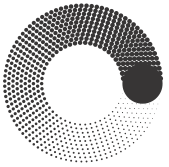
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

***Факультет информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

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

**ЛАБОРАТОРНАЯ РАБОТА № 9**

**Дисциплина: Разработка мобильных приложений для Android**

**Тема: Работа с сенсорами Android-устройства**

**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, позволяющее определить имеющиеся на

устройстве датчики, а также выводящее на экран показания нескольких датчиков.

**ХОД РАБОТЫ**

В начале, был разработан макет главного Activity приложения, показанный на рисунке 1 и состоящий из объекта ScrollView, внутри которого имеется объект LinearLayout, в который будут добавляться объекты TextView для отображения данных с различных сенсоров. Листинг исходного кода макета показан в приложении А.

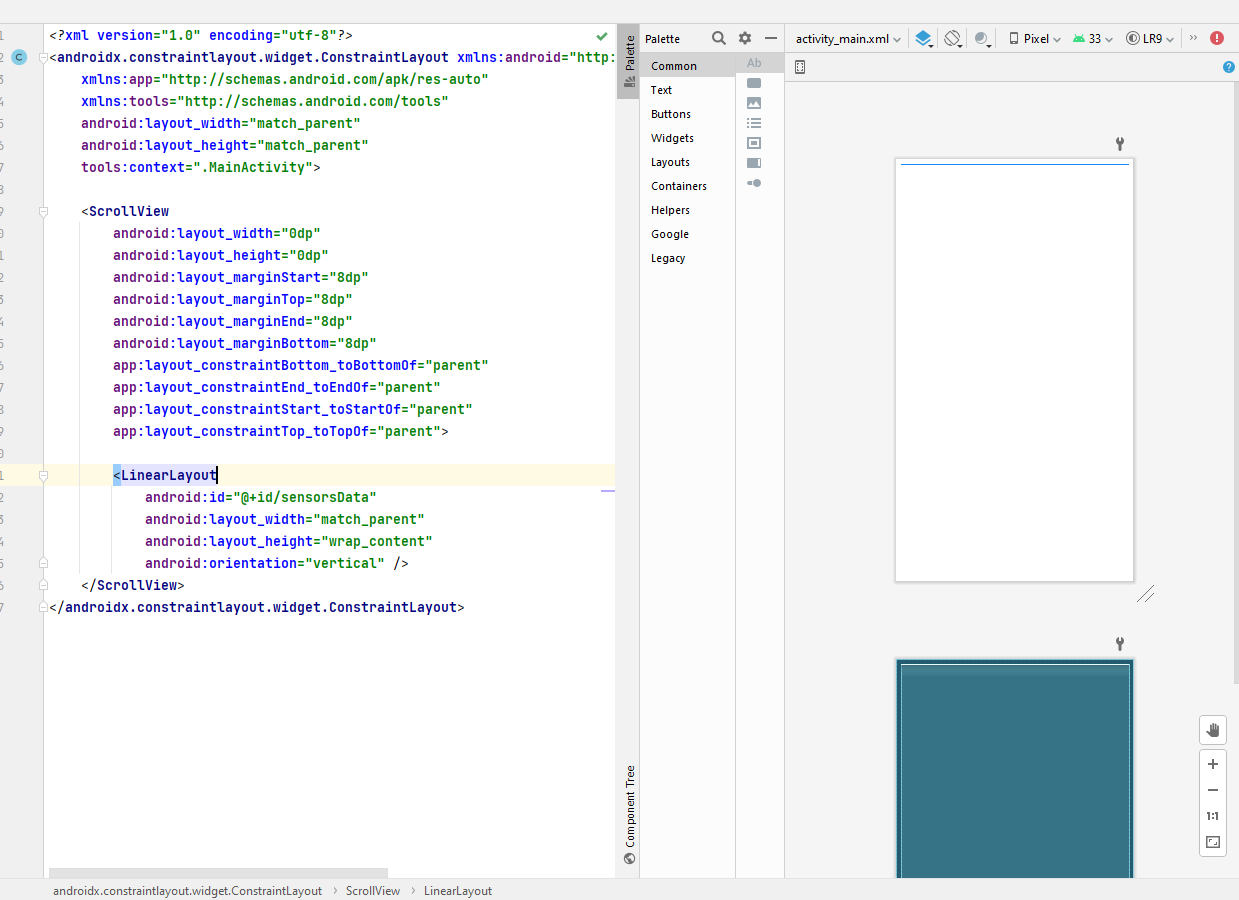


Рисунок 1 – Разработанный макет приложения

Т.к. различные мобильные устройства могут иметь множество разных сенсоров, для более эффективной работы, было принято решение не пытаться инициализировать строго определённые сенсоры, а проводить поиск по существующим в устройстве сенсорам по их ID от 0 до 1000, что покрывает все существующие сенсоры в устройстве. В случае, если данный сенсор имеет название, можно считать, что он имеется в устройстве. В этом случае, сенсор будет добавлен в список, а также создан объект TextView для отображения названия сенсора и данных с него, который также будет добавлен с соответствующий список. Этот процесс поиска сенсоров показан в листинге 1.

