KOD:

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.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="com.tomek.akcelerometr.MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center_horizontal"
        android:orientation="vertical"
        tools:layout_editor_absoluteX="0dp"
        tools:layout_editor_absoluteY="0dp">

        <TextView
            android:id="@+id/tvStart"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="5dp"
            android:layout_marginTop="5dp"
            android:gravity="center"
            android:text="@string/tv1"
            android:textAlignment="center"
            android:textColor="@color/colorPrimary"
            android:textSize="22dp" /> <!-- srodkowanie w starszych
wersjach-->

        <TableLayout
            android:layout_width="220dp"
            android:layout_height="wrap_content">

            <TableRow
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:gravity="center_horizontal">

                <TextView
                    android:id="@+id/textViewt"
                    android:layout_width="150dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:gravity="center"
                    android:text="@string/tvt"
                    android:textAlignment="center" /> <!-- srodkowanie w
starszych wersjach-->

                <TextView
                    android:id="@+id/poleAt"
                    android:layout_width="150dp"
                    android:layout_height="wrap_content"
                    android:layout_marginBottom="5dp"
```

```
                android:layout_weight="1"
                android:text="@string/zera"
                android:textAlignment="center" />


        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:gravity="center_horizontal">

            <TextView
                android:id="@+id/textView2"
                android:layout_width="150dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="center"
                android:text="@string/tv2"
                android:textAlignment="center" /> <!-- srodkowanie w
starszych wersjach-->

            <TextView
                android:id="@+id/poleAx"
                android:layout_width="150dp"
                android:layout_height="wrap_content"
                android:layout_marginBottom="5dp"
                android:layout_weight="1"
                android:text="@string/zera"
                android:textAlignment="center" />


        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:gravity="center_horizontal">

            <TextView
                android:id="@+id/textView4"
                android:layout_width="150dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="center"
                android:text="@string/tv3"
                android:textAlignment="center" /> <!-- srodkowanie w
starszych wersjach-->

            <TextView
                android:id="@+id/poleAy"
                android:layout_width="150dp"
                android:layout_height="20dp"
                android:layout_marginBottom="5dp"
                android:layout_weight="1"
                android:text="@string/zera"
                android:textAlignment="center" />


        </TableRow>
```

```
            <TableRow
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:gravity="center_horizontal">

                <TextView
                    android:id="@+id/textView3"
                    android:layout_width="150dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:gravity="center"
                    android:text="@string/tv4"
                    android:textAlignment="center" /> <!-- srodkowanie w
starszych wersjach-->

                <TextView
                    android:id="@+id/poleAz"
                    android:layout_width="150dp"
                    android:layout_height="wrap_content"
                    android:layout_marginBottom="5dp"
                    android:layout_weight="1"
                    android:text="@string/zera"
                    android:textAlignment="center" />


            </TableRow>

            <TableRow
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:gravity="center_horizontal">

                <TextView
                    android:id="@+id/tvlktv"
                    android:layout_width="150dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:gravity="center"
                    android:text="@string/tvlk"
                    android:textAlignment="center"
                    android:textStyle="bold" /> <!-- srodkowanie w
starszych wersjach-->

                <TextView
                    android:id="@+id/poleLK"
                    android:layout_width="150dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="0"
                    android:textAlignment="center"
                    android:textStyle="bold" />


            </TableRow>

        </TableLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp"
            android:layout_marginTop="5dp"
```

```
            android:orientation="horizontal">

            <Button
                android:id="@+id/btnStart"
                style="@android:style/Widget.Holo.Button"
                android:layout_width="20dp"
                android:layout_height="wrap_content"
                android:layout_marginLeft="20dp"
                android:layout_marginRight="20dp"
                android:layout_weight="1"
                android:background="@color/colorPrimary"
                android:onClick="startWc"
                android:text="@string/btn1"
                android:textAlignment="center"

                android:textSize="16sp" />

            <Button
                android:id="@+id/btnStop"
                android:layout_width="20dp"
                android:layout_height="wrap_content"
                android:layout_marginLeft="20dp"
                android:layout_marginRight="20dp"
                android:layout_weight="1"
                android:background="@color/colorPrimary"
                android:elevation="0dp"
                android:onClick="stopWc"
                android:text="@string/btn2"
                android:textAlignment="center"
                android:textColor="@android:color/background_light"
                android:textSize="16sp"
                android:visibility="visible" />
        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="40dp"
            android:layout_marginTop="3dp"
            android:orientation="horizontal">

