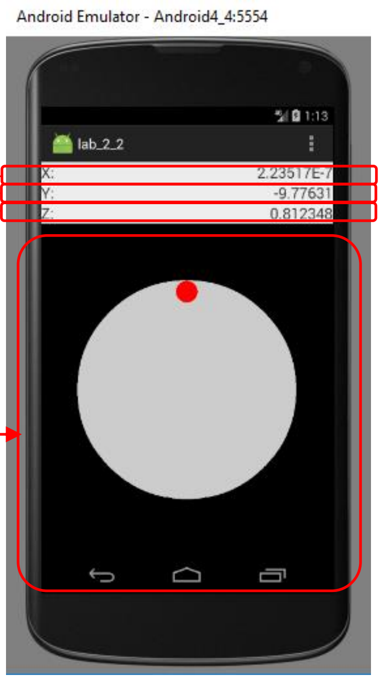
Lab 2	Leveler

Target:

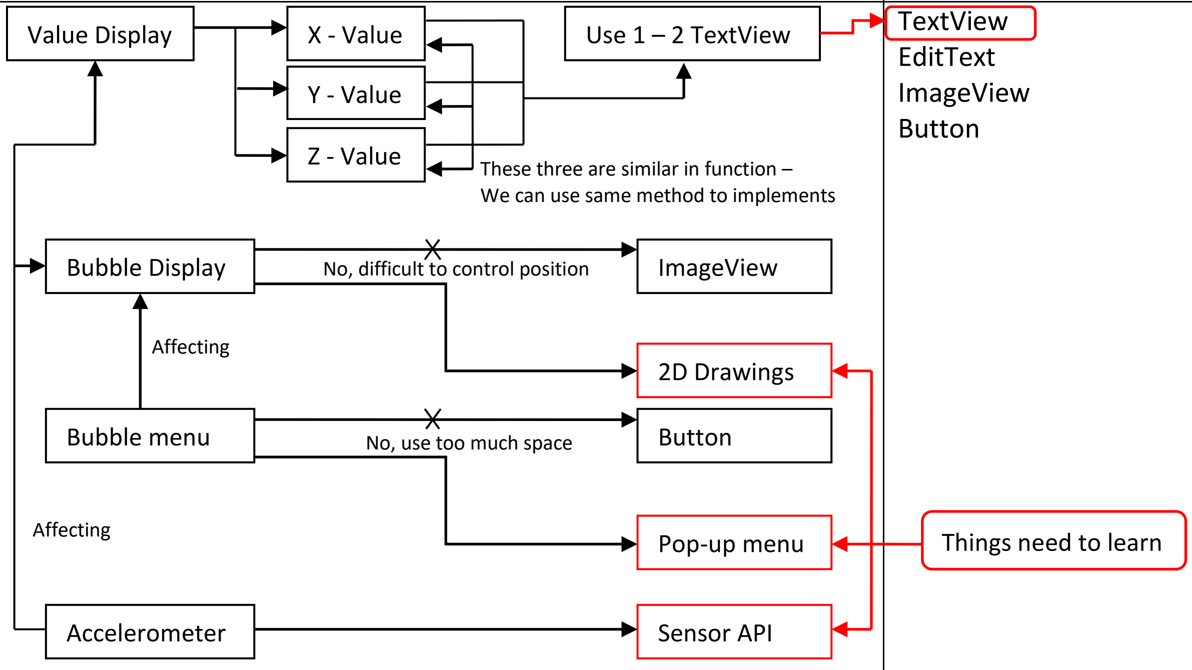
A Leveler

- Value of accelerometer display on the top of the Apps.
    - X value
    - Y value
    - Z value
- A “Bubble” display
- A menu to choose the “Bubble”
    - Different colors
        - Red
        - Green
        - Blue
        - White
    - Different shapes
        - Circle
        - Square
        - Diamond
        - Arc shape



Program design (To understand what you need and what you have)

You need to have	You learned in lab 1
	TextView EditText ImageView Button



Program procedure (To plan what should need to do to reach the target)

Task 1 – Learn 2D Drawings

Task 2 – Learn pop-up menu and control the bubble

Task 3 – Learn sensor API and finish the Apps

Start to do:

Task 1	Learn 2D Drawings

Knowledge learn in this task:

2D Drawings	1) Require a custom surface view. 2) In the custom view, we need to implement onDraw function for custom drawings.

