1. Environment Sensors

1.1 Pressure Sensor

1. Create the Android application with the following attributes.
   - Application Name: **MySensor**
   - Project Name: **MySensor**
   - Package Name: **com.example.mysensor**

2. Check the layout file "**activity_main.xml**", enclose that the id for textView1 is exist.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView" />
</LinearLayout>
```

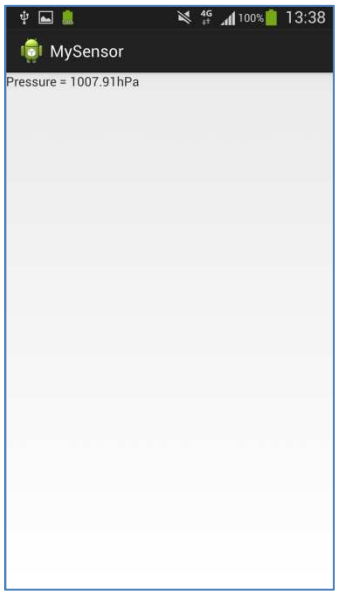
3. Modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
```

```
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_PRESSURE);
    }

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

    @Override
    public void onSensorChanged(SensorEvent event) {
        // Display the pressure on textview
        mTextView.setText("Pressure = " + event.values[0]);
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
```

```
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

4. Save and execute the app, you should able to obtain the pressure in your mobile (This app is not workable in emulator).



1.2 Ambient Temperature Sensor

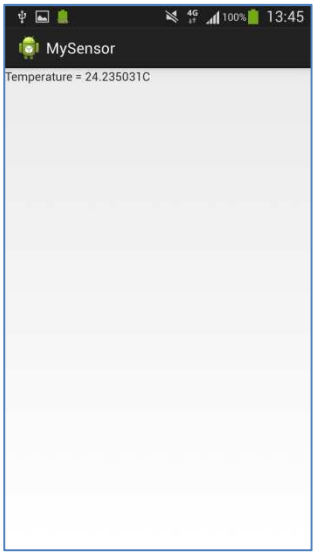
1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;


import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
```

```
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(
                                Sensor.TYPE_AMBIENT_TEMPERATURE);
    }

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

    @Override
    public void onSensorChanged(SensorEvent event) {
        // Display the temperature on textview
        mTextView.setText("Temperature = " + event.values[0] + "C");
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
```

```
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

2. Save and execute the app, you should able to obtain the ambient air temperature in your mobile (This app is not workable in emulator).



1.3 Light Sensor

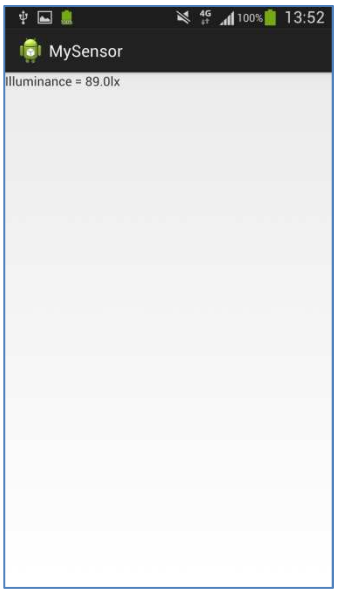
1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;


import android.os.Bundle;
```

```
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
    }

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

    @Override
    public void onSensorChanged(SensorEvent event) {
        // Display the Illuminance on textview
        mTextView.setText("Illuminance = " + event.values[0] + "lx");
    }

    @Override
```

```
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

2. Save and execute the app, you should able to obtain the illuminance in your mobile (This app is not workable in emulator).



1.4 Relative Humidity Sensor

1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_RELATIVE_HUMIDITY)
    }

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

    @Override
    public void onSensorChanged(SensorEvent event) {
```

```
        // Display the Ambient relative humidity on textview
        mTextView.setText("Relative Humidity = " + event.values[0] + "%");
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

2. Save and execute the app, you should able to obtain the ambient relative humidity in your mobile (This app is not workable in emulator).

2. Position Sensors

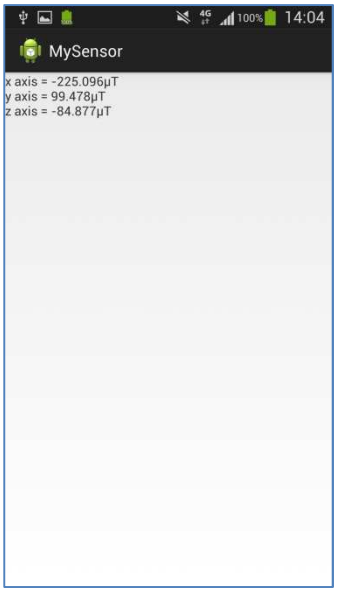
2.1 Geomagnetic Field Sensor

1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
    }
```

```
    @Override
    public void onSensorChanged(SensorEvent event) {
        // Display the geomagnetic field strength on textview
        mTextView.setText("x axis = " + event.values[0] + "μT\n" +
                          "y axis = " + event.values[1] + "μT\n" +
                          "z axis = " + event.values[2] + "μT");
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
                                    SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

2. Save and execute the app, you should able to obtain the geomagnetic field strength along the x, y, z axis from the mobile. (This app is not workable in emulator).



2.2 Proximity Sensor

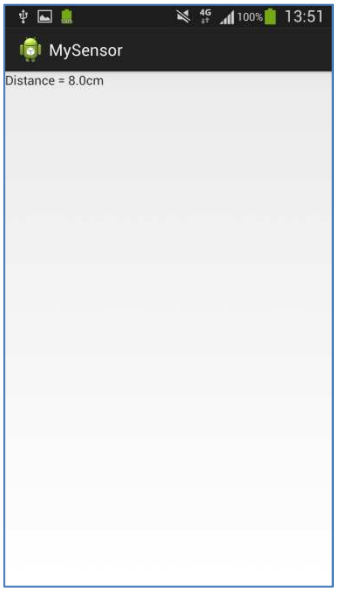
1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

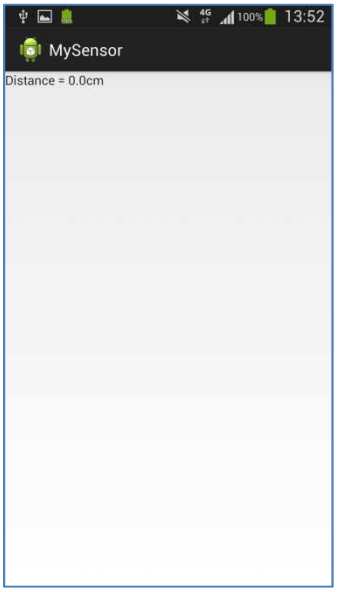
```
package com.example.mysensor;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

```
        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY);
    }

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

    @Override
    public void onSensorChanged(SensorEvent event) {
        // Display the distance on textview
        mTextView.setText("Distance = " + event.values[0] + "cm");
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
```

```
            return true;
        }
}
```