            <Button
                android:id="@+id/btnSave"
                style="@android:style/Widget.Holo.Button"
                android:layout_width="20dp"
                android:layout_height="wrap_content"
                android:layout_marginLeft="20dp"
                android:layout_marginRight="1.5dp"
                android:layout_weight="1"
                android:background="@color/colorPrimary"
                android:onClick="saveFile"
                android:text="@string/btn3"
                android:textAlignment="center"
                android:textSize="16sp" />

            <Button
                android:id="@+id/btnReset"
                android:layout_width="20dp"
                android:layout_height="wrap_content"
                android:layout_marginLeft="1.5dp"
                android:layout_marginRight="20dp"
                android:layout_weight="1"
```

```
                android:background="@color/colorPrimary"
                android:elevation="0dp"
                android:onClick="reset"
                android:text="@string/btn4"
                android:textAlignment="center"
                android:textColor="@android:color/background_light"
                android:textSize="16sp"
                android:visibility="visible" />

        </LinearLayout>

        <LinearLayout
            android:id="@+id/chart"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical"></LinearLayout>

    </LinearLayout>
</android.support.constraint.ConstraintLayout>
```

strings.xml

```
<resources>
    <string name="app_name">Akcelerometr</string>

    <string name="tv1">Akcelerometr</string>

    <string name="tv2">Składowa X:</string>
    <string name="tv3">Składowa Y:</string>
    <string name="tv4">Składowa Z:</string>

    <string name="btn1">▷ Start</string>
    <string name="btn2">□ Stop</string>
    <string name="btn3">💾 Zapisz</string>
    <string name="btn4">"\u274C" Reset</string>
    <string name="tvt">Czas T:</string>
    <string name="tvlk">Liczba kroków:</string>
    <string name="zera">0</string>


</resources>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.tomek.akcelerometr">

    <!--
    w manifescie trzeba ustawic pozwolenia:

    <uses-permission android:name="android.permission.WAKE_LOCK"/> -
pozwalamy, zeby bylo obslugiwane dzialanie przycisku

     <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/> - pozwalamy na
zapis na karcie SD
    -->

    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity"
android:screenOrientation="portrait"> <!-- blokada obrotu ekranu -->
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