Procedure of the task:

Step 1

Create new project in eclipse

Project name: Lab_2 (or your own one)

Build Target: Android 4.4

Package name: ece.course.lab_2

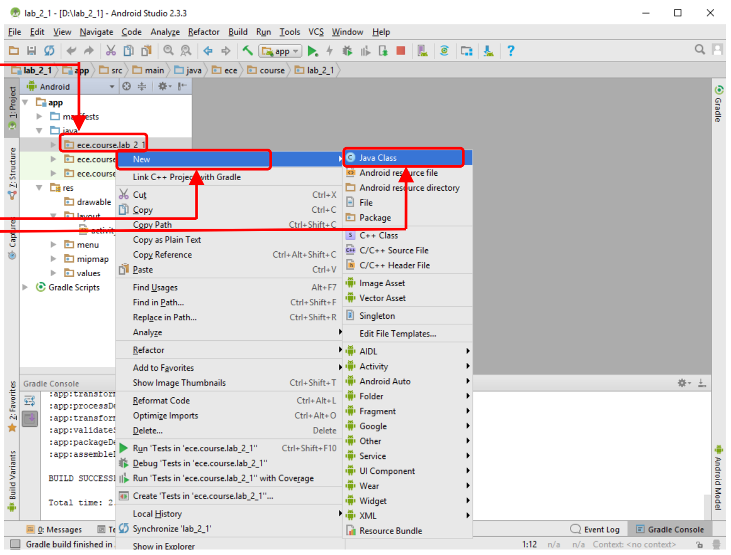
Step 2

Create custom surface view

01) Create new class

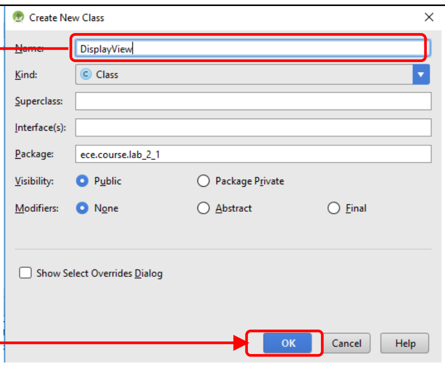
1) Right click <package name>

2) Click on "New" ➔ "Class"



3) Enter "name"
   (Use "DisplayView")

4) Click "OK"



Create New Class

Name:	DisplayView
Kind:	Class
Superclass:	
Interface(s):	
Package:	ece.course.lab_2_1
Visibility:	[✓ Public] Package Private
Modifiers:	[✓ None] Abstract Final

☐ Show Select Overrides Dialog

02) Set up class

1) Extends SurfaceView

Code Change:
public class DisplayView {
➔ public class DisplayView extends SurfaceView {

2) Add Constructor

Code:
```
public DisplayView(Context context) {
        super(context);
}
```

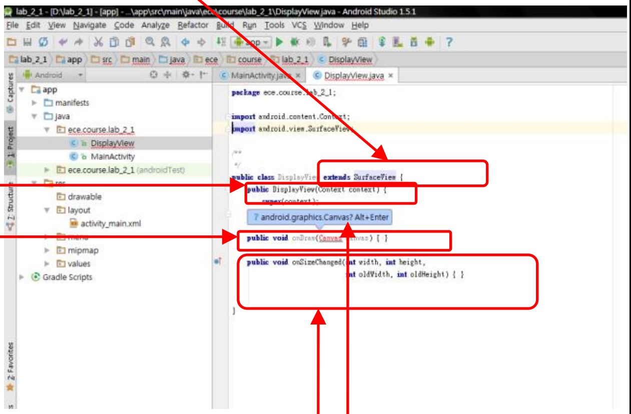
3) Add onDraw

Code:
```
public void onDraw(Canvas canvas) { }
```

4) Add onSizeChanged

Code:
```
Public void onSizeChanged(int width, int height,
        int oldWidth, int oldHeight) { }
```

*if these picture occurs, remember that click on it and select "import *"



lab_2_1 - [D:\lab_2_1] - [app] - ...\app\src\main\java\ece\course\lab_2_1\DisplayView.java - Android Studio 1.5.1

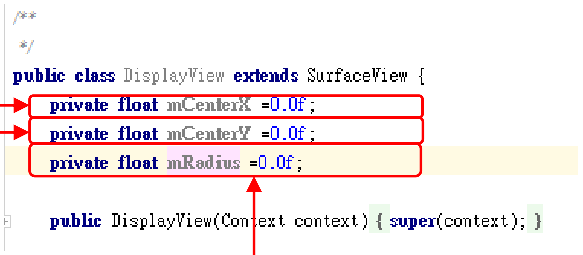
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
lab_2_1 app src main java ece course lab_2_1 DisplayView
Android
MainActivity.java
DisplayView.java
app
manifests
java
ece.course.lab_2_1
DisplayView
MainActivity
ece.course.lab_2_1 (androidTest)
res
drawable
layout
activity_main.xml
menu
mipmap
values
Gradle Scripts

```
package ece.course.lab_2_1;

import android.content.Context;
import android.view.SurfaceView;

/**
 */
public class DisplayView extends SurfaceView {
    public DisplayView(Context context) {
        super(context);
    }
    public void onDraw(Canvas canvas) { }

    public void onSizeChanged(int width, int height,
                              int oldWidth, int oldHeight) { }
}
```

