Ćwiczenia 18 – Android studio – Sensors

Na koniec zajęć prześlij pliki źródłowe (.xml, .java)+ obrazek do zasobu w teams.

1. Utwórz projekt o nazwie WorkWithSensors na podstawie Empty Activity, dobierz odpowiednie API (28 – Android 9).
2. Otworzyć dokumentację:

https://developer.android.com/guide/topics/sensors
https://developer.android.com/guide/topics/sensors/sensors_overview
https://developer.android.com/reference/android/hardware/Sensor
https://developer.android.com/reference/android/hardware/SensorManager
https://developer.android.com/reference/android/hardware/SensorEvent
https://developer.android.com/reference/android/hardware/SensorEventListener

3. Zidentyfikuj czujniki znajdujące się na urządzeniu.

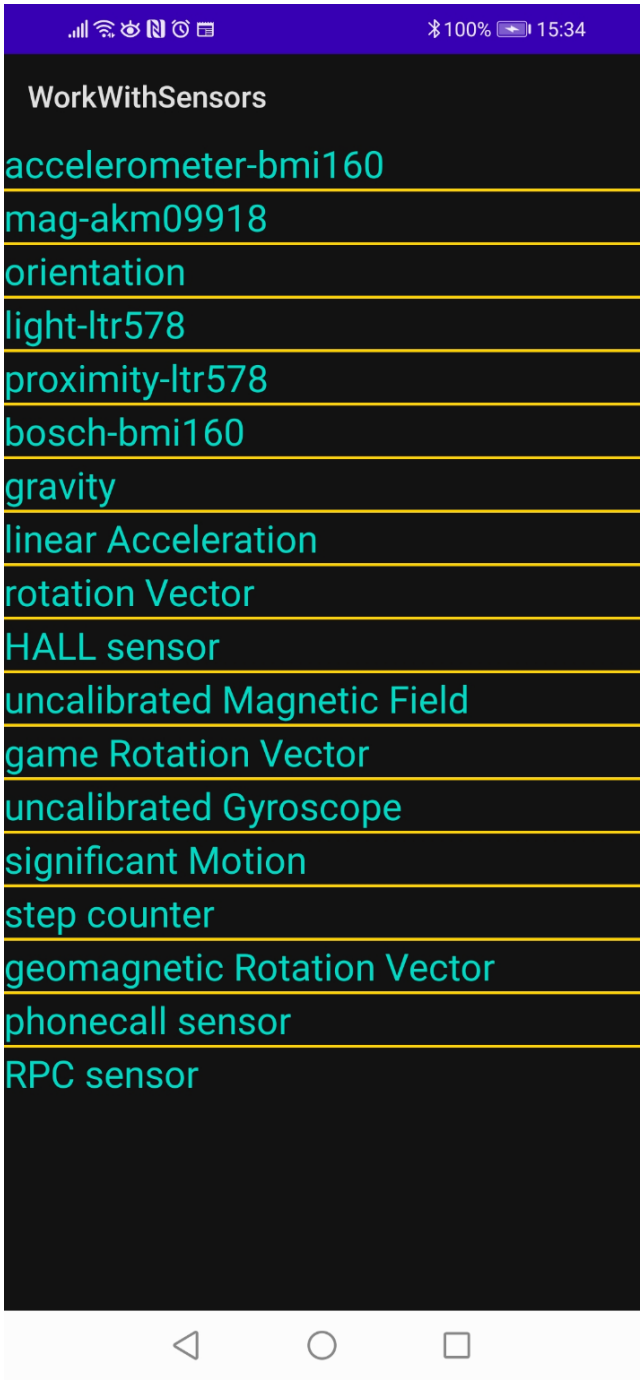
```
10  public class MainActivity extends AppCompatActivity {
11      private SensorManager sensorManager;
12      @Override
13      protected void onCreate(Bundle savedInstanceState) {
14          super.onCreate(savedInstanceState);
15          setContentView(R.layout.activity_main);
16          sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
```

4. Uzyskaj listę wszystkich czujników na urządzeniu.

```
16          sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
17          List<Sensor> deviceSensors = sensorManager.getSensorList(Sensor.TYPE_ALL);
18      }
```