MainActivity.java

```
package com.tomek.akcelerometr;

import android.app.Activity;
import android.content.Context;
import android.graphics.Color;
import android.graphics.Paint;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.os.PowerManager;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import org.achartengine.ChartFactory;
import org.achartengine.GraphicalView;
import org.achartengine.chart.PointStyle;
import org.achartengine.model.XYMultipleSeriesDataset;
import org.achartengine.model.XYSeries;
import org.achartengine.renderer.XYMultipleSeriesRenderer;
import org.achartengine.renderer.XYSeriesRenderer;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintWriter;
import java.text.DecimalFormat;
import java.text.NumberFormat;


public class MainActivity extends Activity implements SensorEventListener {
//implements Sensor... - czujnik

    private SensorManager mySensorManager; //menedzer czujnikow
    private Sensor myAccelerometer; //obiekt klasy Sensor (czyli czujnik)
    private boolean pomiar = false; //czy nacisnieto przycisk Start?
    private boolean marker = false;

    private TextView poleAx, poleAy, poleAz, poleAt, poleLK;
    private String string = "T:" + "\t" + "X:" + "\t" + "Y:" + "\t" + "Z:"
+ "\n"; //naglowek do txt
    private int licznik = 0;
    private PowerManager.WakeLock mWakeLock;

    private float aX;
    private float aY;
    private float aZ;
    private double aT;
    private double NS2S = 0.000000001; //do konwersji nanosekund do sekund
    private double startTime;
    private double timeStop;
```

```
    // definicje serii
    private XYSeries xTSeria = new XYSeries("X(t)");
    private XYSeries yTSeria = new XYSeries("Y(t)");
    private XYSeries zTSeria = new XYSeries("Z(t)");


    private XYSeriesRenderer rendererXT;

    private XYSeriesRenderer rendererYT;

    private XYSeriesRenderer rendererZT;

    private XYMultipleSeriesRenderer mrenderer;

    private XYMultipleSeriesDataset dataset;

    private LinearLayout chartLayout;
    private GraphicalView chartView;

// algorytm liczenia krokow - przechowywanie wart. przysp
    private float xAccel;
    private float yAccel;
    private float zAccel;

//algorytm lcizenia krokow - wczesniejsze wawrtosci przysp
    private float oldAX;
    private float oldAY;
    private float oldAZ;
    private boolean firstUpdate = true; //true = pierwsza zmiana
przyspieszenia
    private final float shakeThreshold = 2f; //próg klasyfikacji jako
wstrząs
    private boolean shakeInitiated = false; //czy wstrzas zostal rozpoczety
(ruch w jednym kierunku)

    private int counter;

    Button buttonStart, buttonStop, buttonSave, buttonReset;

    PowerManager pm;

    NumberFormat numberFormat = new DecimalFormat("0.000000"); //format
wyswietlania liczb na ekranie + wartosci czasu w txt
    NumberFormat numberFormat2 = new DecimalFormat("0.0000000000");
//format wyswietlania wartosci XYZ w txt


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        //sensor
        mySensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        myAccelerometer =
mySensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        mySensorManager.registerListener(this, myAccelerometer,
SensorManager.SENSOR_DELAY_FASTEST);
```

```
        //umozliwienie dzialania programu pomimo zgaszonego ekranu
        pm = (PowerManager) getSystemService(Context.POWER_SERVICE);
        mWakeLock = pm.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK, "My
Tag");

        buttonStart = (Button) findViewById(R.id.btnStart);
        buttonStop = (Button) findViewById(R.id.btnStop);
        buttonSave = (Button) findViewById(R.id.btnSave);
        buttonReset = (Button) findViewById(R.id.btnReset);

        buttonStop.setVisibility(View.GONE);
        buttonSave.setVisibility(View.GONE);
        buttonReset.setVisibility(View.GONE);
        buttonStart.setVisibility(View.VISIBLE);

        //przygotowanie do zachowania funkcjonalnosci aplikacji po obrocie
urzadzenia

        if (savedInstanceState != null) {
            aX = savedInstanceState.getFloat("aX");
            aY = savedInstanceState.getFloat("aY");
            aZ = savedInstanceState.getFloat("aZ");
            aT = savedInstanceState.getFloat("aT");
            licznik = savedInstanceState.getInt("licznik");
            pomiar = savedInstanceState.getBoolean("pomiar");
            marker = savedInstanceState.getBoolean("marker");
            string = savedInstanceState.getString("string");
            startTime = savedInstanceState.getDouble("startTime");

        }

        if (marker) {
            buttonStart.setVisibility(View.GONE);
