**Listing 1. Opis interfejsu użytkownika w pliku layout/activity\_main.xml**

<?xml version=**"1.0"** encoding=**"utf-8"**?>

<LinearLayout xmlns:android=**"http://schemas.android.com/apk/res/android"**

android:layout\_width=**"match\_parent"**

android:layout\_height=**"match\_parent"**

android:orientation=**"vertical"** >

<TextView

android:id=**"@+id/temp\_textView"**

android:textColor=**"@android:color/white"**

android:textSize=**"40sp"**

android:textStyle=**"bold"**

android:padding=**"10sp"**

android:background=**"#59c7fa"**

android:layout\_width=**"match\_parent"**

android:layout\_height=**"wrap\_content"**/>

<com.github.mikephil.charting.charts.LineChart

android:id=**"@+id/temp\_chart"**

android:layout\_weight=**"1"**

android:layout\_width=**"match\_parent"**

android:layout\_height=**"match\_parent"**/>

<TextView

android:id=**"@+id/hum\_textView"**

android:textColor=**"@android:color/white"**

android:textSize=**"40sp"**

android:textStyle=**"bold"**

android:padding=**"10sp"**

android:background=**"#fa6868"**

android:layout\_width=**"match\_parent"**

android:layout\_height=**"wrap\_content"**/>

<com.github.mikephil.charting.charts.LineChart

android:id=**"@+id/hum\_chart"**

android:layout\_weight=**"1"**

android:layout\_width=**"match\_parent"**

android:layout\_height=**"match\_parent"**/>

</LinearLayout>

**Listing 2. Uzyskanie dostępu do magistrali I2C i modułu HTU21private I2cDevice htu21**

PeripheralManagerService service **=** **new** PeripheralManagerService**();**

/\* open HTU21 device \*/

**try** **{**

htu21 **=** service**.**openI2cDevice**(**"I2C1"**,** 0x40**);**

**}** **catch** **(**IOException e**)** **{**

Log**.**w**(**TAG**,** "Unable to access HTU21 device"**,** e**);**

**}**

**Listing 3. Generyczna funkcja dla pomiarów temperatury i wilgotności w trybie „Hold Master”**

private float htu21\_readData **(**I2cDevice htu21**,** int htu21\_cmd**)** **throws** Exception **{**

byte**[]** cmd **=** **{(**byte**)** htu21\_cmd**};**

byte**[]** output **=** **new** byte**[**3**];**

int data\_with\_crc**,** polynomial**;**

/\* trigger measurement - 'hold Master' mode \*/

htu21**.**write**(**cmd**,** 1**);**

htu21**.**read**(**output**,** 3**);**

/\* check CRC \*/

data\_with\_crc **=** output**[**0**]** **<<** 16 **|** **(**output**[**1**]** **&** 0xFF**)** **<<** 8 **|** **(**output**[**2**]** **&** 0xFF**);**