5. Wyświetl nazwy czujników w listView, skorzystaj z ArrayAdapter<String> oraz w LogCat:

```
'com.example.workwithsensors V/mysensors: --------------->step counter
'com.example.workwithsensors V/mysensors: --------------->geomagnetic Rotation Vector
'com.example.workwithsensors V/mysensors: --------------->phonecall sensor
'com.example.workwithsensors V/mysensors: --------------->RPC sensor
```



6. Sprawdź wybrany czujnik domyślny i wyświetl szczegółowe dane, np.:

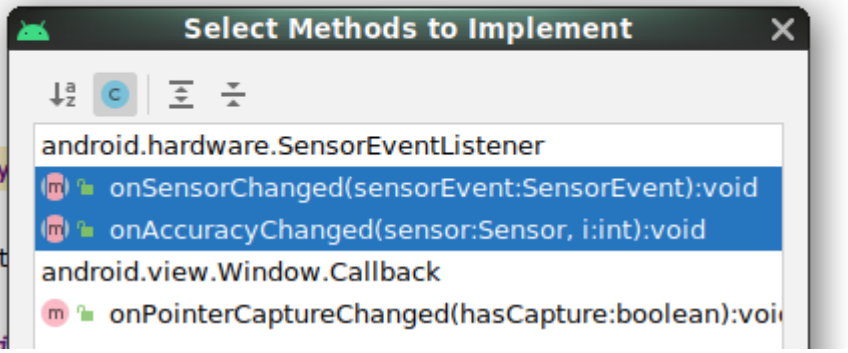
```
private void checkSensorMagnetic() {
    sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
    if (sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD) != null){
        Log.v(TAG, msg: "Success! There's a magnetometer."+
                sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD) );
    } else {
        Log.v(TAG,  msg: "Failure! No magnetometer.");
    }
}
```

7. Znajdź i wypisz wszystkie czujniki grawitacji.

```
if (sensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY) != null){
    List<Sensor> gravSensors = sensorManager.getSensorList(Sensor.TYPE_GRAVITY);
    for(int i=0; i<gravSensors.size(); i++) {
        Log.v(TAG, msg: "Vendor:" + gravSensors.get(i).getVendor()+
                " name: " + gravSensors.get(i).getName()+"\n");
```

8. Zaimplementuj klasę SensorEventListener oraz dodaj metody:

```
atActivity implements SensorEventListener{
sensors";
```



9. Rejestruj i wyrejestruj detektor zdarzeń czujnika w metodach onResume() i onPause().

```
@Override
protected void onResume() {
    super.onResume();
    sensorManager.registerListener( listener: this, mLight, SensorManager.SENSOR_DELAY_NORMAL);
}
@Override
protected void onPause() {
    super.onPause();
    // unregister a listener
    sensorManager.unregisterListener( this);
}
```

10. Wykorzystaj powyższe cztery metody do monitorowania danych z czujnika światła.

```
private Sensor mLight;
mLight = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
```

```
else if (sensorEvent.sensor.getType() == Sensor.TYPE_LIGHT) {
    System.arraycopy(sensorEvent.values,
            i: 0, lightReading,
            i1: 0, lightReading.length);
}
```

```
String s="";
for(int i=0;i<sensorEvent.values.length; i++) {
    s+=sensorEvent.values[i]+" ";
}
datalight.setText("reading: " + s);
Log.v(TAG,  msg: "reading: " + s + "\n");
```