Листинг 1 – Поиск сенсоров в устройстве

*// Iterate all possible sensors***for** (**int** i = 0; i < ***SENSOR\_ID\_MAX***; i++) {  
 *// Try to get them* **try** {  
 Sensor sensor = **sensorManager**.getDefaultSensor(i);  
  
 *// Check if sensor has name* **if** (sensor.getName().length() > 1) {  
 *// Try to add it to the List* **sensorsList**.add(sensor);  
  
 *// Create new TextView for sensor data* TextView textView = **new** TextView(**this**);  
  
 *// Set sensor name* textView.setText(String.*format*(**"Sensor name: %s\nNo data yet...\n"**,  
 sensor.getName()));  
  
 *// Add this TextView to the LinearLayout* linearLayout.addView(textView);  
  
 *// Also, add it to the List of text views* **textViewsList**.add(textView);  
 }  
 } **catch** (Exception ignored) { }  
}

Далее, вызывается метод registerListeners(), что показано в листинге 2, подключающий метод для обработки данных с сенсоров.

Листинг 2 – подключение обработчика данных с сенсоров.

*/\*\*  
 \* Registers (connects) listener for all sensors from List  
 \*/***private void** registerListeners() {  
 **if** (**sensorManager** != **null**) {  
 **for** (Sensor sensor: **sensorsList**) {  
 **try** {  
 **sensorManager**.registerListener(**this**, sensor,  
 SensorManager.***SENSOR\_DELAY\_NORMAL***);  
 } **catch** (Exception ignored) { }  
 }  
 }  
}

Когда приходят новые данные с сенсора, метод onSensorChanged вызывает метод getTextViewBySensor, возвращающий объект TextView из списка по заданному объекту Sensor, что показано в листинге 3.

В случае, если TextView найдет успешно, в него происходит запись названия сенсора и данных с него, что позволяет универсально обрабатывать данных с различных типов сенсоров.

Листинг 3 – Обработка данных с сенсоров

*/\*\*  
 \* Retrieves TextView from textViewsList based on Sensor object from sensorsList  
 \** ***@param sensor*** *Sensor object (from sensorsList)  
 \** ***@return*** *TextView (textViewsList)  
 \*/***private** TextView getTextViewBySensor(Sensor sensor) {  
 *// Try to find sensor* **for** (**int** i = 0; i < **sensorsList**.size(); i++) {  
 **if** (**sensorsList**.get(i).equals(sensor)) {  
 **return textViewsList**.get(i);  
 }  
 }  
  
 *// No sensor found* **return null**;  
}  
  
*/\*\*  
 \* Updates sensor's data into TextViews  
 \** ***@param sensorEvent*** *SensorEvent object (from connected listener)  
 \*/*@Override  
**public void** onSensorChanged(SensorEvent sensorEvent) {  
 *// Try to get matching TextView* TextView textView = getTextViewBySensor(sensorEvent.**sensor**);  
  
 *// Check if TextView exists* **if** (textView != **null**)  
 textView.setText(String.*format*(**"Sensor name: %s\nSensor data: %s\n"**,  
 sensorEvent.**sensor**.getName(), Arrays.*toString*(sensorEvent.**values**)));  
}

Пример работы приложения показан на рисунке 2. Как можно заметить, данные с некоторых сенсоров приходят не сразу, т.к. некоторые сенсоры обновляются только при определённых обстоятельствах.

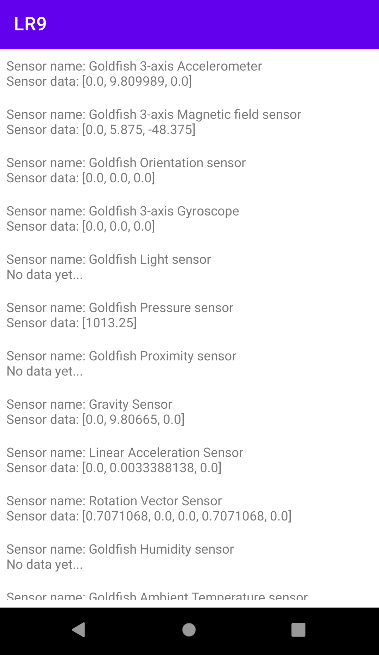


Рисунок 9 – Демонстрация работы приложения

**ВЫВОД**

В результате выполнения данной лабораторной работы были изучены навыки чтения данных с различных сенсоров мобильного устройства и их отображения.

