных.

**Ход работы**

Создадим новый проект. Доработаем код из методических указаний, добавив новый столбец с датой создания строки и вывод значений на экран.

Листинг части кода (полный код по ссылке <https://github.com/VV-I-S/trmp_labs/tree/main/Lab5> ):

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 mBtnAdd = (Button) findViewById(R.id.*buttonAdd*);  
 mBtnAdd.setOnClickListener(this:: onClick);  
 mBtnRead = (Button) findViewById(R.id.*buttonRead*);  
 mBtnRead.setOnClickListener(this :: onClick);  
 mBtnClear = (Button) findViewById(R.id.*buttonClear*);  
 mBtnClear.setOnClickListener(this :: onClick);  
 mEdtName = (EditText) findViewById(R.id.*editTextName*);  
 mEdtEmail = (EditText) findViewById(R.id.*editTextEmail*);  
 mDbHelper = new DBHelper(this);  
 DBInfo = (TextView) findViewById(R.id.*DBInfo*);  
}  
@SuppressLint("NonConstantResourceId")  
public void onClick(View v) {  
 // создаем объект для данных  
 ContentValues cv = new ContentValues();  
 // получаем данные из полей ввода  
 String name = mEdtName.getText().toString();  
 String email = mEdtEmail.getText().toString();  
 String date = new Date().toString();  
 // подключаемся к БД  
 SQLiteDatabase db = mDbHelper.getWritableDatabase();  
 if (v.getId() == R.id.*buttonAdd*){  
 Log.*d*(LOG\_TAG, "--- Insert in mytable: ---");  
 // подготовим данные для вставки в виде пар: наименование столбца -  
 // значение  
 cv.put("date\_of\_create", date);  
 cv.put("name", name);  
 cv.put("email", email);  
 // вставляем запись и получаем ее ID  
 long rowID = db.insert("mytable", null, cv);  
 Log.*d*(LOG\_TAG, "row inserted, ID = " + rowID);  
 }  
 if (v.getId() == R.id.*buttonRead*){  
 Log.*d*(LOG\_TAG, "--- Rows in mytable: ---");  
 // делаем запрос всех данных из таблицы mytable, получаем Cursor  
 Cursor c = db.query("mytable", null, null, null, null, null, null);  
 // ставим позицию курсора на первую строку выборки  
 // если в выборке нет строк, вернется false  
 if (c.moveToFirst()) {  
 // определяем номера столбцов по имени в выборке  
 int idColIndex = c.getColumnIndex("id");  
 int idColDate = c.getColumnIndex("date\_of\_create");  
 int nameColIndex = c.getColumnIndex("name");  
 int emailColIndex = c.getColumnIndex("email");  
 String info ="";  
 do {  
 // получаем значения по номерам столбцов и пишем все в лог  
 info+=  
 "ID = " + c.getInt(idColIndex) + ",date\_of\_create"  
 + c.getString(idColDate) + ", name = "  
 + c.getString(nameColIndex) + ", email = "  
 + c.getString(emailColIndex)+"\n";  
 } while (c.moveToNext());  
 DBInfo.setText(info);  
 } else  
 Log.*d*(LOG\_TAG, "0 rows");  
 c.close();  
 }  
 if (v.getId() == R.id.*buttonClear*){  
 Log.*d*(LOG\_TAG, "--- Clear mytable: ---");  
 // удаляем все записи  
 int clearCount = db.delete("mytable", null, null);  
 Log.*d*(LOG\_TAG, "deleted rows count = " + clearCount);  
 }  
 // закрываем подключение к БД  
 mDbHelper.close();  
}  
class DBHelper extends SQLiteOpenHelper {  
 public DBHelper(Context context) {  
 // конструктор суперкласса  
 super(context, "myDB", null, 1);  
 }  
 public void onCreate(SQLiteDatabase db) {  
 Log.*d*(LOG\_TAG, "--- onCreate database ---");  
 // создаем таблицу с полями  
 db.execSQL("create table mytable ("  
 + "id integer primary key autoincrement,"  
 + "date\_of\_create text,"  
 + "name text,"  
 + "email text" + ");");  
 }