11. Przetestuj aplikację, uruchom na urządzeniu.

12. Szablon dla pozostałych czujników, zastąp nazwę szablon nazwą czujnika

```
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.util.Log;
import android.view.WindowManager;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class Szablon extends AppCompatActivity implements SensorEventListener {

  TextView text_name;
  TextView text_readings;
  private SensorManager sensorManager;
  private Sensor mSensor;
  boolean isSensorPresent;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_szablon);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
    text_name = findViewById(R.id.text_name);
    text_readings = findViewById(R.id.text_readings);

    sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);

    if (sensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY) != null ){
      mSensor = sensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY);
      isSensorPresent = true;
      Log.v(Config.TAG,"detect ... sensor");
    } else {
      text_name.setText("Sensor is not present");
      isSensorPresent = false;
    }
  }
```

```
    @Override
    public void onSensorChanged(SensorEvent sensorEvent) {
        text_readings.setText((int) sensorEvent.values[0]);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int i) {

    }
    @Override
    protected void onResume() {
        super.onResume();
        Log.v(Config.TAG,"----------> onResume()");

        if (mSensor != null) {
            sensorManager.registerListener(this, mSensor,
                    SensorManager.SENSOR_DELAY_NORMAL);
            Log.v(Config.TAG,"----------> OK sensor registerListener");
        }


    }
    @Override
    protected void onPause() {
        super.onPause();
        // unregister a listener. Don't receive any more updates from either sensor
        if (mSensor != null) {
            sensorManager.unregisterListener(this, mSensor);
        }
        Log.v(Config.TAG,"----------> onPause()");
    }
}
```

13. Layout do szablonu:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    tools:context=".Gravity">

    <TextView
        android:id="@+id/text_name"
        android:text="Name of sensor"
        android:textSize="35sp"
        android:textAlignment="center"
        android:layout_width="match_parent"
        android:layout_height="75dp">
    </TextView>
    <TextView
        android:id="@+id/text_readings"
        android:text="0"
        android:textSize="35sp"
        android:textAlignment="center"
        android:layout_width="match_parent"
        android:layout_height="75dp">
    </TextView>
</LinearLayout>
```

14. W AndroidManifest.xml sprawdź:

```
<uses-feature android:name="android.hardware.sensor.accelerometer"
              android:required="true" />
```

If you add this element and descriptor to your application's manifest, users will see your application on Google Play only if their device has an accelerometer.

You should set the descriptor to `android:required="true"` only if your application relies entirely on a specific sensor. If your application uses a sensor for some functionality, but still runs without the sensor, you should list the sensor in the `<uses-feature>` element, but set the descriptor to `android:required="false"`. This helps ensure that devices can install your app even if they do

15. Część druga – inne czujniki. Dostępne na urządzeniu.
https://developer.android.com/guide/topics/sensors/sensors_motion

16. Przetestuj czujnik położenia, orientacji (orientation) (poziomica). Do określenia orientacji urządzenia można wykorzystać odczyty z akcelerometru urządzenia i czujnika pola geomagnetycznego.
https://developer.android.com/guide/topics/sensors/sensors_position
https://developer.android.com/guide/topics/sensors/sensors_position#sensors-pos-orient

17. Przetestuj czujnik przyspieszenia (accelerometr)
https://developer.android.com/guide/topics/sensors/sensors_motion#sensors-motion-accel

```
private SensorManager sensorManager;
private Sensor sensor;
  ...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
```

18. Przetestuj czujnik grawitacji (gravity)
https://developer.android.com/guide/topics/sensors/sensors_motion#sensors-motion-grav

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY);
```

19. Przetestuj żyroskop (gyroscope)

https://developer.android.com/guide/topics/sensors/sensors_motion#sensors-motion-gyro

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
```

20. Przetestuj czujnik pola geomagnetycznego (geomagnetic)
https://developer.android.com/guide/topics/sensors/sensors_position#sensors-pos-mag

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
```

21. Przetestuj czujnik przyspieszenia liniowego (linear accelerometr)
https://developer.android.com/guide/topics/sensors/sensors_motion#sensors-motion-linear

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);
```

22. Dodatkowe zadania
a) przetestuj działanie czujnika zbliżeniowego

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY);
```

b) przetestuj działanie czujnika wykrywania kroków

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_STEP_DETECTOR);
```

c) sygnificant motion sensor

https://developer.android.com/reference/android/hardware/TriggerEventListener

```
private SensorManager sensorManager;
private Sensor sensor;
private TriggerEventListener triggerEventListener;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_SIGNIFICANT_MOTION);

triggerEventListener = new TriggerEventListener() {
    @Override
    public void onTrigger(TriggerEvent event) {
        // Do work
    }
};

sensorManager.requestTriggerSensor(triggerEventListener, mSensor);
```

23. KONIEC.