Exercise 4: Sensors

Due: Mar. 19, 23:59 PDT

Overview

This exercise provides examples for using various sensors.

There will be 2 different apps (weather & shaker). Although they are unrelated to each other, please put both into one project named “Exercise4YourName”, similar to what you did for exercise 3.

App 1: Weather

This app gets data from the ambient light and proximity sensors and displays that data.

App 2: Shaker

The app uses accelerometer sensor. It displays the x, y and z axis accelerometer readings in a continuous fashion. If the phone is shaken, the background color of the text changes and a toast is displayed, as shown in Fig. 1.

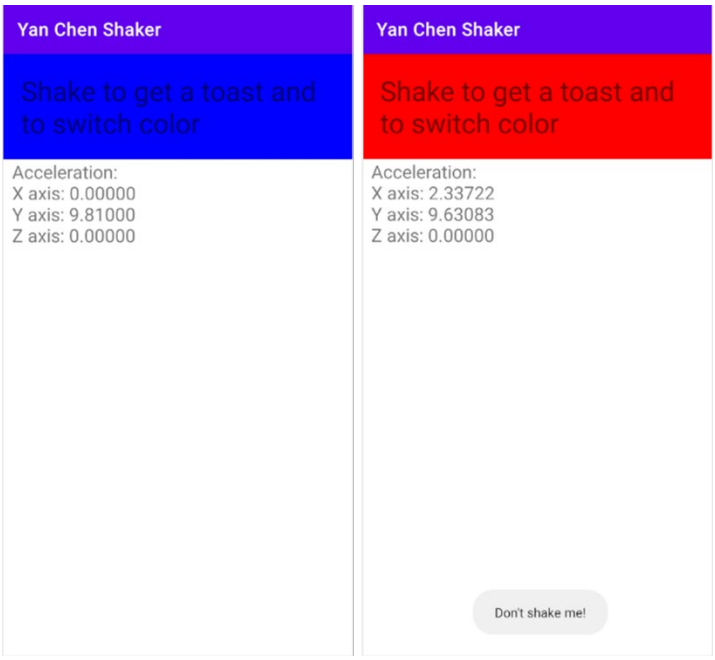

Fig. 1 Sample run of the shaker app. Left: when the device does not move; Right: when the device is shaken.

Prerequisites:

- Know the process of submitting your work (exercise 0)
- Be familiar with previous topics related to project structure and UI (exercise 1 - 3, lesson 2 - 4)
- Has set up an emulator (API 29)

Procedure related to the above topics will not be provided in the instruction. Refer to corresponding exercise/lecture notes if needed.

If you set any different view id’s, filenames, etc., remember to modify the corresponding part of code.

Step 0. Change module name

Suppose the new added module is “shaker”, let’s change the default module (“app)” added when you created a project to “weather”.

Right-click on the module (“app”) -> Refactor -> Rename. The curial part is you should choose “Rename module” in the popup window.

To change your package name, right-click on the package Refactor -> Rename, and choose “Rename package”. Let’s change the package name to “weather” instead of exercise4yourname. The result file list should be similar to Fig. 2. You may need to sync your project (shown in Exercise 2).

Note that the modules are ordered alphabetically, not chronologically.

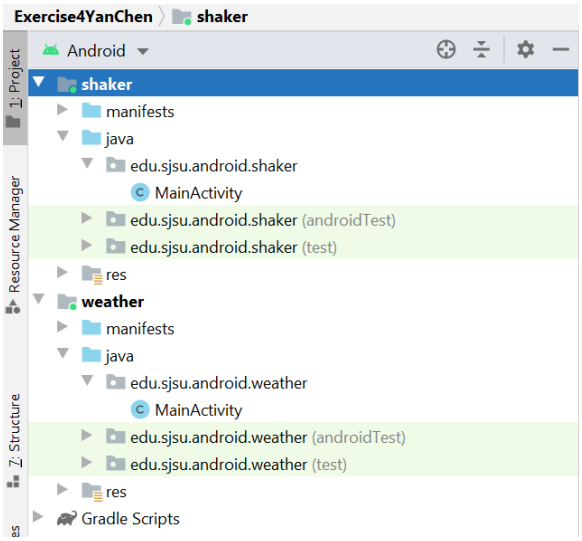

Fig. 2 File list

Step 1. Finish the weather app

Here is the instruction for completing the weather app: https://developer.android.com/codelabs/advanced-android-training-sensor-data?index=..%2F..advanced-android-training#3 (Task 2 of the code lab).

