Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

Факультет компьютерных технологий

Кафедра ПУРИС

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА

по дисциплине «Программирование мобильных устройств»

Студент группы 0ВТб-1 Н.Д. Малышев

Преподаватель В.А. Тихомиров

2023

Задание

В приложении три активности. Активности перелистываются влево, когда аппарат соответственно наклоняют влево, и перелистываются назад, когда аппарат наклоняют вправо.

Содержание

1 Справка к коду

В начале кода определены несколько переменных:

- mSensorManager (менеджер датчиков);
- mAccelerometer (объект датчика);
- mViewFlipper (ViewFlipper для смены экранов);
- mLastX (последнее значение координаты x).

Затем в методе onCreate() осуществляется инициализация приложения:

- установка флагов для окна (без заголовка и на весь экран);
- создание ViewFlipper и добавление трех макетов (activity_main.xml, activity_right.xml, activity_left.xml);
- установка ViewFlipper как текущего layout'a для приложения;
- блокировка поворота экрана;
- регистрация обработчика датчиков ускорения (Accelerometer).

Метод lockScreenOrientation() используется для блокировки поворота экрана в портретную или альбомную ориентацию.

Методы onResume() и onPause() регистрируют и отменяют регистрацию обработчика датчика при входе/выходе из активности.

Методы onAccuracyChanged() и onSensorChanged() определяют поведение приложения при изменении точности датчика и при изменении показаний датчика.

Метод onSensorChanged() вызывается каждый раз при изменении показаний датчика ускорения. В методе происходит следующее:

- получение координаты x с помощью метода event.values[0];
- проверка условия, если координата x равна 0 и предыдущее значение координаты было меньше 0, то вызываем методы mViewFlipper.setInAnimation() и mViewFlipper.setOutAnimation() для установки анимации перехода между экранами и метод mViewFlipper.setDisplayedChild() для отображения нужного экрана;

- если координата x равна 0 и предыдущее значение координаты было больше 0, то вызываем те же методы, но с другими анимациями и отображаем тот же экран;
- если координата x больше 0 и предыдущее значение координаты было меньше или равно 0, то вызываем те же методы, но с другими анимациями и отображаем третий экран;
- если координата x меньше 0 и предыдущее значение координаты было больше или равно 0, то вызываем те же методы, но с другими анимациями и отображаем второй экран;
- обновление значения mLastX для хранения предыдущей координаты;
- вывод значения mLastX в лог с помощью метода Log.d().

2 Разработка приложения

В активности MainActivity (Листинг 2.1) написан код, реализующий поставленную задачу.

В листинге 2.2 – 2.4 описаны макеты, которые используются в программе.

Также создаются 4 анимации для плавного переключения между активностями, все 4 анимации описаны в листинге 2.5 – 2.8.