2. Save and execute the app, you should able to determine how far away an object is from a device. However, some proximity sensors such as Samsung S4 only provide binary values representing near and far. (This app is not workable in emulator).





3. Motion Sensors

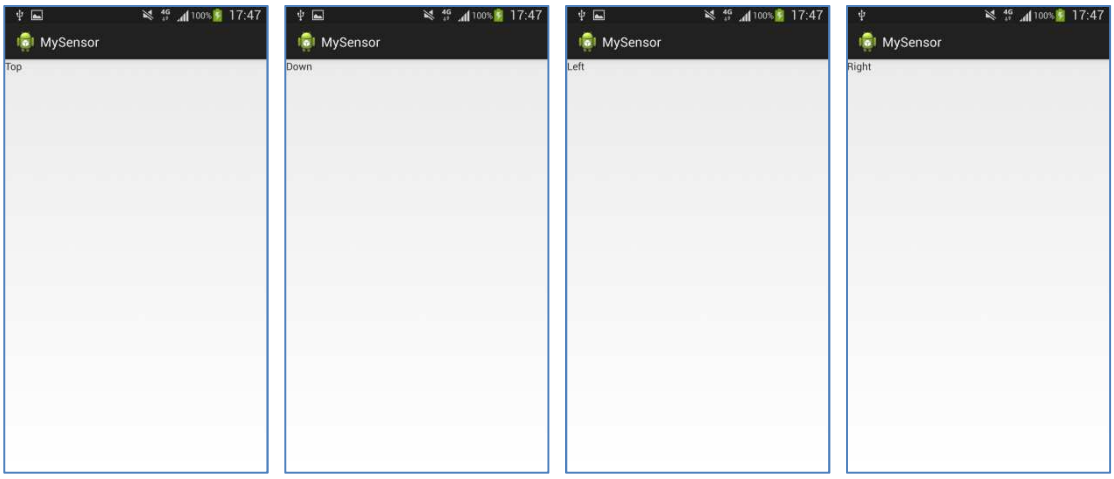
3.1 Rotation Vector Sensor

1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;