Note that the solution code (a GitHub repository) is also given. To have a better practice, please follow the steps and do that on your own instead of just cloning the solution code. Please put an extra TextView to show your name as shown in Fig. 3.

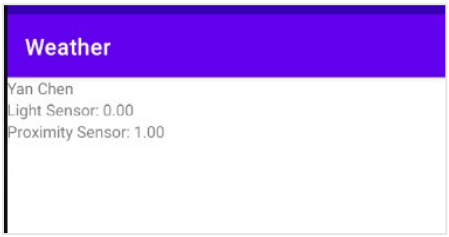

Fig. 3 A sample run of the weather app. Make sure your name is there and the app name should be Weather

Step 2 and step 3 are for the shaker app.

Step 2. Setup the UI

Change the app name to "Your Name Shaker".

The positions of the widgets do not matter but you should include the elements listed below.

Widget	id	Text
TextView	text	"Shake to get a toast and to switch color"
TextView	result	Will set later

Step 3. Implement MainActivity

The MainActivity needs to implement SensorEventListener interface. The interface includes 2 methods; add them by pressing Alt + Enter (Option + Enter for Mac) and choose "Implement methods". We will implement the onSensorChanged later.

Here are the class attributes besides binding or TextView objects you need:

```
private SensorManager manager;
private Sensor acc;
private long lastUpdate = System.currentTimeMillis();
```

1.1 Identify if There is an Accelerometer

We will use accelerometer. So first check if the device has an accelerometer in the onCreate method

```
manager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
if (manager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) != null)
    acc = manager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
else
    binding.result.setText("Missing accelerometer");
```

1.2 Register and Unregister Sensor Event Listener

Register the listener in onStart and unregister the listener in onStop. Note that we can use "this" here since the current class implements SensorEventListener interface. And SENSOR_DELAY_UI is the predefined rate "suitable for the user interface".

```
@Override
protected void onStart() {
    super.onStart();
    if(acc != null)
        manager.registerListener(this, acc, SensorManager.SENSOR_DELAY_UI);
}

@Override
protected void onStop() {
    super.onStop();
    manager.unregisterListener(this);
}
```

1.3 Monitor the Sensor Event

Implement the onSensorChanged method to monitor the sensor event. Wrap the code in an if to ensure we are monitor the accelerometer.

```
@Override
public void onSensorChanged(SensorEvent sensorEvent) {
    if(sensorEvent.sensor.getType() == Sensor.TYPE_ACCELEROMETER){
        // Your code here
    }
}
```

First, let's get the sensor data and display that on the screen. Recall that the data for accelerometer is a float array with 3 floats representing the acceleration with gravity of the 3 axes.

```
float x = sensorEvent.values[0];
float y = sensorEvent.values[1];
float z = sensorEvent.values[2];

String result = String.format("Acceleration:\n" +
        "X axis: %.5f\nY axis: %.5f\nZ axis: %.5f",
        x, y, z);
binding.result.setText(result);
```

Then let's toast a message and change color if a shake is detected. The algorithm is modified from this tutorial: https://www.vogella.com/tutorials/AndroidSensor/article.html. There are other possible algorithms (probably better) to detect a shake, you can do some research if you are interested.

```
float accSquareRoot = (x * x + y * y + z * z)
            / (SensorManager.GRAVITY_EARTH * SensorManager.GRAVITY_EARTH);
long actualTime = System.currentTimeMillis();
if (accSquareRoot >= 2 && actualTime - lastUpdate > 200) {
    lastUpdate = actualTime;
    Toast.makeText(this, "Don't shake me!", Toast.LENGTH_LONG).show();
    binding.text.setBackgroundColor(Color.RED);
} else binding.text.setBackgroundColor(Color.BLUE);
```

Step 4. Test your apps

Test the two apps using virtual sensors as shown in Lesson 11, pages 7 - 9. For the weather app, test it by changing the values of the corresponding sensors under the "Additional sensors" tab. For the shaker app, under the "Device pose" tab, switch to "Move" mode, then drag and move the emulator. When your movement is large and fast enough, the color of the top text will change to red, and a toast will popup. It may be easier to take a video instead of a screenshot to show that your app works.

Submission

- Push your project to a Bitbucket repository (name it “exercise4”) by the due date.
- Invite and share your Bitbucket repository the grader (edmond.lin@sjsu.edu) and the instructor (yan.chen01@sjsu.edu).
- Submit repository links, etc. by answering all the questions in the “Exercise 4 - Sensors” quiz on Canvas.
- Only your last submission before deadline will be graded based on the following criteria:
  2 pts if meets all requirements (for both apps);
  1 pt if app failed/missing any requirement.