Листинг 2.1 – MainActivity

```
package ru.startandroid.develop.myapplication;

import android.content.Context;
import android.content.pm.ActivityInfo;
import android.content.res.Configuration;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.util.Log;
import android.view.Window;
import android.view.WindowManager;
import android.widget.ViewFlipper;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity implements
SensorEventListener {

    private SensorManager mSensorManager;// менеджер датчиков
    private Sensor mAccelerometer;// объект датчика
    private ViewFlipper mViewFlipper;// ViewFlipper для смены
экранов
    private float mLastX;// последнее значение координаты x

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);

getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);

        // Создание ViewFlipper и добавление трех макетов
        mViewFlipper = new ViewFlipper(this);

mViewFlipper.addView(getLayoutInflater().inflate(R.layout.activi
```

```
ty_main, null));

mViewFlipper.addView(getLayoutInflater().inflate(R.layout.activi
ty_right, null));

mViewFlipper.addView(getLayoutInflater().inflate(R.layout.activi
ty_left, null));

        setContentView(mViewFlipper);

        //Выключить поворот экрана
        lockScreenOrientation();

        //Подключить сенсоры
        mSensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        mAccelerometer =
mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        mSensorManager.registerListener(this, mAccelerometer,
SensorManager.SENSOR_DELAY_NORMAL);
    }

    //Процедура блокирования поворота экрана
    private void lockScreenOrientation() {
        int currentOrientation =
getResources().getConfiguration().orientation;
        if (currentOrientation ==
Configuration.ORIENTATION_PORTRAIT) {

setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT
);
        } else {

setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAP
E);
        }
    }

    //Регистрация обработчика датчиков
    protected void onResume() {
        super.onResume();
        mSensorManager.registerListener(this, mAccelerometer,
SensorManager.SENSOR_DELAY_NORMAL);
    }
    //Снятие обработчика
    protected void onPause() {
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
        // Метод вызывается, если точность датчика изменяется
```

```
        // Игнорируем в данном случае
        }

    @Override
    public void onSensorChanged(SensorEvent event) {
        // Метод вызывается при изменении показаний датчика
        float x = event.values[0];// Получение координаты x
        //mLastX = x;
        if (x == 0 && mLastX < 0) {
            mViewFlipper.setInAnimation(this,
R.anim.slide_in_left);
            mViewFlipper.setOutAnimation(this,
R.anim.slide_out_right);
            mViewFlipper.setDisplayedChild(0);

        }
        else if (x == 0 && mLastX > 0) {
            mViewFlipper.setInAnimation(this,
R.anim.slide_in_right);
            mViewFlipper.setOutAnimation(this,
R.anim.slide_out_left);
            mViewFlipper.setDisplayedChild(0);

        }

        else if (x > 0 && mLastX <= 0) {
            mViewFlipper.setInAnimation(this,
R.anim.slide_in_left);
            mViewFlipper.setOutAnimation(this,
R.anim.slide_out_right);
            mViewFlipper.setDisplayedChild(2);

        }
        else if (x < 0 && mLastX >= 0) {
            mViewFlipper.setInAnimation(this,
R.anim.slide_in_right);
            mViewFlipper.setOutAnimation(this,
R.anim.slide_out_left);
            mViewFlipper.setDisplayedChild(1);
        }
        mLastX = x;
        Log.d("T_001", "X=" + Float.toString(mLastX));
    }
}
```

Листинг 2.2 – activity_main.xml

```
<?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"
    android:background="@drawable/c_back"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="CENTER"
        android:textSize="50sp"
        android:textStyle="bold"
        android:textColor="#000000"
        android:gravity="center"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Листинг 2.3 – activity_left.xml

```
<?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"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/l_back">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="LEFT"
        android:textSize="50sp"
        android:textStyle="bold"
        android:textColor="#000000"
        android:gravity="center"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Листинг 2.4 – activity_right.xml

```
<?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"
    android:background="@drawable/r_back">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="RIGHT"
        android:textSize="50sp"
        android:textStyle="bold"
        android:textColor="#000000"
        android:gravity="center"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Листинг 2.5 – slide_in_left.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
>
    <translate
        android:duration="600"
        android:fromXDelta="100%"
        android:toXDelta="0%" >
    </translate>
</set>
```

Листинг 2.6 – slide_in_right.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
>
    <translate
        android:duration="600"
        android:fromXDelta="-100%"
        android:toXDelta="0%" >
    </translate>
</set>
```

Листинг 2.7 – slide_out_left.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
>
    <translate
        android:duration="600"
        android:fromXDelta="0%"
        android:toXDelta="-100%" >
```

```
    </translate>
</set>
```