polynomial **=** 0x98800000**;** /\* CRC Polynomial: x^8 + x^5 + x^4 + 1 \*/

**for** **(**int cnt **=** 0**;** cnt **<** 24**;** cnt**++)** **{**

**if** **((**data\_with\_crc **&** 0x80000000**)** **!=** 0**)** data\_with\_crc **^=** polynomial**;**

data\_with\_crc **<<=** 1**;**

**}**

**if** **(**data\_with\_crc **!=** 0**)** **throw** **new** Exception **(**"CRC Error"**);**

**return** **(**output**[**0**]** **<<** 8 **|** **(**output**[**1**]** **&** 0xFF**));**

**}**

**Listing 4. Inicjalizacja czujnika HTU21 w metodzie onCreate()**

**if** **(**htu21 **!=** **null)** **{**

**if** **(**htu21\_softReset**(**htu21**))** **{**

htu21\_handler**.**postDelayed**(**htu21\_runnable**,** 1000**);**

**}** **else** **{**

/\* ERROR ! \*/

**/ .. /**

**}**

**}**

**Listing 5. Kod funkcji htu21\_softReset()**

private boolean htu21\_softReset **(**I2cDevice htu21**)** **{**

byte**[]** cmd **=** **{(**byte**)** 0xFE**};**

/\* trigger measurement - 'hold Master' mode \*/

**try** **{**

htu21**.**write**(**cmd**,** 1**);**

**}** **catch** **(**IOException e**)** **{**

**return** **false;**

**}**

**return** **true;**

**}**

**Listing 6. Kod funkcji htu21\_softReset()**

private Runnable htu21\_runnable **=** **new** Runnable**()** **{**

@Override

public void run**()** **{**

float temp**;**

float hum**;**

/\* read data from HTU21 \*/

**try** **{**

temp **=** htu21\_readData**(**htu21**,** **(**byte**)** 0xE3**);**

hum **=** htu21\_readData**(**htu21**,** **(**byte**)** 0xE5**);**

temp **\*=** 175.72**;**

temp **/=** 65536**;**

temp **-=** 46.85**;**

hum **\*=** 125**;**

hum **/=** 65536**;**

hum **-=** 6**;**

/\* AKTUALIZACJA GUI \*/

**/ ... /**

**}** **catch** **(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

/\* schedule another event after 1 sec. delay \*/

htu21\_handler**.**postDelayed**(**htu21\_runnable**,** 1000**);**

**}**

**};**

#Listing 7. Wyświetlenie prostego wzoru graficznego z wykorzystaniem sterownika SSD1306

**try** **{**

Ssd1306 ssd1306 **=** **new** Ssd1306**(**"I2C1"**);**

**for** **(**int i **=** 0**;** i **<** ssd1306**.**getLcdWidth**();** i**++)** **{**

**for** **(**int j **=** 0**;** j **<** ssd1306**.**getLcdHeight**();** j**++)** **{**

ssd1306**.**setPixel**(**i**,** j**,** **(**i **%** 2**)** **==** **(**j **%** 2**));**

**}**

**}**

ssd1306**.**show**();**

**}** **catch** **(**IOException e**)** **{**

Log**.**e**(**TAG**,** "Error while opening sdd1306 display"**,** e**);**

**}**

**Listing 8. Inicjalizacja komponentów graficznego interfejsu użytkownika**

protected void onCreate**(**Bundle savedInstanceState**)** **{**

**super.**onCreate**(**savedInstanceState**);**

setContentView**(**R**.**layout**.**activity\_main**);**

temp\_chart **=** findViewById**(**R**.**id**.**temp\_chart**);**

hum\_chart **=** findViewById**(**R**.**id**.**hum\_chart**);**

temp\_textView **=** findViewById**(**R**.**id**.**temp\_textView**);**

hum\_textView **=** findViewById**(**R**.**id**.**hum\_textView**);**

/\* setup charts \*/

setupChart **(**temp\_chart**,** Color**.**parseColor**(**"#59c7fa"**),** Color**.**parseColor**(**"#9cdefc"**),** 15**,** 35**);**

setupChart **(**hum\_chart**,** Color**.**parseColor**(**"#fa6868"**),** Color**.**parseColor**(**"#fc9c9c"**),** 45**,** 65**);**