            buttonStop.setVisibility(View.VISIBLE);
            buttonSave.setVisibility(View.GONE);
            buttonReset.setVisibility(View.GONE);
            mWakeLock.acquire();
        }
    }

    @Override
    public void onSensorChanged(SensorEvent sensorEvent) { //gdy czujnik
wykryje zmiane


        if (pomiar) {

            int sensorType = sensorEvent.sensor.getType();

            if (sensorEvent.sensor.getType() == Sensor.TYPE_ACCELEROMETER)
;

            //if do ustalenia prawidlowego czasu - zapis pierwszego odczytu
i od kolejnych odejmowanie jego wartosci - czas startuje dzieki temu od
zera
            if (licznik == 0) {
                startTime = sensorEvent.timestamp;
            }

            aX = sensorEvent.values[0]; //skladowa X przyspieszenia
            aY = sensorEvent.values[1]; //skladowa Y przyspieszenia
```

```
            aZ = sensorEvent.values[2]; //skladowa Z przyspieszenia

            aT = (sensorEvent.timestamp - startTime) * NS2S; //timestamp z
konwersją na sekundy

            poleAx = (TextView) findViewById(R.id.poleAx);
            poleAy = (TextView) findViewById(R.id.poleAy);
            poleAz = (TextView) findViewById(R.id.poleAz);
            poleAt = (TextView) findViewById(R.id.poleAt);
            poleLK = (TextView) findViewById(R.id.poleLK);


            //wyswietlanie sformatowanych danych pomiarowych
            poleAx.setText(String.valueOf(numberFormat.format(aX)));
            poleAy.setText(String.valueOf(numberFormat.format(aY)));
            poleAz.setText(String.valueOf(numberFormat.format(aZ)));
            poleAt.setText(String.valueOf(numberFormat.format(aT)));

            //kolejne wiersze z parametrami - do zapisu w txt
            string = string + numberFormat.format(aT) + "\t" +
numberFormat2.format(aX) + "\t" + numberFormat2.format(aY) + "\t" +
numberFormat2.format(aZ) + "\n";

            // dodanie zmiennych do serii
            xTSeria.add(aT, aX);
            yTSeria.add(aT, aY);
            zTSeria.add(aT, aZ);


            licznik++;

            // umozliwienie wyswietlania wykresu "na biezaco"
            mrenderer.setXAxisMin(xTSeria.getMaxX() - 2);
            mrenderer.setXAxisMax(xTSeria.getMaxX());
            chartView.repaint();

// ALGORYTM POMIARU LICZBY KROKOW:
            updateAccelParameters(aX, aY, aZ);

            if ((!shakeInitiated) && isAccelerationChanged()) {
                shakeInitiated = true;
            } else if ((shakeInitiated) && isAccelerationChanged()) {
                executeShakeAction();

            } else if ((shakeInitiated) && (!isAccelerationChanged())) {
                shakeInitiated = false;
            }


        }

    }


    private void updateAccelParameters(float xNewAccel, float yNewAccel,
float zNewAccel) {

        if (firstUpdate) {
            oldAX = xNewAccel;
            oldAY = yNewAccel;
            oldAZ = zNewAccel;
```

```
            firstUpdate = false;
        } else {
            oldAX = xAccel;
            oldAY = yAccel;
            oldAZ = zAccel;
        }
        xAccel = xNewAccel;
        yAccel = yNewAccel;
        zAccel = zNewAccel;

    }

    private boolean isAccelerationChanged() {
        float deltaX = Math.abs(oldAX - xAccel);
        float deltaY = Math.abs(oldAY - yAccel);
        float deltaZ = Math.abs(oldAZ - zAccel);
        return (deltaX > shakeThreshold && deltaY > shakeThreshold)
                || (deltaX > shakeThreshold && deltaZ > shakeThreshold)
                || (deltaY > shakeThreshold && deltaZ > shakeThreshold);
    }

    private void executeShakeAction() {

        counter++;
        poleLK.setText(String.valueOf(counter));

    }


//KONIEC ALGORYTMU


    @Override
    public void onAccuracyChanged(Sensor sensor, int i) { //jak na razie
nie zwracamy na nia uwagi

    }

    //this - oznacza, ze uzywamy metod onSensorChanged oraz
onAccuracyChanged
    //NS2S - stala, ktora ewentualnie mozna sobie zdefiniowac : 1*10^-9
(nanoseconds to seconds - czas z urzadzenia jest w ns)
    //w pkt 2 - umozliwianie pomiaru przyspieszen przy wylaczonym ekranie.
Trzeba dodac .release(), gdyz jesli tego nie dodamy, to akcelerometr bedzie
caly czas dzialal

    public void startWc(View view) {
        buttonStart.setVisibility(View.GONE);
        buttonStop.setVisibility(View.VISIBLE);
        buttonSave.setVisibility(View.GONE);
        buttonReset.setVisibility(View.GONE);
        pomiar = true;
        marker = true;

        counter = 0;
        mWakeLock.acquire();

        //dane do wykresu
        dataset = new XYMultipleSeriesDataset();
        dataset.addSeries(xTSeria);
        dataset.addSeries(yTSeria);
```