Листинг 2.8 – slide_out_right.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
>
    <translate
        android:duration="600"
        android:fromXDelta="0%"
        android:toXDelta="100%" >
    </translate>
</set>
```

3 Тестирование

Работа приложения представлена на рисунке 3.1 – 3.3.

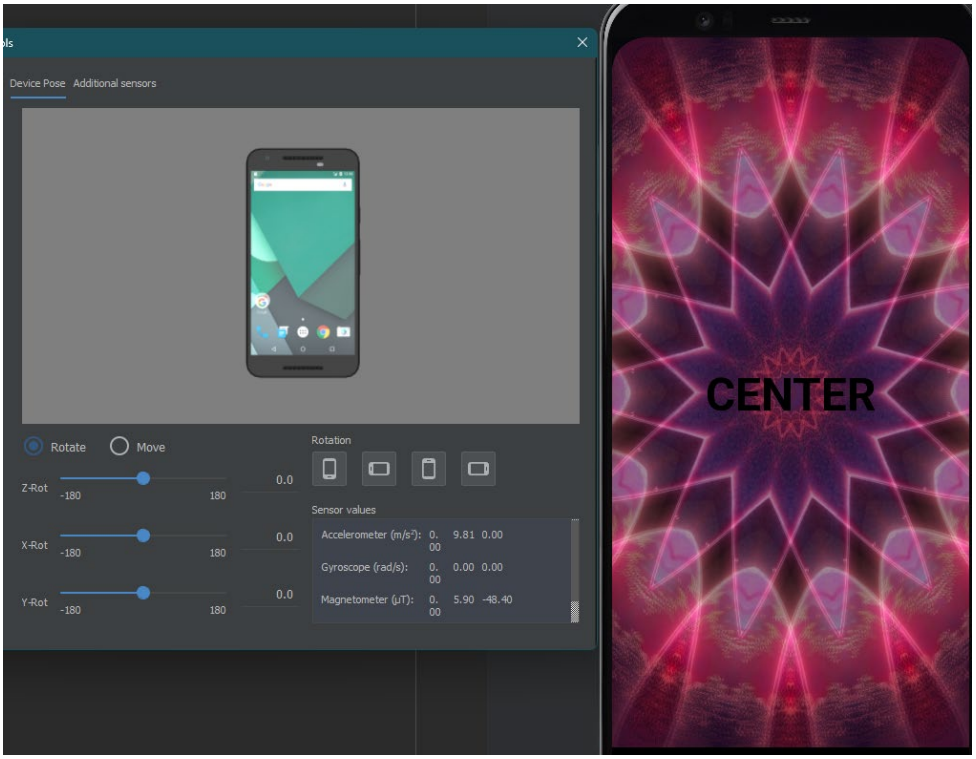

Рисунок 3.1 – Пример работы приложения