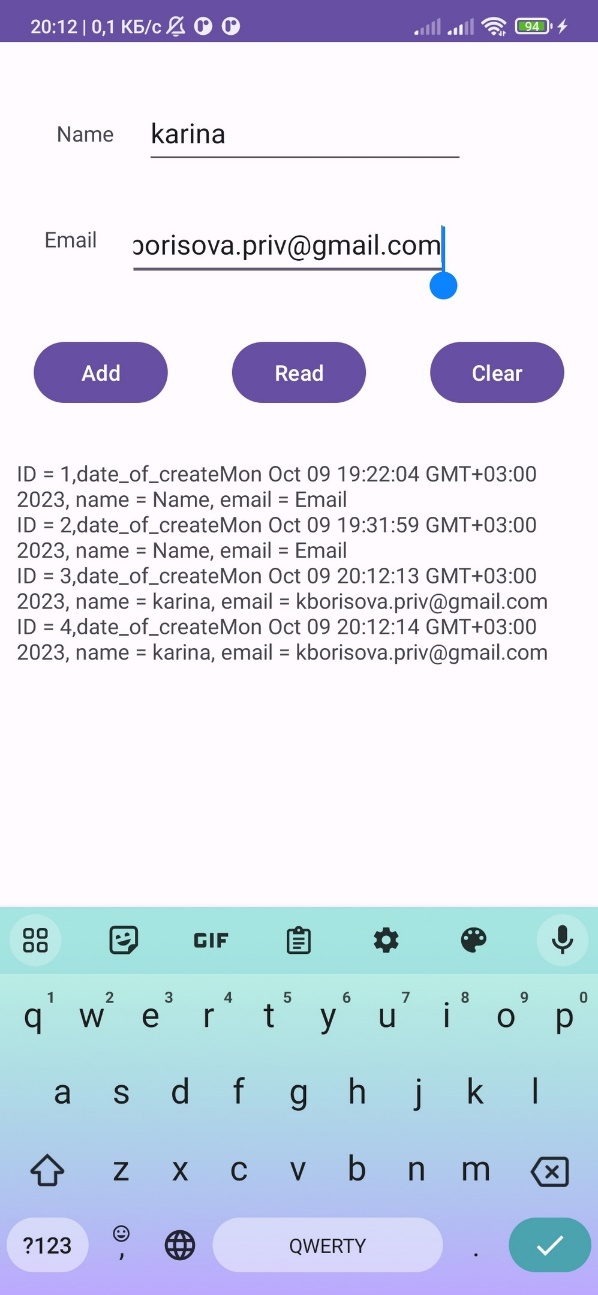
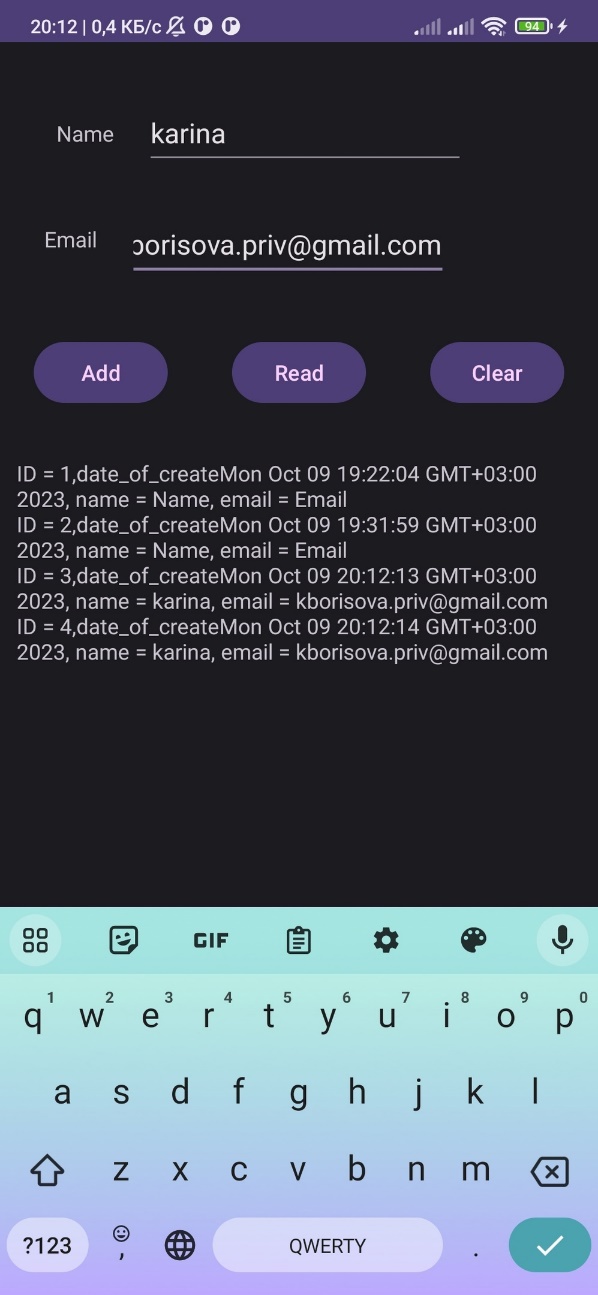
Результат работы в разных системных темах:

Рисунок . Результат работы

**Вывод**

В ходе данной работы была изучена работа сенсоров на примере платформы Android и разработано мобильное приложение, которое считывает сигналы сенсоров устройства и представляет их в удобном для восприятия виде.

Также была изучена работа с базой данных в ОС Android на примере SQLite и получены практические навыки разработки мобильного приложения, взаимодействующего с базой данных.