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

        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_ROTATION_VECTOR);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
    }
```

```
    @Override
    public void onSensorChanged(SensorEvent event) {
        // Obtain the rotate direction
        if (event.values[0] < 0 && event.values[1] > 0) {
            mTextView.setText("Top");
        } else if (event.values[0] > 0 && event.values[1] < 0) {
            mTextView.setText("Down");
        } else if (event.values[0] < 0 && event.values[1] < 0) {
            mTextView.setText("Left");
        } else if (event.values[0] > 0 && event.values[1] > 0) {
            mTextView.setText("Right");
        }
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
        mSensorManager.registerListener(this, mSensor,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

2. Save and execute the app, rotate your mobile to obtain the change. (This app is not workable in emulator)



3.2 Acceleration Sensor

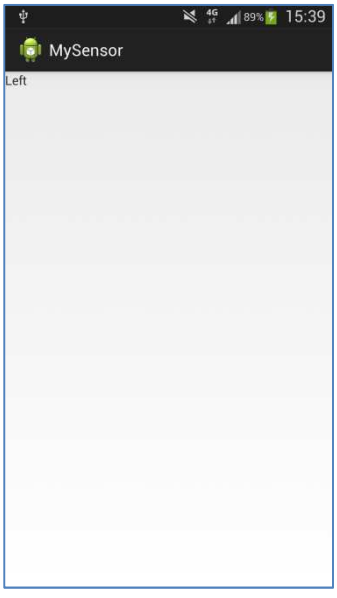
1. Open the previous project and modify the source file "**MainActivity.java**" as follow:

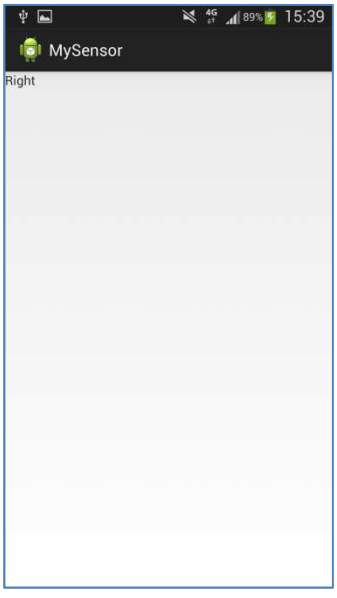
```
package com.example.mysensor;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mSensor;
    private TextView mTextView;
    float x, init_x;
    long lastUpdateTime = 0;

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

```
        // Finds a view that was identified by the ID attribute from the XML
        mTextView = (TextView) findViewById(R.id.textView1);

        // Get an instance of the sensor service, and use that to get an instance of
        // a particular sensor.
        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
    }
    @Override
    public void onSensorChanged(SensorEvent event) {
        // Capture the initial value
        if (init_x == 0) {
            init_x = event.values[0];
        }
        x = init_x - event.values[0];

        // Time interval = Current Time - Last Update Time
        long timeInterval = System.currentTimeMillis() - lastUpdateTime;

        // Obtain capture user shake > 300 ms
        if (timeInterval < 300) {
            lastUpdateTime = System.currentTimeMillis();
        } else {
            if (x < -5) {
                mTextView.setText("Left");
            } else if (x > 5) {
                mTextView.setText("Right");
            }
        }
    }

    @Override
    protected void onResume() {
        // Register a listener for the sensor
        super.onResume();
```

```
        mSensorManager.registerListener(this, mSensor,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // Be sure to unregister the sensor when the activity pauses.
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

2. Save and execute the app, you should able to obtain the direction (left or right) by shaking your mobile. (This app is not workable in emulator).