**/ ... /**

**}**

**Listing 9. Konfiguracja wykresów LineChart w funkcji setupChart()**

private void setupChart **(**LineChart chart**,**

int bg\_color**,** int grid\_color**,**

int axis\_min**,** int axis\_max**)** **{**

/\* disable all 'interactions' with the chart \*/

chart**.**setTouchEnabled**(false);**

chart**.**setDragEnabled**(false);**

chart**.**setScaleEnabled**(false);**

chart**.**setPinchZoom**(false);**

/\* configure axis \*/

chart**.**getAxisRight**().**setEnabled**(false);**

chart**.**getXAxis**().**setEnabled**(false);**

/\* configure AxisLeft \*/

chart**.**getAxisLeft**().**setEnabled**(true);**

chart**.**getAxisLeft**().**setTextColor**(**Color**.**WHITE**);**

chart**.**getAxisLeft**().**setTextSize**(**15f**);**

chart**.**getAxisLeft**().**setGridColor**(**grid\_color**);**

chart**.**getAxisLeft**().**setGridLineWidth**(**1f**);**

chart**.**getAxisLeft**().**setGranularity**(**0.5f**);**

chart**.**getAxisLeft**().**setAxisMinimum**(**axis\_min**);**

chart**.**getAxisLeft**().**setAxisMaximum**(**axis\_max**);**

chart**.**getDescription**().**setEnabled**(false);**

chart**.**setDrawGridBackground**(false);**

chart**.**setBackgroundColor**(**bg\_color**);**

chart**.**setViewPortOffsets**(**70**,** 30**,** 70**,** 30**);**

chart**.**setData**(new** LineData**());**

/\* don't show legend \*/

Legend l **=** chart**.**getLegend**();**

l**.**setEnabled**(false);**

**}**

**Listing 10. Funkcja addEntry() dodająca nowe pomiary do wykresów temperatury i wilgotności**

private void addEntry **(**LineChart mChart**,** float value**)** **{**

LineData data **=** mChart**.**getData**();**

**if** **(**data **!=** **null)** **{**

ILineDataSet set **=** data**.**getDataSetByIndex**(**0**);**

**if** **(**set **==** **null)** **{**

set **=** createSet**();**

data**.**addDataSet**(**set**);**

**}**

data**.**addEntry**(new** Entry**(**set**.**getEntryCount**(),** value**),** 0**);**

data**.**notifyDataChanged**();**

/\* let the chart know it's data has changed \*/

mChart**.**notifyDataSetChanged**();**

/\* limit the number of visible entries \*/

mChart**.**setVisibleXRangeMaximum**(**60**);**

/\* move to the latest entry \*/

mChart**.**moveViewToX**(**data**.**getEntryCount**());**

**}**

**}**

**Listing 11. Kompletny kod klasy MainActivity**

package com**.**skalski**.**lukasz**.**androidthings\_i2c**;**

**import** android**.**app**.**Activity**;**

**import** android**.**graphics**.**Bitmap**;**

**import** android**.**graphics**.**BitmapFactory**;**

**import** android**.**graphics**.**Color**;**

**import** android**.**os**.**Bundle**;**

**import** android**.**os**.**Handler**;**

**import** android**.**util**.**Log**;**

**import** android**.**widget**.**TextView**;**

**import** java**.**io**.**IOException**;**

**import** java**.**util**.**Locale**;**

**import** com**.**google**.**android**.**things**.**contrib**.**driver**.**ssd1306**.**BitmapHelper**;**

**import** com**.**google**.**android**.**things**.**contrib**.**driver**.**ssd1306**.**Ssd1306**;**

**import** com**.**google**.**android**.**things**.**pio**.**I2cDevice**;**

**import** com**.**google**.**android**.**things**.**pio**.**PeripheralManagerService**;**

**import** com**.**github**.**mikephil**.**charting**.**charts**.**LineChart**;**

**import** com**.**github**.**mikephil**.**charting**.**components**.**Legend**;**

**import** com**.**github**.**mikephil**.**charting**.**data**.**Entry**;**

**import** com**.**github**.**mikephil**.**charting**.**data**.**LineData**;**

**import** com**.**github**.**mikephil**.**charting**.**data**.**LineDataSet**;**

**import** com**.**github**.**mikephil**.**charting**.**interfaces**.**datasets**.**ILineDataSet**;**

public class MainActivity **extends** Activity **{**

private static final String TAG **=** "MainActivity"**;**

private LineChart temp\_chart**;**

private LineChart hum\_chart**;**

private TextView temp\_textView**;**

private TextView hum\_textView**;**

private Ssd1306 ssd1306**;**

private I2cDevice htu21**;**

private Handler htu21\_handler **=** **new** Handler**();**

private static int HTU21\_ADDRESS **=** 0x40**;**

private static int HTU21\_TEMP\_HOLD\_MODE **=** 0xE3**;**

private static int HTU21\_HUM\_HOLD\_MODE **=** 0xE5**;**

private static int HTU21\_SOFT\_RESET **=** 0xFE**;**

@Override

protected void onCreate**(**Bundle savedInstanceState**)** **{**

**super.**onCreate**(**savedInstanceState**);**

setContentView**(**R**.**layout**.**activity\_main**);**

temp\_chart **=** findViewById**(**R**.**id**.**temp\_chart**);**

hum\_chart **=** findViewById**(**R**.**id**.**hum\_chart**);**

temp\_textView **=** findViewById**(**R**.**id**.**temp\_textView**);**

hum\_textView **=** findViewById**(**R**.**id**.**hum\_textView**);**

/\* setup charts \*/

setupChart**(**temp\_chart**,** Color**.**parseColor**(**"#59c7fa"**),** Color**.**parseColor**(**"#9cdefc"**),** 15**,** 35**);**

setupChart**(**hum\_chart**,** Color**.**parseColor**(**"#fa6868"**),** Color**.**parseColor**(**"#fc9c9c"**),** 45**,** 65**);**