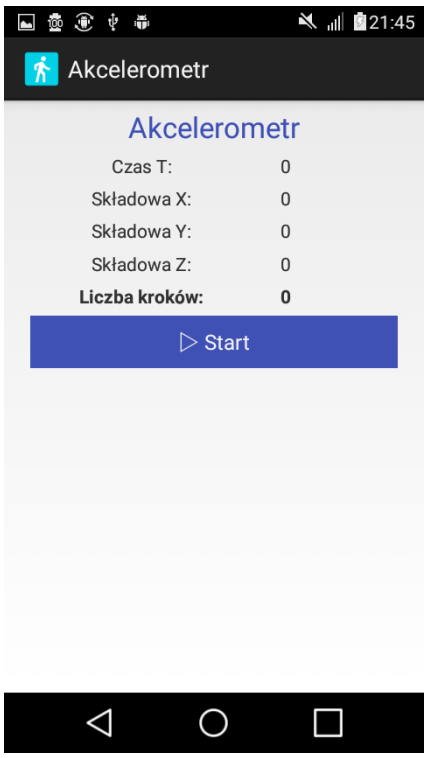
```
        dataset.addSeries(zTSeria);


        // rendery dla kolejnych serii
        rendererXT = new XYSeriesRenderer();
        rendererXT.setLineWidth(2);
        rendererXT.setColor(Color.BLUE);
        rendererXT.setPointStyle(PointStyle.CIRCLE);

        rendererYT = new XYSeriesRenderer();
        rendererYT.setLineWidth(2);
        rendererYT.setColor(Color.RED);
        rendererYT.setPointStyle(PointStyle.CIRCLE);

        rendererZT = new XYSeriesRenderer();
        rendererZT.setLineWidth(2);
        rendererZT.setColor(Color.GREEN);
        rendererZT.setPointStyle(PointStyle.CIRCLE);

        //multiple series render, modyfikacje estetyczne
        mrenderer = new XYMultipleSeriesRenderer();
        mrenderer.addSeriesRenderer(rendererXT);
        mrenderer.addSeriesRenderer(rendererYT);
        mrenderer.addSeriesRenderer(rendererZT);
        mrenderer.setYAxisMax(25);
        mrenderer.setYAxisMin(-25);
        mrenderer.setShowGrid(true);
        mrenderer.setMarginsColor(Color.WHITE);
        mrenderer.setGridColor(Color.LTGRAY);
        mrenderer.setAxesColor(Color.BLACK);
        mrenderer.setXLabelsColor(Color.BLACK);
        mrenderer.setYLabelsColor(0, Color.BLACK);
        mrenderer.setYLabelsAlign(Paint.Align.RIGHT);
        mrenderer.setLabelsTextSize(15);
        mrenderer.setLegendTextSize(15);

        dataset = new XYMultipleSeriesDataset();
        dataset.addSeries(xTSeria);
        dataset.addSeries(yTSeria);
        dataset.addSeries(zTSeria);

        chartLayout = (LinearLayout) findViewById(R.id.chart);
        chartView = ChartFactory.getLineChartView(this, dataset,
mrenderer);
        chartLayout.addView(chartView);
        chartView.repaint();
    }


    public void stopWc(View view) {
        if (pomiar) {
            buttonStart.setVisibility(View.VISIBLE);
            buttonStop.setVisibility(View.GONE);
            buttonSave.setVisibility(View.VISIBLE);
            buttonReset.setVisibility(View.VISIBLE);
            pomiar = false;
            timeStop = xTSeria.getMaxX();
            marker = true;
            mWakeLock.release();
        }
    }
```

```
    public void saveFile(View view) {
        zapiszPlik("/AKCELEROMETR/", "pomiar.txt");


    }

    private void zapiszPlik(String folder, String fileName) {


        File root = android.os.Environment.getExternalStorageDirectory();
        File dir = new File(root.getAbsolutePath() + folder); //tworzenie
nowego pliku w nowym folderze
        dir.mkdirs();
        File file = new File(dir, fileName);

        String test = file.getAbsolutePath();
        Log.i("My", "FILE LOCATION: " + test);


        try {
            FileOutputStream f = new FileOutputStream(file);
            PrintWriter pw = new PrintWriter(f);


            pw.print(string);

            pw.flush();
            pw.close();
            f.close();


            Toast.makeText(getApplicationContext(),

                    "Zapisano plik", //wiadomosc gdy zapis zakonczyl sie
sukcesem

                    Toast.LENGTH_LONG).show();


        } catch (FileNotFoundException e) {
            e.printStackTrace();
            Log.i("My", "******* File not found. Did you" +
                    " add a WRITE_EXTERNAL_STORAGE permission to the
manifest?");


        } catch (IOException e) {
            e.printStackTrace();
        }
    }


    public void reset(View view) {

        onCreate(new Bundle()); //na nowo metoda onCreate

        //czyszczenie danych dla kazdej serii
        xTSeria.clearSeriesValues();
        yTSeria.clearSeriesValues();
        zTSeria.clearSeriesValues();
```

```
        string = "T:" + "\t" + "X:" + "\t" + "Y:" + "\t" + "Z:" + "\n";
//zachowanie nagłówka do txt po resecie
    }


    //przygotowanie do zachowania funkcjonalnosci aplikacji po obrocie
urzadzenia
    @Override
    protected void onSaveInstanceState(Bundle savedInstanceState) {

        savedInstanceState.putFloat("aX",
Float.parseFloat(String.valueOf(aX)));
        savedInstanceState.putFloat("aY",
Float.parseFloat(String.valueOf(aY)));
        savedInstanceState.putFloat("aZ",
Float.parseFloat(String.valueOf(aZ)));
        savedInstanceState.putFloat("aT",
Float.parseFloat(String.valueOf(aT)));
        savedInstanceState.putInt("licznik", licznik);
        savedInstanceState.putBoolean("pomiar", pomiar);
        savedInstanceState.putBoolean("marker", marker);
        savedInstanceState.putString("string", string);
        savedInstanceState.putDouble("startTime", startTime);

    }

}

}
```