**Приложение А. 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"  
 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"**>  
  
 <**ScrollView  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginBottom="8dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"**>  
  
 <**LinearLayout  
 android:id="@+id/sensorsData"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"** />  
 </**ScrollView**>  
</**androidx.constraintlayout.widget.ConstraintLayout**>

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

**package** com.labs.lr9;  
  
**import** android.content.Context;  
**import** android.hardware.Sensor;  
**import** android.hardware.SensorEvent;  
**import** android.hardware.SensorEventListener;  
**import** android.hardware.SensorManager;  
**import** android.os.Bundle;  
**import** android.widget.LinearLayout;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
  
**import** java.util.ArrayList;  
**import** java.util.Arrays;  
**import** java.util.List;  
  
**public class** MainActivity **extends** AppCompatActivity **implements** SensorEventListener {  
 *// System service for managing sensors* **private** SensorManager **sensorManager**;  
  
 *// Search all sensors with ID up to this value* **private static final int *SENSOR\_ID\_MAX*** = 1000;  
  
 *// Lists for sensors and TextViews* **private final** List<Sensor> **sensorsList** = **new** ArrayList<>();  
 **private final** List<TextView> **textViewsList** = **new** ArrayList<>();  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// Get system service* **sensorManager** = (SensorManager) getSystemService(Context.***SENSOR\_SERVICE***);  
  
 *// Check if we have this service* **if** (**sensorManager** != **null**) {  
 *// Get linear layout from activity* LinearLayout linearLayout = findViewById(R.id.***sensorsData***);  
  
 *// Iterate all possible sensors* **for** (**int** i = 0; i < ***SENSOR\_ID\_MAX***; i++) {  
 *// Try to get them* **try** {  
 Sensor sensor = **sensorManager**.getDefaultSensor(i);  
  
 *// Check if sensor has name* **if** (sensor.getName().length() > 1) {  
 *// Try to add it to the List* **sensorsList**.add(sensor);  
  
 *// Create new TextView for sensor data* TextView textView = **new** TextView(**this**);  
  
 *// Set sensor name* textView.setText(String.*format*(**"Sensor name: %s\nNo data yet...\n"**,  
 sensor.getName()));  
  
 *// Add this TextView to the LinearLayout* linearLayout.addView(textView);  
  
 *// Also, add it to the List of text views* **textViewsList**.add(textView);  
 }  
 } **catch** (Exception ignored) { }  
 }  
  
 *// Connect listeners* registerListeners();  
 }  
  
 *// We don't have service for sensors =(* **else** {  
 Toast.*makeText*(**this**, **"Sensor service not available"**,  
 Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 */\*\*  
 \* Registers (connects) listener for all sensors from List  
 \*/* **private void** registerListeners() {  
 **if** (**sensorManager** != **null**) {  
 **for** (Sensor sensor: **sensorsList**) {  
 **try** {  
 **sensorManager**.registerListener(**this**, sensor,  
 SensorManager.***SENSOR\_DELAY\_NORMAL***);  
 } **catch** (Exception ignored) { }  
 }  
 }  
 }  
  
 */\*\*  
 \* Retrieves TextView from textViewsList based on Sensor object from sensorsList  
 \** ***@param sensor*** *Sensor object (from sensorsList)  
 \** ***@return*** *TextView (textViewsList)  
 \*/* **private** TextView getTextViewBySensorID(Sensor sensor) {  
 *// Try to find sensor* **for** (**int** i = 0; i < **sensorsList**.size(); i++) {  
 **if** (**sensorsList**.get(i).equals(sensor)) {  
 **return textViewsList**.get(i);  
 }  
 }  
  
 *// No sensor found* **return null**;  
 }  
  
 */\*\*  
 \* Updates sensor's data into TextViews  
 \** ***@param sensorEvent*** *SensorEvent object (from connected listener)  
 \*/* @Override  
 **public void** onSensorChanged(SensorEvent sensorEvent) {  
 *// Try to get matching TextView* TextView textView = getTextViewBySensorID(sensorEvent.**sensor**);  
  
 *// Check if TextView exists* **if** (textView != **null**)  
 textView.setText(String.*format*(**"Sensor name: %s\nSensor data: %s\n"**,  
 sensorEvent.**sensor**.getName(), Arrays.*toString*(sensorEvent.**values**)));  
 }  
  
 @Override  
 **public void** onAccuracyChanged(Sensor sensor, **int** accuracy) {  
  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
  
 *// Register listeners back after activity resuming* registerListeners();  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
  
 *// Unregister listeners on activity pause* **sensorManager**.unregisterListener(**this**);  
 }  
}