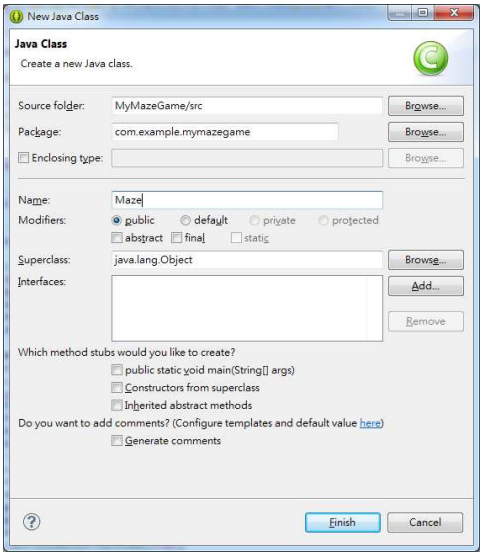


4. Maze Game

4.1 Maze Game

1. Create the Android application with the following attributes.
   - Application Name: **MyMazeGame**
   - Project Name: **MyMazeGame**
   - Package Name: **com.example.mymazegame**

2. Right click the package **src/com.example.mymazegame**, and select **New ➔ Class**. Input "**Maze**" as the class name and press **[Finish]** button.



3. Modify the source code for "**Maze.java**" as follow:

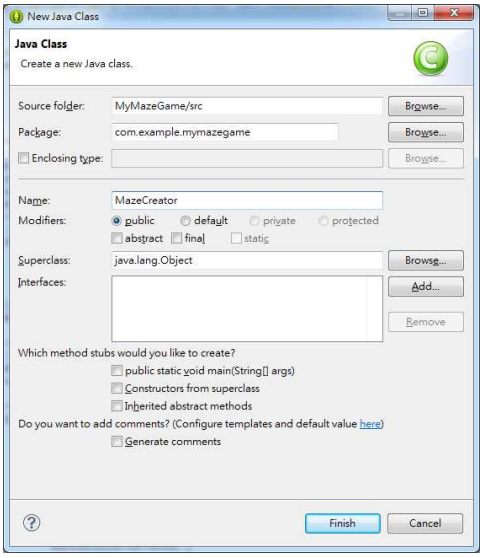
```
package com.example.mymazegame;


import java.io.Serializable;


public class Maze implements Serializable {
    private static final long serialVersionUID = 1L;
    public static final int UP = 0, DOWN = 1, RIGHT = 2, LEFT = 3;
    private boolean[][] verticalLines;
    private boolean[][] horizontalLines;
    private int sizeX, sizeY;          // Stores the width and height of the maze
    private int currentX, currentY;    // Stores the current location of the ball
    private int finalX, finalY;        // Stores the finishing of the maze
    private boolean gameComplete;
```

```
    public int getMazeWidth() {
        return sizeX;
    }

    public int getMazeHeight() {
        return sizeY;
    }

    public boolean move(int direction) {
        boolean moved = false;
        if (direction == UP) {
            if(currentY != 0 && !horizontalLines[currentY-1][currentX]) {
                currentY--;
                moved = true;
            }
        }
        if (direction == DOWN) {
            if(currentY != sizeY-1 && !horizontalLines[currentY][currentX]) {
                currentY++;
                moved = true;
            }
        }
        if (direction == RIGHT) {
            if(currentX != sizeX-1 && !verticalLines[currentY][currentX]) {
                currentX++;
                moved = true;
            }
        }
        if (direction == LEFT) {
            if(currentX != 0 && !verticalLines[currentY][currentX-1]) {
                currentX--;
                moved = true;
            }
        }
        if (moved) {
            if(currentX == finalX && currentY == finalY) {
                gameComplete = true;
            }
        }
```

```
        return moved;
    }

    public boolean isGameComplete() {
        return gameComplete;
    }

    public void setStartPosition(int x, int y) {
        currentX = x;
        currentY = y;
    }

    public int getFinalX() {
        return finalX;
    }

    public int getFinalY() {
        return finalY;
    }

    public void setFinalPosition(int x, int y) {
        finalX = x;
        finalY = y;
    }

    public int getCurrentX() {
        return currentX;
    }

    public int getCurrentY() {
        return currentY;
    }

    public boolean[][] getHorizontalLines() {
        return horizontalLines;
    }

    public void setHorizontalLines(boolean[][] lines) {
        horizontalLines = lines;
```

```
            sizeX = horizontalLines[0].length;
        }

        public boolean[][] getVerticalLines() {
            return verticalLines;
        }

        public void setVerticalLines(boolean[][] lines) {
            verticalLines = lines;
            sizeY = verticalLines.length;
        }
}
```

4. Right click the package **src/com.example.mymazegame**, and select **New ➔ Class**. Input "**MazeCreator**" as the class name and press **[Finish]** button.