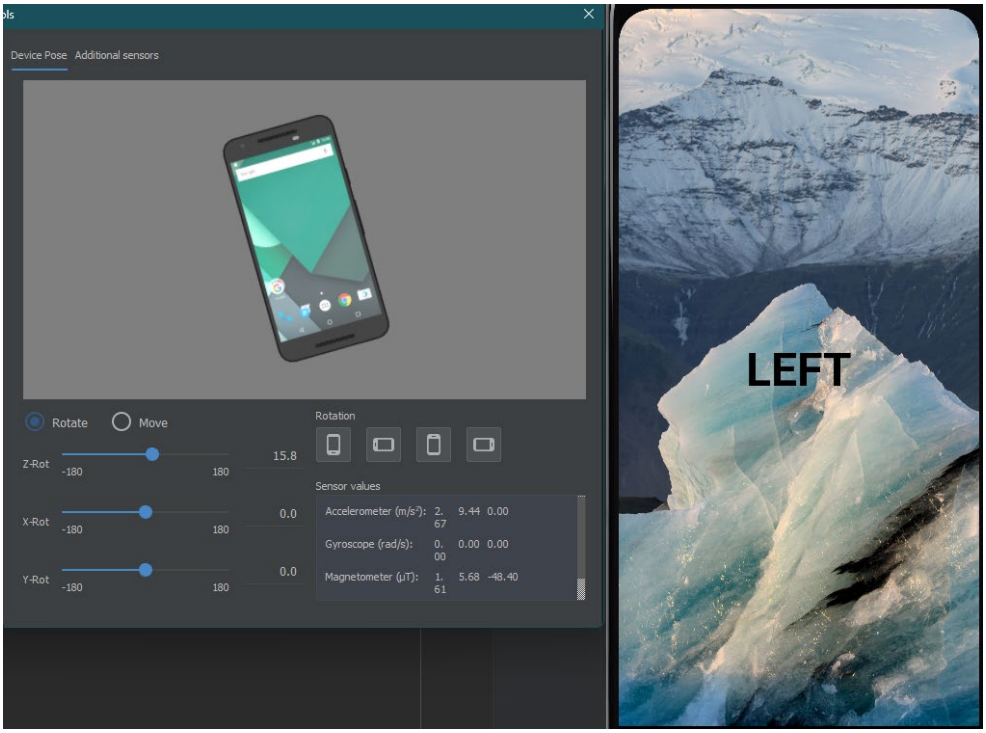

Рисунок 3.2 – Пример работы приложения

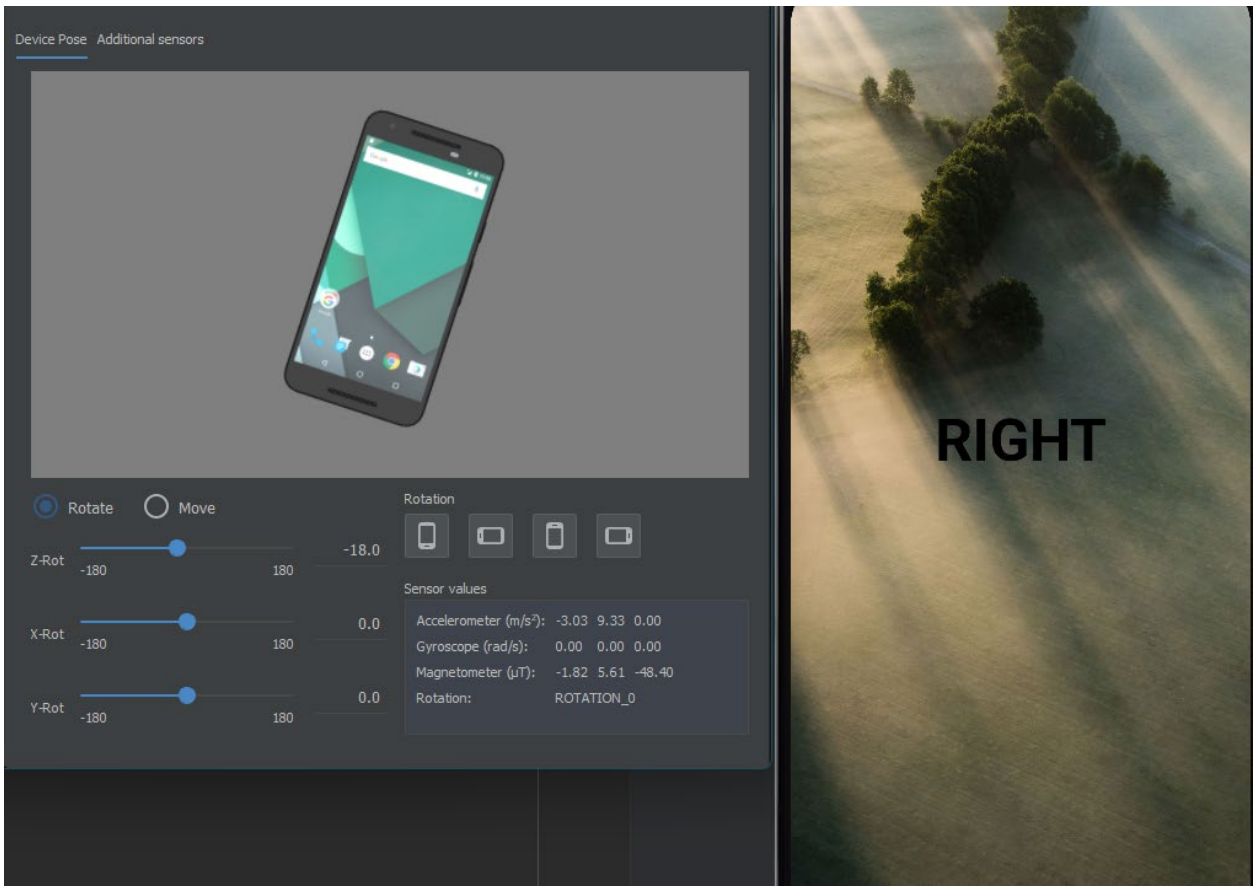

Рисунок 3.3 – Пример работы приложения

Список использованных источников

1 РД ФГБОУ ВО «КнАГТУ» 013-2016. Текстовые студенческие работы. Правила оформления. – Введ. 2016-03-10. – Комсомольск-на-Амуре: ФГБОУ ВО «КнАГТУ», 2016. – 55 с.