? android.graphics.Canvas? Alt+Enter

03) Set up the variable relative the changing of View size

1) A variable for the x value of the center.
   (float mCenterX)
2) A variable for the y value of the center.
   (float mCenterY)
3) A variable for the radius of the circle.
   (float mRadius)

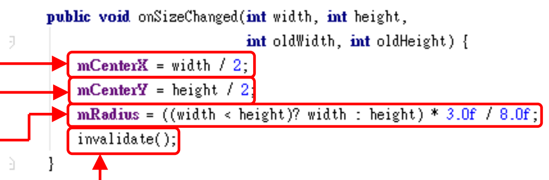


```
/**
 */
public class DisplayView extends SurfaceView {
    private float mCenterX  =0.0f;
    private float mCenterY  =0.0f;
    private float mRadius =0.0f;

    public DisplayView(Context context) { super(context); }
```

04) Write the code to update the size relatable variable when View size is change

@onSizeChanged

1) mCenterX should be half of width
2) mCenterY should be half of height
3) mRadius should be 3/8(You can change this to see what happen) of the shorter side
4) call "invalidate();" to update the View



```
public void onSizeChanged(int width, int height,
                          int oldWidth, int oldHeight) {
    mCenterX = width / 2;
    mCenterY = height / 2;
    mRadius = ((width < height)? width : height) * 3.0f / 8.0f;
    invalidate();
}
```

05) Draw the pictures

@onDraw

1) Set Blackground

Code:
canvas.drawColor(Color.BLACK);

2) Draw the board

Code:
Paint paint = new Paint();
paint.setColor(Color.LTGRAY);
canvas.drawCircle(mCenterX, mCenterY, mRadius, paint);

3) Draw red circle bubble

Code:
paint.setColor(Color.RED);
canvas.drawCircle(40.0f, 40.0f, 10.0f, paint);

4) Draw blue square bubble

Code:
paint.setColor(Color.BLUE);
canvas.drawRect(70.0f, 30.0f, 90.0f, 50.0f, paint);

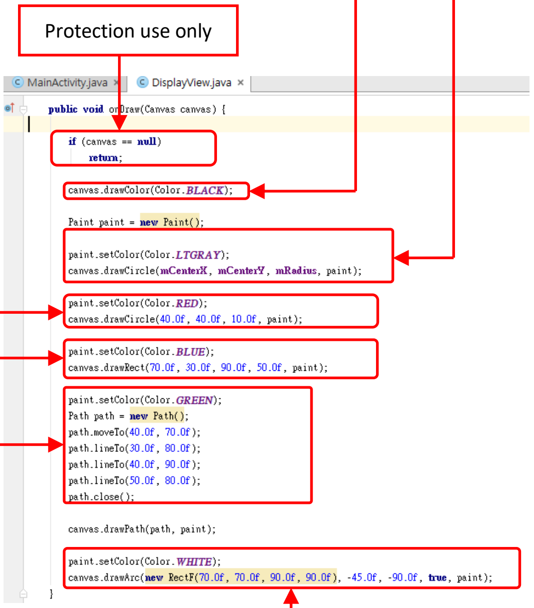
5) Draw green diamond

Code:
paint.setColor(Color.GREEN);
Path path = new Path();
path.moveTo(40.0f, 70.0f);
path.lineTo(30.0f, 80.0f);
path.lineTo(40.0f, 90.0f);
path.lineTo(50.0f, 80.0f);
path.close();

canvas.drawPath(path, paint);

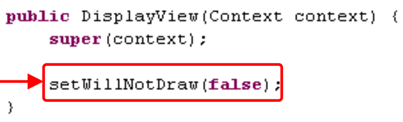
6) Draw white "arc" bubble

Code:
paint.setColor(Color.WHITE);
canvas.drawArc(new RectF(70.0f, 70.0f, 90.0f, 90.0f),
-45.0f, -90.0f, true, paint);



```
public void onDraw(Canvas canvas) {

    if (canvas == null)
        return;

    canvas.drawColor(Color.BLACK);

    Paint paint = new Paint();

    paint.setColor(Color.LTGRAY);
    canvas.drawCircle(mCenterX, mCenterY, mRadius, paint);

    paint.setColor(Color.RED);
    canvas.drawCircle(40.0f, 40.0f, 10.0f, paint);

    paint.setColor(Color.BLUE);
    canvas.drawRect(70.0f, 30.0f, 90.0f, 50.0f, paint);

    paint.setColor(Color.GREEN);
    Path path = new Path();
    path.moveTo(40.0f, 70.0f);
    path.lineTo(30.0f, 80.0f);
    path.lineTo(40.0f, 90.0f);
    path.lineTo(50.0f, 80.0f);
    path.close();

    canvas.drawPath(path, paint);

    paint.setColor(Color.WHITE);
    canvas.drawArc(new RectF(70.0f, 70.0f, 90.0f, 90.0f), -45.0f, -90.0f, true, paint);
}
```