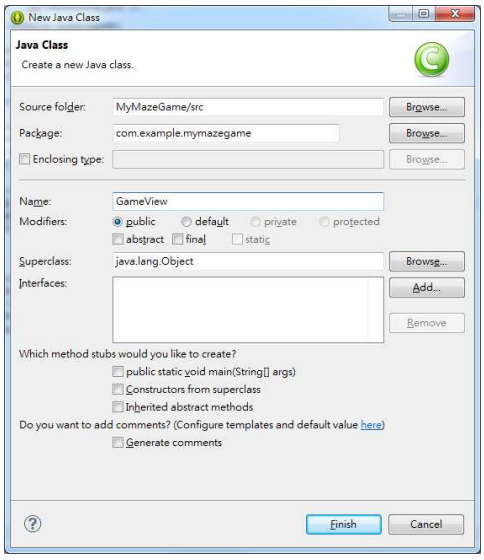


5. Modify the source code for "**MazeCreator.java**" as follow:

```
package com.example.mymazegame;

public class MazeCreator {
    public static Maze getMaze() {
        Maze maze = null;

        maze = new Maze();
        boolean[][] vLines = new boolean[][]{
                {true ,false,false,false,true ,false,false},
                {true ,false,false,true ,false,true ,true },
```

```
                {false,true ,false,false,true ,false,false},
                {false,true ,true ,false,false,false,true },
                {true ,false,false,false,true ,true ,false},
                {false,true ,false,false,true ,false,false},
                {false,true ,true ,true ,true ,true ,false},
                {false,false,false,true ,false,false,false} };

        boolean[][] hLines = new boolean[][]{
                {false,false,true ,true ,false,false,true ,false},
                {false,false,true ,true ,false,true ,false,false},
                {true ,true ,false,true ,true ,false,true ,true },
                {false,false,true ,false,true ,true ,false,false},
                {false,true ,true ,true ,true ,false,true ,true },
                {true ,false,false,true ,false,false,true ,false},
                {false,true ,false,false,false,true ,false,true } };

        maze.setVerticalLines(vLines);
        maze.setHorizontalLines(hLines);
        maze.setStartPosition(0, 0);
        maze.setFinalPosition(7, 7);

        return maze;
    }
}
```

6. Right click the package **src/com.example.mymazegame**, and select **New ➔ Class**. Input "**GameView**" as the class name and press **[Finish]** button:



7. Modify the source code for "**GameView.java**" as follow

```
package com.example.mymazegame;


import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.graphics.Canvas;
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.view.View;

public class GameView extends View implements SensorEventListener {
    // Width and height of the whole maze and width of lines which make the walls
    private int width, height, lineWidth;

    // Size of the maze i.e. number of cells in it
    private int mazeSizeX, mazeSizeY;

    // Width and height of cells in the maze
    float cellWidth, cellHeight;

    // Store result of cellWidth+lineWidth and cellHeight+lineWidth respectively
    float totalCellWidth, totalCellHeight;

    // The finishing point of the maze
    private int mazeFinishX, mazeFinishY;
    private Maze maze;
    private Activity context;
    private Paint line, red, background;
    boolean dragging = false;

    // Sensor for motion detection
    public SensorManager mSensorManager;
    public Sensor mSensor;
```

```
public GameView(Context context, Maze maze) {
     super(context);
     this.context = (Activity)context;
     this.maze = maze;
     mazeFinishX = maze.getFinalX();
     mazeFinishY = maze.getFinalY();
     mazeSizeX = maze.getMazeWidth();
     mazeSizeY = maze.getMazeHeight();
     line = new Paint();
     line.setColor(Color.BLACK);
     red = new Paint();
     red.setColor(Color.RED);
     background = new Paint();
     background.setColor(Color.YELLOW);
     setFocusable(true);
     this.setFocusableInTouchMode(true);

     // Get an instance of the sensor service, and use that to get an instance of
     // a particular sensor.
     mSensorManager = (SensorManager)
                  context.getSystemService(Context.SENSOR_SERVICE);
     mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_ROTATION_VECTOR);
     mSensorManager.registerListener(this, mSensor,
                  SensorManager.SENSOR_DELAY_NORMAL);
}

protected void onSizeChanged(int w, int h, int oldw, int oldh) {
     width = (w < h)?w:h;
     height = width;            // For now square mazes
     lineWidth = 1;             // For now 1 pixel wide walls
     cellWidth = (width - ((float)mazeSizeX*lineWidth)) / mazeSizeX;
     totalCellWidth = cellWidth+lineWidth;
     cellHeight = (height - ((float)mazeSizeY*lineWidth)) / mazeSizeY;
     totalCellHeight = cellHeight+lineWidth;
     red.setTextSize(cellHeight*0.75f);
     super.onSizeChanged(w, h, oldw, oldh);
}
```

```
protected void onDraw(Canvas canvas) {
    // Fill in the background
    canvas.drawRect(0, 0, width, height, background);

    boolean[][] hLines = maze.getHorizontalLines();
    boolean[][] vLines = maze.getVerticalLines();

    // Iterate over the boolean arrays to draw walls
    for(int i = 0; i < mazeSizeX; i++) {
        for(int j = 0; j < mazeSizeY; j++){
            float x = j * totalCellWidth;
            float y = i * totalCellHeight;
            if (j < mazeSizeX - 1 && vLines[i][j]) {
                // we'll draw a vertical line
                canvas.drawLine(x + cellWidth, // Start X
                                y,             // Start Y
                                x + cellWidth, // Stop X
                                y + cellHeight, // Stop Y
                                line);
            }
            if (i < mazeSizeY - 1 && hLines[i][j]) {
                // We'll draw a horizontal line
                canvas.drawLine(x,             // StartX
                                y + cellHeight, // StartY
                                x + cellWidth, // StopX
                                y + cellHeight, // StopY
                                line);
            }
        }
    }

    int currentX = maze.getCurrentX(),currentY = maze.getCurrentY();

    // Draw the ball
    canvas.drawCircle((currentX * totalCellWidth)+(cellWidth/2),  // x
            (currentY * totalCellHeight)+(cellWidth/2),          // y
            (cellWidth*0.45f),                                   // Radius
            red);
```

```
        // Draw the finishing point indicator
        canvas.drawText("F",
                (mazeFinishX * totalCellWidth)+(cellWidth*0.25f),
                (mazeFinishY * totalCellHeight)+(cellHeight*0.75f),
                red);
    }

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

    @Override
    public void onSensorChanged(SensorEvent event) {
        if (true) {
            // Either X or Y changed
            boolean moved = false;

            // Obtain the rotate direction
            if (event.values[0] < 0 && event.values[1] > 0) {
                moved = maze.move(Maze.UP);
            } else if (event.values[0] > 0 && event.values[1] < 0) {
                moved = maze.move(Maze.DOWN);
            } else if (event.values[0] < 0 && event.values[1] < 0) {
                moved = maze.move(Maze.LEFT);
            } else if (event.values[0] > 0 && event.values[1] > 0) {
                moved = maze.move(Maze.RIGHT);
            }

            if (moved) {
                // The ball was moved so we'll redraw the view
                invalidate();
                if(maze.isGameComplete()) {
                    // Creating alert Dialog with one Button
                    AlertDialog alertDialog1 = new
                            AlertDialog.Builder(context).create();

                    // Setting Dialog Title
                    alertDialog1.setTitle("Level Complete");
```

```
                    // Setting Dialog Message
                    alertDialog1.setMessage("Press [OK] to quit");

                    // Setting OK Button
                    alertDialog1.setButton("OK", new
                            DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                int which) {
                            // Handle button click
                            context.finish();
                        }
                    });

                    // Showing Alert Message
                    alertDialog1.show();
                }
            }
        }
    }
}
```

8. Modify the source code for "**MainActivity.java**" as follow.

```
package com.example.mymazegame;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;

public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
//      setContentView(R.layout.activity_main);

        // Use helper class for creating the Maze
        Maze maze = MazeCreator.getMaze();
        GameView view = new GameView(this, maze);
        setContentView(view);
    }
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

9. Save and execute the app, then you can rotate your mobile to play the game.