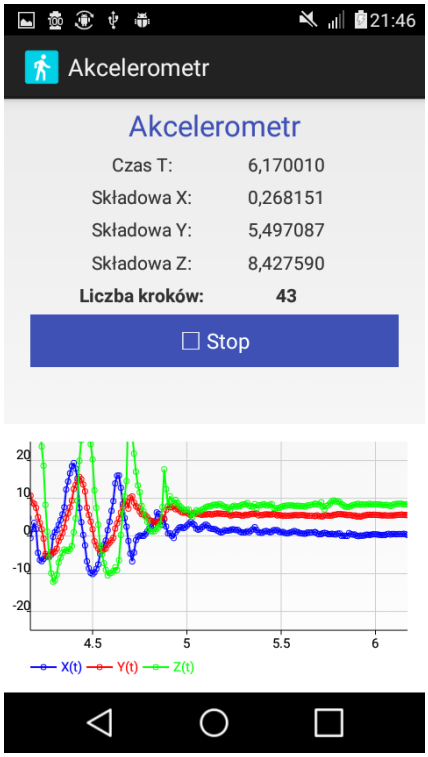
KOMENTARZ:

1. Ogólny opis:
    1) Aplikacja po jej starcie ukazuje użytkownikowi interfejs:



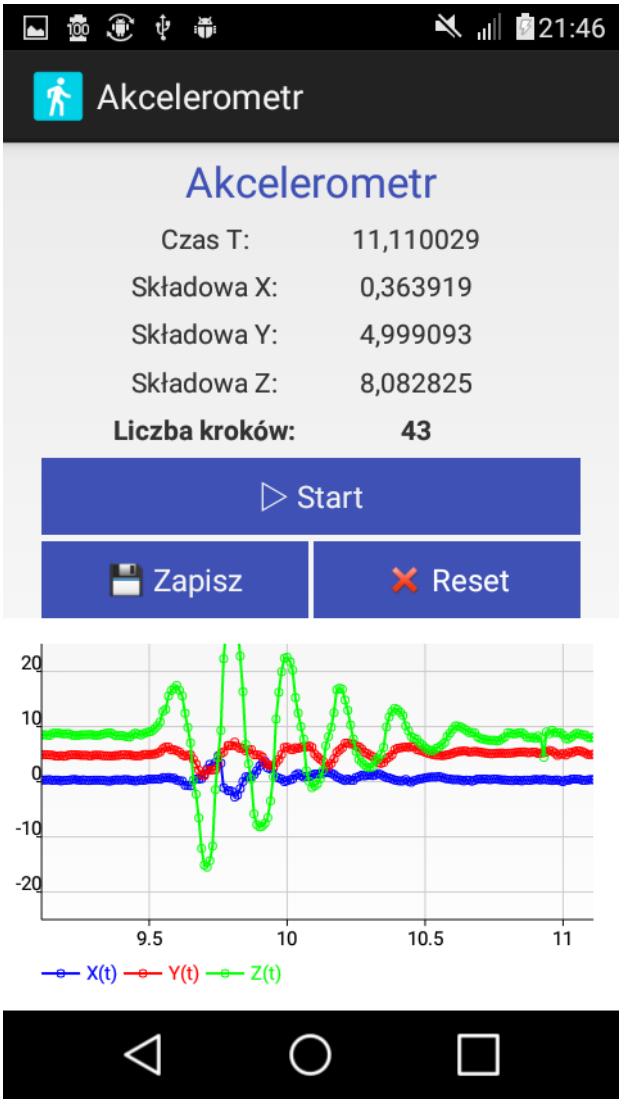
Użytkownik w tym momencie ma możliwość naciśnięcia przycisku „Start” i rozpoczęcia pomiaru.

    2) Rozpoczęcie pomiaru:

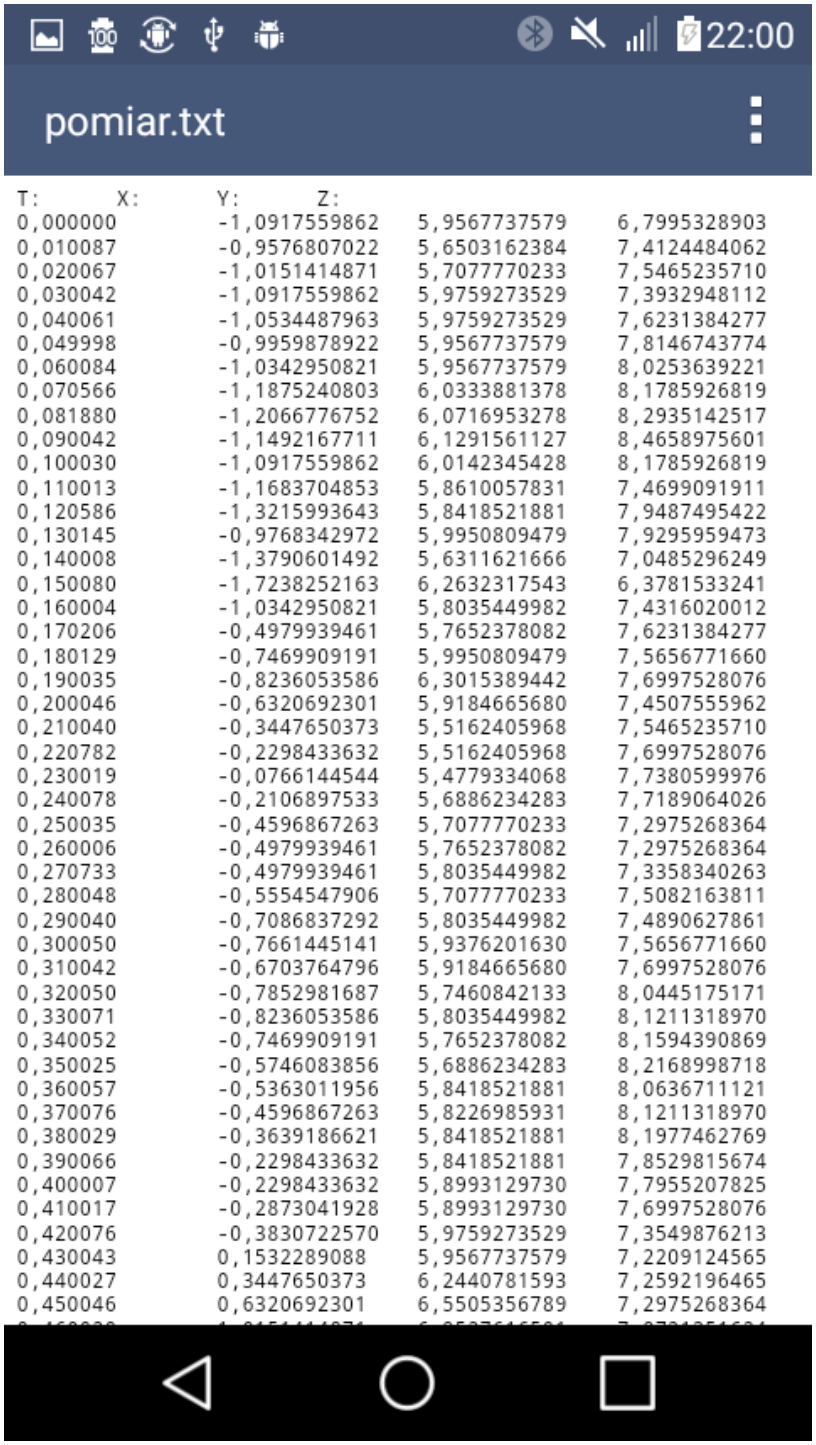


Znika przycisk „Start”, pojawia się na jego miejscu przycisk „Stop”. Następuje wyświetlanie aktualnych wartości czasu pomiaru oraz przyspieszeń w kierunkach osi X, Y i Z. Te wartości liczbowe są formatowane do stałych sześciu miejsc po przecinku dla zwiększenia czytelności. Poniżej wyświetla się wykres korzystający z biblioteki AChartEngine. Aktualizowany jest on na bieżąco, dzięki temu użytkownik od razu widzi efekty swoich ruchów urządzeniem. Gdy użytkownik będzie trząsł telefonem wyświetli się liczba „kroków”, którą wykonał.

3) Zatrzymanie pomiaru, zapis oraz reset:



Po wciśnięciu przycisku stop z pkt.2 na jego miejscu pojawia się z powrotem przycisk „Start” (możliwa kontynuacja pomiaru), a także przyciski „Zapisz” oraz „Reset”. Pomiar zostaje wstrzymany, tak samo rysowanie wykresu. Można go pomniejszać, powiększać i „scrollować”. Przycisk „Zapisz” umożliwia zapis i eksport danych do pamięci telefonu, w następującej formie:



T:	X:	Y:	Z:
0,000000		-1,0917559862	5,9567737579	6,7995328903
0,010087		-0,9576807022	5,6503162384	7,4124484062
0,020067		-1,0151414871	5,7077770233	7,5465235710
0,030042		-1,0917559862	5,9759273529	7,3932948112
0,040061		-1,0534487963	5,9759273529	7,6231384277
0,049998		-0,9959878922	5,9567737579	7,8146743774
0,060084		-1,0342950821	5,9567737579	8,0253639221
0,070566		-1,1875240803	6,0333881378	8,1785926819
0,081880		-1,2066776752	6,0716953278	8,2935142517
0,090042		-1,1492167711	6,1291561127	8,4658975601
0,100030		-1,0917559862	6,0142345428	8,1785926819
0,110013		-1,1683704853	5,8610057831	7,4699091911
0,120586		-1,3215993643	5,8418521881	7,9487495422
0,130145		-0,9768342972	5,9950809479	7,9295959473
0,140008		-1,3790601492	5,6311621666	7,0485296249
0,150080		-1,7238252163	6,2632317543	6,3781533241
0,160004		-1,0342950821	5,8035449982	7,4316020012
0,170206		-0,4979939461	5,7652378082	7,6231384277
0,180129		-0,7469909191	5,9950809479	7,5656771660
0,190035		-0,8236053586	6,3015389442	7,6997528076
0,200046		-0,6320692301	5,9184665680	7,4507555962
0,210040		-0,3447650373	5,5162405968	7,5465235710
0,220782		-0,2298433632	5,5162405968	7,6997528076
0,230019		-0,0766144544	5,4779334068	7,7380599976
0,240078		-0,2106897533	5,6886234283	7,7189064026
0,250035		-0,4596867263	5,7077770233	7,2975268364
0,260006		-0,4979939461	5,7652378082	7,2975268364
0,270733		-0,4979939461	5,8035449982	7,3358340263
0,280048		-0,5554547906	5,7077770233	7,5082163811
0,290040		-0,7086837292	5,8035449982	7,4890627861
0,300050		-0,7661445141	5,9376201630	7,5656771660
0,310042		-0,6703764796	5,9184665680	7,6997528076
0,320050		-0,7852981687	5,7460842133	8,0445175171
0,330071		-0,8236053586	5,8035449982	8,1211318970
0,340052		-0,7469909191	5,7652378082	8,1594390869
0,350025		-0,5746083856	5,6886234283	8,2168998718
0,360057		-0,5363011956	5,8418521881	8,0636711121
0,370076		-0,4596867263	5,8226985931	8,1211318970
0,380029		-0,3639186621	5,8418521881	8,1977462769
0,390066		-0,2298433632	5,8418521881	7,8529815674
0,400007		-0,2298433632	5,8993129730	7,7955207825
0,410017		-0,2873041928	5,8993129730	7,6997528076
0,420076		-0,3830722570	5,9759273529	7,3549876213
0,430043		0,1532289088	5,9567737579	7,2209124565
0,440027		0,3447650373	6,2440781593	7,2592196465
0,450046		0,6320692301	6,5505356789	7,2975268364

Zapis został uzyskany poprzez dodawanie kolejnych wartości do ciągu znaków - zmiennej string. Dla czasu zastosowano formatowanie do sześciu miejsc po przecinku, dla wartości X, Y oraz Z – do 10 miejsc po przecinku. W chwili, gdy zapis został przeprowadzony pomyślnie, wyświetla się powiadomienie „Zapisano plik”:

Przycisk „Reset” umożliwia powrót do metody onCreate() (wygląd jak na powyższym pierwszym zrzucie ekranu) oraz zresetowanie wartości serii danych wykresu i zmiennej „string”, co pozwala na poprawne ponowne wykonanie i zapis pomiaru.

2. Algorytm zliczania kroków:

Za zdarzenie określające wykonany krok algorytm przyjmuje zmianę przyspieszenia w jednym, a następnie w innym kierunku (wstrząs).

Po starcie aplikacji następuje kopiowanie wartości przyspieszenia ze zdarzenia SensorEvent do zmiennych określających stan. Służy do tego metoda updateAccelParameters(). Następnie sprawdzane jest, czy nastąpiła nagła zmiana przyspieszenia (oraz czy miało to miejsce pierwszy raz – jeśli tak, to zapis, że nastąpiła). Ponownie sprawdzane jest, czy zaszła kolejna zmiana przyspieszenia, lecz tym razem wykorzystywane są do tego zapisane wcześniej informacje. Jeśli taka zmiana nastąpiła, to rozumiemy to jako wstrząs, a więc zostaje wykonana metoda executeShakeAction() zliczająca kroki poprzez inkrementację zmiennej, której wartość jest potem wyświetlana. W ostatnim kroku zmienna markująca wstrząs jest zerowana. Metoda isAccelerationChanged() porównuje nowe wartości przyspieszenia ze starymi – jeśli przynajmniej dwie zmiany przekraczają próg – metoda zwraca true, czyli użytkownik wstrząsnął telefonem, a to oznacza, że wykonał krok.

Algorytm ten zakłada, że użytkownik nosi telefon w kieszeni spodni. Niestety próg „klasyfikacji wstrząsu” należałoby najlepiej dopasować do danego użytkownika.