06) Add "setWillNotDraw(false);" into the constructor in order to let the View redraw when calling "invalidate();"



```
public DisplayView(Context context) {
    super(context);
    setWillNotDraw(false);
}
```

Step 3

Update the main activity

01) Add the custom SurfaceView as a variable

Code:

private DisplayView mDisplayView;

02) Contructing a new SurfaceView

Code:

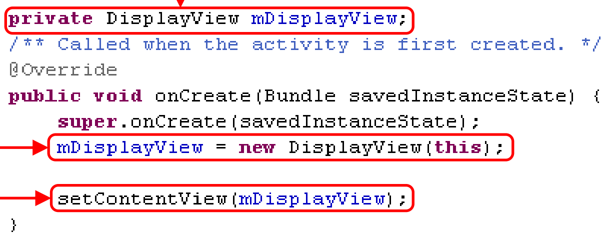
mDisplayView = new DisplayView(this);

03) Change the display to the custom SurfaceView

Code changed:

setContentView(R.layout.main);

➔ setContentView(mDisplayView);



```
private DisplayView mDisplayView;
/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    mDisplayView = new DisplayView(this);
    setContentView(mDisplayView);
}
```

Step 4

Test the apps by emulator

Task 2	Learn pop-up menu and control the bubble

Knowledge learn in this task:

Pop up menu	1) The menu pop up when pressing the menu button 2) It can be a multi-layer menu

Procedure of the task:

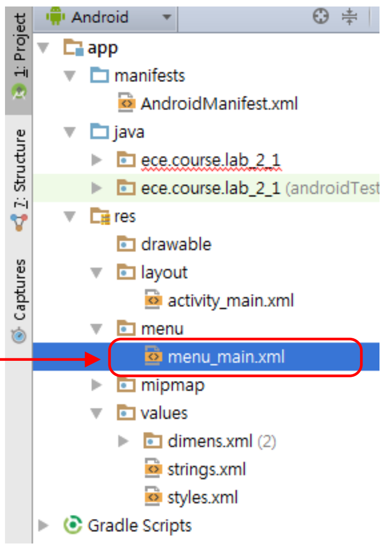
Step 1

Set up menu resources

(If menu folder is not there, create one as in step 01), otherwise, goto step 02)

01) Click in the "menu" folder in "res"

1) Double click and edit the menu_main.xml file



02) Edit the “menu_main.xml”

1) Add the xml header in the beginnings of file

xml header is use to tell the system that is a xml file

Code:

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

2) Add the “bubble” type item and the sub menu

Code for item:

```
<item android:id="@+id/menuPtrType"

        android:title="Type"

        android:orderInCategory="1">
</item>
```

id part

This is for setting the id using in the program

String displayed in the menu

Opening tag

Closing tag

Order in the menu

Code for sub-menu:

```
<item>
        <menu>
                <item />
        </menu>
</item>
```

`<item>` Opening tag of parent item

`<menu>` Opening tag of menu, just “<menu>”

`<item />` Items in sub menu

`</menu>` Closing tag of menu

`</item>` Closing tag of parent item

Item in Sub menu:

1.	id: menuPtrBall	title: Ball	order: 1
2.	id: menuPtrSquare	title: Square	order: 2
3.	id: menuPtrDiamond	title: Diamond	order: 3
4.	id: menuPtrArc	title: Arc	order: 4

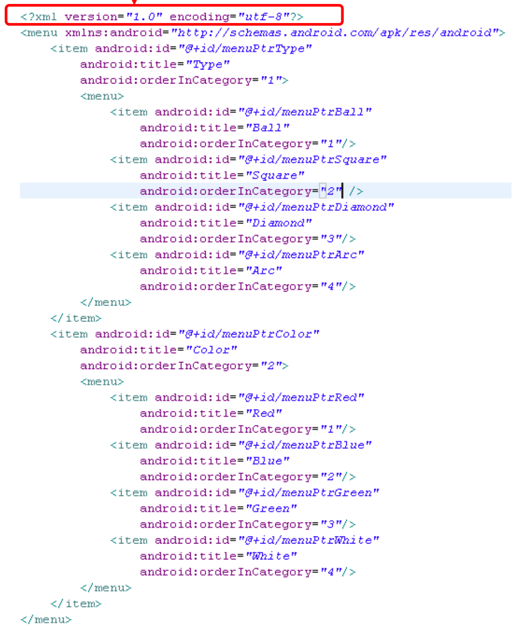
3) Add the “bubble” color item and the sub menu