/\* connect to ssd1306 display \*/

PeripheralManagerService service **=** **new** PeripheralManagerService**();**

**try** **{**

ssd1306 **=** **new** Ssd1306**(**"I2C1"**);**

Bitmap bmp **=** BitmapFactory**.**decodeResource**(**getResources**(),** R**.**drawable**.**logo**);**

BitmapHelper**.**setBmpData**(**ssd1306**,** 0**,** 0**,** bmp**,** **false);**

ssd1306**.**show**();**

**}** **catch** **(**IOException e**)** **{**

Log**.**e**(**TAG**,** "Error while opening sdd1306 display"**,** e**);**

**}**

/\* open HTU21 device \*/

**try** **{**

htu21 **=** service**.**openI2cDevice**(**"I2C1"**,** HTU21\_ADDRESS**);**

**}** **catch** **(**IOException e**)** **{**

Log**.**w**(**TAG**,** "Unable to access HTU21 device"**,** e**);**

**}**

/\* send RESET to HTU21 \*/

**if** **(**htu21 **!=** **null)** **{**

**if** **(**htu21\_softReset**(**htu21**))** **{**

htu21\_handler**.**postDelayed**(**htu21\_runnable**,** 1000**);**

**}** **else** **{**

temp\_textView**.**setText**(**getString**(**R**.**string**.**htu21\_error**));**

hum\_textView**.**setText**(**getString**(**R**.**string**.**htu21\_error**));**

**}**

**}**

**}**

/\*

\* htu21\_runnable

\*/

private Runnable htu21\_runnable **=** **new** Runnable**()** **{**

@Override

public void run**()** **{**

float temp**;**

float hum**;**

/\* read data from HTU21 \*/

**try** **{**

temp **=** htu21\_readData**(**htu21**,** HTU21\_TEMP\_HOLD\_MODE**);**

hum **=** htu21\_readData**(**htu21**,** HTU21\_HUM\_HOLD\_MODE**);**

temp **\*=** 175.72**;**

temp **/=** 65536**;**

temp **-=** 46.85**;**

hum **\*=** 125**;**

hum **/=** 65536**;**

hum **-=** 6**;**

/\* update charts \*/

addEntry**(**temp\_chart**,** temp**);**

addEntry**(**hum\_chart**,** hum**);**

/\* update textViews \*/

temp\_textView**.**setText**(**"Temperature: " **+** String**.**format**(**Locale**.**US**,** "%.1f"**,** temp**)** **+** " \u2103"**);**

hum\_textView**.**setText**(**"Humidity: " **+** **(**int**)**hum **+** "%"**);**

**}** **catch** **(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

/\* schedule another event after 1 sec. delay \*/

htu21\_handler**.**postDelayed**(**htu21\_runnable**,** 1000**);**

**}**

**};**

/\*

\* htu21\_readData()

\*/

private float htu21\_readData **(**I2cDevice htu21**,** int htu21\_cmd**)** **throws** Exception **{**

byte**[]** cmd **=** **{(**byte**)** htu21\_cmd**};**

byte**[]** output **=** **new** byte**[**3**];**

int data\_with\_crc**,** polynomial**;**

/\* trigger measurement - 'hold Master' mode \*/

htu21**.**write**(**cmd**,** 1**);**

htu21**.**read**(**output**,** 3**);**

/\* check CRC \*/

data\_with\_crc **=** output**[**0**]** **<<** 16 **|** **(**output**[**1**]** **&** 0xFF**)** **<<** 8 **|** **(**output**[**2**]** **&** 0xFF**);**

polynomial **=** 0x98800000**;** /\* CRC Polynomial: x^8 + x^5 + x^4 + 1 \*/

**for** **(**int cnt **=** 0**;** cnt **<** 24**;** cnt**++)** **{**

**if** **((**data\_with\_crc **&** 0x80000000**)** **!=** 0**)**

data\_with\_crc **^=** polynomial**;**

data\_with\_crc **<<=** 1**;**

**}**

**if** **(**data\_with\_crc **!=** 0**)**

**throw** **new** Exception **(**"CRC Error"**);**

**return** **(**output**[**0**]** **<<** 8 **|** **(**output**[**1**]** **&** 0xFF**));**

**}**

/\*

\* htu21\_softReset()