Item in menu:

- id: menuPtrColor title: Color order: 2

Item in sub menu:

1.	id: menuPtrRed	title: Red	order: 1
2.	id: menuPtrBlue	title: Blue	order: 2
3.	id: menuPtrGreen	title: Green	order: 3
4.	id: menuPtrWhite	title: White	order: 4



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/menuPtrType"
        android:title="Type"
        android:orderInCategory="1">
        <menu>
            <item android:id="@+id/menuPtrBall"
                android:title="Ball"
                android:orderInCategory="1"/>
            <item android:id="@+id/menuPtrSquare"
                android:title="Square"
                android:orderInCategory="2" />
            <item android:id="@+id/menuPtrDiamond"
                android:title="Diamond"
                android:orderInCategory="3"/>
            <item android:id="@+id/menuPtrArc"
                android:title="Arc"
                android:orderInCategory="4"/>
        </menu>
    </item>
    <item android:id="@+id/menuPtrColor"
        android:title="Color"
        android:orderInCategory="2">
        <menu>
            <item android:id="@+id/menuPtrRed"
                android:title="Red"
                android:orderInCategory="1"/>
            <item android:id="@+id/menuPtrBlue"
                android:title="Blue"
                android:orderInCategory="2"/>
            <item android:id="@+id/menuPtrGreen"
                android:title="Green"
                android:orderInCategory="3"/>
            <item android:id="@+id/menuPtrWhite"
                android:title="White"
                android:orderInCategory="4"/>
        </menu>
    </item>
</menu>
```

Step 2

Set up Display View

01) Upadte the DisplayView so it is ready for the changes

1) Add constants for the type of "bubble"

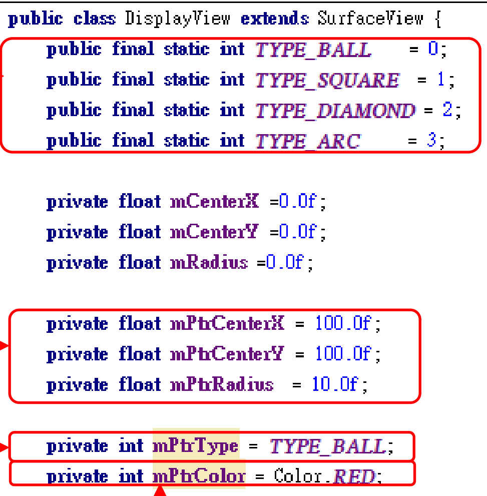
Code:
public final staic int TYPE_BALL = 0;

Constant type

Constant name

Constant value

Similary, add followings
TYPE_SQUARE = 1
TYPE_DIAMOND = 2
TYPE_ARC = 3



```
public class DisplayView extends SurfaceView {
    public final static int TYPE_BALL    = 0;
    public final static int TYPE_SQUARE  = 1;
    public final static int TYPE_DIAMOND = 2;
    public final static int TYPE_ARC     = 3;

    private float mCenterX =0.0f;
    private float mCenterY =0.0f;
    private float mRadius =0.0f;

    private float mPtrCenterX = 100.0f;
    private float mPtrCenterY = 100.0f;
    private float mPtrRadius  = 10.0f;

    private int mPtrType = TYPE_BALL;
    private int mPtrColor = Color.RED;
```

2) Add variables that the "Bubble" can change it position by diffenrent input.
   i. mPtrCenterX (float) to store the x position of "Bubble"
   ii. mPtrCenterY (float) to store the y position of "Bubble"
   iii. mPtrRadius (float) to store the size of "Bubble"

3) Add variable "mPtrType" (int type) to store "Bubble" type

4) Add variable "mPtrColor" (int type) to store "Bubble" color

02) Add functions in the DisplayView Class

```
public void setPtrColor(int color) {
        mPtrColor = color;
        invalidate();
}
```

```
public void setPtrType(int type) {
        mPtrType = type;
        invalidate();
}
```

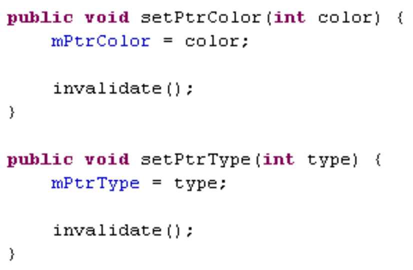
"Bubble" color control function

Update mPtrColor

Update Display

Update mPtrType

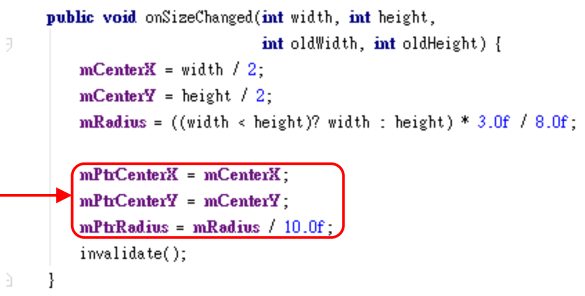
"Bubble" type control function



```
public void setPtrColor(int color) {
    mPtrColor = color;

    invalidate();
}

public void setPtrType(int type) {
    mPtrType = type;

    invalidate();
}
```

03) Add the followings code in the onSizeChanged in order to make the "Bubble" related parameter is relative to the size of view. (just before the "invalidate();")

Code:
mPtrCenterX = mCenterX;
mPtrCenterY = mCenterY;
mPtrRadius = mRadius / 10.0f;



```
public void onSizeChanged(int width, int height,
                          int oldWidth, int oldHeight) {
    mCenterX = width / 2;
    mCenterY = height / 2;
    mRadius = ((width < height)? width : height) * 3.0f / 8.0f;

    mPtrCenterX = mCenterX;
    mPtrCenterY = mCenterY;
    mPtrRadius = mRadius / 10.0f;
    invalidate();
}
```

04) Update the code of onDraw

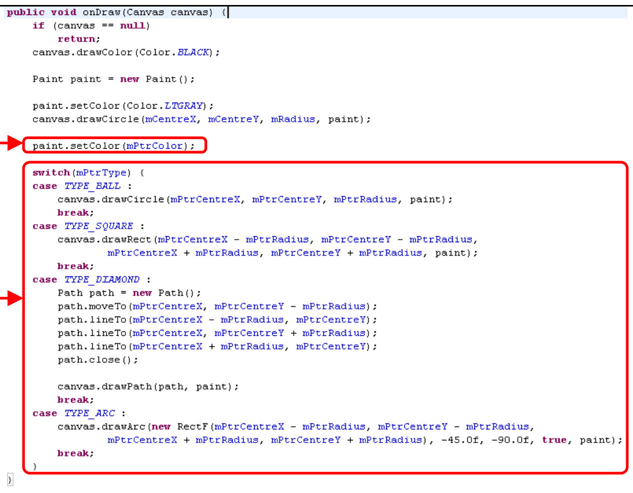
1) Delete the code of draw red circle, blue square, green diamond and white arc shape.

2) Add following code, so the color of "Bubble" is following the mPtrColor change

Code:

```
paint.setColor(mPtrColor);
```

3) Add following code, so the shape of "Bubble" is following the mPtrType change



```
public void onDraw(Canvas canvas) {
    if (canvas == null)
        return;
    canvas.drawColor(Color.BLACK);

    Paint paint = new Paint();

    paint.setColor(Color.LTGRAY);
    canvas.drawCircle(mCentreX, mCentreY, mRadius, paint);

    paint.setColor(mPtrColor);

    switch(mPtrType) {
    case TYPE_BALL :
        canvas.drawCircle(mPtrCentreX, mPtrCentreY, mPtrRadius, paint);
        break;
    case TYPE_SQUARE :
        canvas.drawRect(mPtrCentreX - mPtrRadius, mPtrCentreY - mPtrRadius,
                mPtrCentreX + mPtrRadius, mPtrCentreY + mPtrRadius, paint);
        break;
    case TYPE_DIAMOND :
        Path path = new Path();
        path.moveTo(mPtrCentreX, mPtrCentreY - mPtrRadius);
        path.lineTo(mPtrCentreX - mPtrRadius, mPtrCentreY);
        path.lineTo(mPtrCentreX, mPtrCentreY + mPtrRadius);
        path.lineTo(mPtrCentreX + mPtrRadius, mPtrCentreY);
        path.close();

        canvas.drawPath(path, paint);
        break;
    case TYPE_ARC :
        canvas.drawArc(new RectF(mPtrCentreX - mPtrRadius, mPtrCentreY - mPtrRadius,
                mPtrCentreX + mPtrRadius, mPtrCentreY + mPtrRadius), -45.0f, -90.0f, true, paint);
        break;
    }
}
```

Code:

```
switch(mPtrType) {
case TYPE_BALL:
        canvas.drawCircle(mPtrCenterX, mPtrCenterY, mPtrRadius, paint);
        break;
case TYPE_SQUARE :
        canvas.drawRect(mPtrCenterX - mPtrRadius, mPtrCenterY - mPtrRadius,
                mPtrCenterX + mPtrRadius, mPtrCenterY + mPtrRadius, paint);
        break;
case TYPE_DIAMOND :
        Path path = new Path();
        path.moveTo(mPtrCenterX, mPtrCenterY - mPtrRadius);
        path.lineTo(mPtrCenterX - mPtrRadius, mPtrCenterY);
        path.lineTo(mPtrCenterX, mPtrCenterY + mPtrRadius);
        path.lineTo(mPtrCenterX + mPtrRadius, mPtrCenterY);
        path.close();

        canvas.drawPath(path, paint);
        break;
case TYPE_ARC :
        canvas.drawArc(new RectF(mPtrCenterX - mPtrRadius, mPtrCenterY - mPtrRadius,
                mPtrCenterX + mPtrRadius, mPtrCenterY + mPtrRadius), -45.0f, -90.0f, true, paint);
        break;
}
```

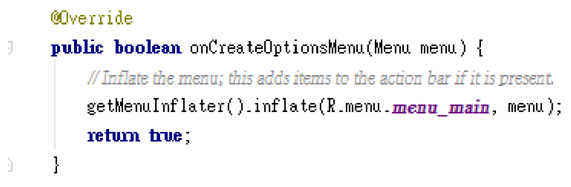
Step 3

Set up pop up menu

01) Create the pop up menu in the Activity

Code:

```
public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_main, menu);
        return true;
}
```



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

02) Set the menu item responses

Code: Return true or false value to indicate the event is handled or not

```
public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.menuPtrBall :
                mDisplayView.setPtrType(DisplayView.TYPE_BALL);
                return true;
        case R.id.menuPtrSquare :
                mDisplayView.setPtrType(DisplayView.TYPE_SQUARE);
                return true;
        case R.id.menuPtrDiamond :
                mDisplayView.setPtrType(DisplayView.TYPE_DIAMOND);
                return true;
        case R.id.menuPtrArc :
                mDisplayView.setPtrType(DisplayView.TYPE_ARC);
                return true;
        case R.id.menuPtrRed :
                mDisplayView.setPtrColor(Color.RED);
                return true;
        case R.id.menuPtrBlue :
                mDisplayView.setPtrColor(Color.BLUE);
                return true;
        case R.id.menuPtrGreen :
                mDisplayView.setPtrColor(Color.GREEN);
                return true;
        case R.id.menuPtrWhite :
                mDisplayView.setPtrColor(Color.WHITE);
                return true;
        }
        return false;
}
```

Use this to get which item is clicked

Step 4

Test the apps by emulator

Task 3	Learn sensor API and finish the Apps

Knowledge learn in this task:

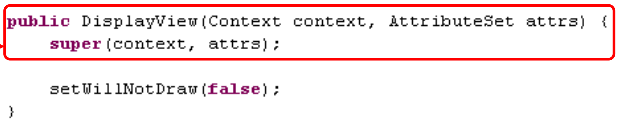
Sensor	1) We can use API to access the sensor data 2) Sensors included, tempature sensor, accelerometer, gyroscope.

Procedure of the task:

Step 1

Set up the layout

01) Update the constructor of the DisplayView, so it can use in the xml file



```
public DisplayView(Context context, AttributeSet attrs) {
    super(context, attrs);

    setWillNotDraw(false);
}
```

Codes changed:

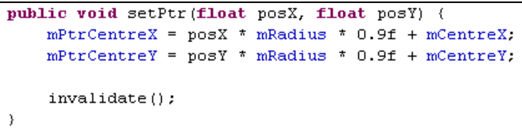
public DisplayView(Context context) {
➔public DisplayView(Context context, AttributeSet attrs) {

super(context);
➔super(context, attrs);

02) Add setPtr function to control the position of the "Bubble"

Code:

```
public void setPtr(float posX, float posY) {
		mPtrCenterX = posX * mRadius * 0.9f + mCenterX;
		mPtrCenterY = posY * mRadius * 0.9f + mCenterY;
		invalidate();
}
```



```
public void setPtr(float posX, float posY) {
    mPtrCentreX = posX * mRadius * 0.9f + mCentreX;
    mPtrCentreY = posY * mRadius * 0.9f + mCentreY;