\*/

private boolean htu21\_softReset **(**I2cDevice htu21**)** **{**

byte**[]** cmd **=** **{(**byte**)** HTU21\_SOFT\_RESET**};**

/\* trigger measurement - 'hold Master' mode \*/

**try** **{**

htu21**.**write**(**cmd**,** 1**);**

**}** **catch** **(**IOException e**)** **{**

**return** **false;**

**}**

**return** **true;**

**}**

/\*

\* onDestroy()

\*/

@Override

protected void onDestroy**()** **{**

**super.**onDestroy**();**

/\* remove handler events on close \*/

htu21\_handler**.**removeCallbacks**(**htu21\_runnable**);**

/\* close I2C device \*/

**if** **(**htu21 **!=** **null)** **{**

**try** **{**

htu21**.**close**();**

htu21 **=** **null;**

**}** **catch** **(**IOException e**)** **{**

Log**.**w**(**TAG**,** "Unable to close I2C device"**,** e**);**

**}**

**}**

/\* close SSD1306 device \*/

**if** **(**ssd1306 **!=** **null)** **{**

**try** **{**

ssd1306**.**close**();**

ssd1306 **=** **null;**

**}** **catch** **(**IOException e**)** **{**

Log**.**w**(**TAG**,** "Unable to close SSD1306 device"**,** e**);**

**}**

**}**

**}**

/\*

\* setupChart()

\*/

private void setupChart **(**LineChart chart**,** int bg\_color**,** int grid\_color**,** int axis\_min**,** int axis\_max**)** **{**

/\* disable all 'interactions' with the chart \*/

chart**.**setTouchEnabled**(false);**

chart**.**setDragEnabled**(false);**

chart**.**setScaleEnabled**(false);**

chart**.**setPinchZoom**(false);**

/\* configure axis \*/

chart**.**getAxisRight**().**setEnabled**(false);**

chart**.**getXAxis**().**setEnabled**(false);**

/\* configure AxisLeft \*/

chart**.**getAxisLeft**().**setEnabled**(true);**

chart**.**getAxisLeft**().**setTextColor**(**Color**.**WHITE**);**

chart**.**getAxisLeft**().**setTextSize**(**15f**);**

chart**.**getAxisLeft**().**setGridColor**(**grid\_color**);**

chart**.**getAxisLeft**().**setGridLineWidth**(**1f**);**

chart**.**getAxisLeft**().**setGranularity**(**0.5f**);**

chart**.**getAxisLeft**().**setAxisMinimum**(**axis\_min**);**

chart**.**getAxisLeft**().**setAxisMaximum**(**axis\_max**);**

chart**.**getDescription**().**setEnabled**(false);**

chart**.**setDrawGridBackground**(false);**

chart**.**setBackgroundColor**(**bg\_color**);**

chart**.**setViewPortOffsets**(**70**,** 30**,** 70**,** 30**);**

chart**.**setData**(new** LineData**());**

/\* don't show legend \*/

Legend l **=** chart**.**getLegend**();**

l**.**setEnabled**(false);**

**}**

/\*

\* createSet()

\*/

private LineDataSet createSet**()** **{**

LineDataSet dataSet **=** **new** LineDataSet**(null,** "Test"**);**

dataSet**.**setLineWidth**(**2f**);**

dataSet**.**setCircleRadius**(**5f**);**

dataSet**.**setCircleHoleRadius**(**5f**);**

dataSet**.**setColor**(**Color**.**WHITE**);**

dataSet**.**setCircleColor**(**Color**.**WHITE**);**

dataSet**.**setDrawValues**(false);**

**return** dataSet**;**

**}**

/\*

\* addEntry()

\*/

private void addEntry **(**LineChart mChart**,** float value**)** **{**

LineData data **=** mChart**.**getData**();**

**if** **(**data **!=** **null)** **{**

ILineDataSet set **=** data**.**getDataSetByIndex**(**0**);**

**if** **(**set **==** **null)** **{**

set **=** createSet**();**

data**.**addDataSet**(**set**);**

**}**

data**.**addEntry**(new** Entry**(**set**.**getEntryCount**(),** value**),** 0**);**

data**.**notifyDataChanged**();**

/\* let the chart know it's data has changed \*/

mChart**.**notifyDataSetChanged**();**

/\* limit the number of visible entries \*/

mChart**.**setVisibleXRangeMaximum**(**60**);**

/\* move to the latest entry \*/

mChart**.**moveViewToX**(**data**.**getEntryCount**());**

**}**

**}**

**}**