    invalidate();
}
```

Using the formula to calculating the position of bubble
Range of input variable: 0.0f <= (posX, posY) <= 1.0f

03) Edit the layout

1) Open <project> ➔ "res" ➔ "layout" ➔ "main.xml"
2) Open the xml editor of the main.xml(can be refer to lab 1)
3) Change the code into the followings code

Code:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
		android:orientation="vertical"
		android:layout_width="fill_parent"
		android:layout_height="fill_parent">
		<LinearLayout android:layout_width="fill_parent"
				android:layout_height="wrap_content">
				<TextView android:text="X:"
						android:textSize="20sp"
						android:layout_width="wrap_content"
						android:layout_height="wrap_content" />
				<TextView android:id="@+id/tvValueX"
						android:textSize="20sp"
						android:gravity="right"
						android:layout_width="fill_parent"
						android:layout_height="wrap_content" />
		</LinearLayout>
		<LinearLayout android:layout_width="fill_parent"
				android:layout_height="wrap_content">
				<TextView android:text="Y:"
						android:textSize="20sp"
						android:layout_width="wrap_content"
						android:layout_height="wrap_content" />
				<TextView android:id="@+id/tvValueY"
						android:textSize="20sp"
						android:gravity="right"
						android:layout_width="fill_parent"
						android:layout_height="wrap_content" />
		</LinearLayout>
		<LinearLayout android:layout_width="fill_parent"
				android:layout_height="wrap_content">
				<TextView android:text="Z:"
						android:textSize="20sp"
						android:layout_width="wrap_content"
						android:layout_height="wrap_content" />
				<TextView android:id="@+id/tvValueZ"
						android:textSize="20sp"
						android:gravity="right"
						android:layout_width="fill_parent"
						android:layout_height="wrap_content" />
		</LinearLayout>
		<ece.course.lab_2.DisplayView
				android:id="@+id/mDisplayView"
				android:layout_width="fill_parent"
				android:layout_height="fill_parent" />
</LinearLayout>
```

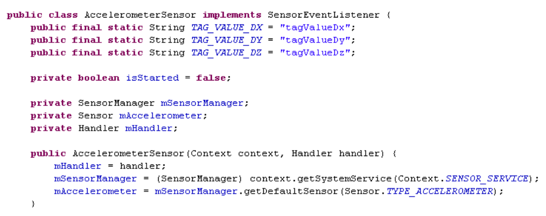
This part is for Value X

Calling Custom View

Step 2

Create the accelerometer class

01) Create new class "AccelerometerSensor"



```
public class AccelerometerSensor implements SensorEventListener {
    public final static String TAG_VALUE_DX = "tagValueDx";
    public final static String TAG_VALUE_DY = "tagValueDy";
    public final static String TAG_VALUE_DZ = "tagValueDz";

    private boolean isStarted = false;

    private SensorManager mSensorManager;
    private Sensor mAccelerometer;
    private Handler mHandler;

    public AccelerometerSensor(Context context, Handler handler) {
        mHandler = handler;
        mSensorManager = (SensorManager) context.getSystemService(Context.SENSOR_SERVICE);
        mAccelerometer = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    }
```

02) Implements SensorEventListener

Code Changed:

public class AccelerometerSensor {

➔ public class AccelerometerSensor implements SensorEventListener {

03) Create some useful constants

1) TAG_VALUE_DX = "tagValueDx" (String type)
2) TAG_VALUE_DY = "tagValueDy" (String type)
3) TAG_VALUE_DZ = "tagValueDz" (String type)

04) Set up the useful variable

Code:

```
private boolean isStarted = false;
private SensorManager mSensorManager;
private Sensor mAccelerometer;
private Handler mHandler;
```

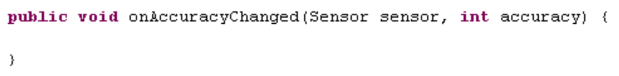
05) Create the constructor

Code:

Use this to return data back to the parent

```
public AccelerometerSensor(Context context, Handler handler) {
        mHandler = handler;
        mSensorManager = (SensorManager) context.getSystemService(Context.SENSOR_SERVICE);
        mAccelerometer = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
}
```

06) Create the function onAccuracyChanged



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

}
```

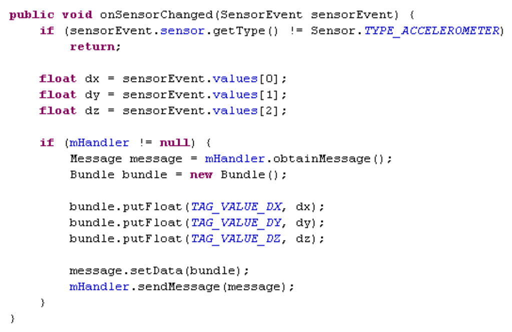
Code:

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

07) Create the function onSensorChanged

Code:

```
public void onSensorChanged(SensorEvent sensorEvent) {
        if (sensorEvent.sensor.getType() !=
        Sensor.TYPE_ACCELEROMETER)
                return;

        float dx = sensorEvent.values[0];
        float dy = sensorEvent.values[1];
        float dz = sensorEvent.values[2];

        if (mHandler != null) {
                Message message = mHandler.obtainMessage();
                Bundle bundle = new Bundle();

                bundle.putFloat(TAG_VALUE_DX, dx);
                bundle.putFloat(TAG_VALUE_DY, dy);
                bundle.putFloat(TAG_VALUE_DZ, dz);

                message.setData(bundle);
                mHandler.sendMessage(message);
        }
}
```

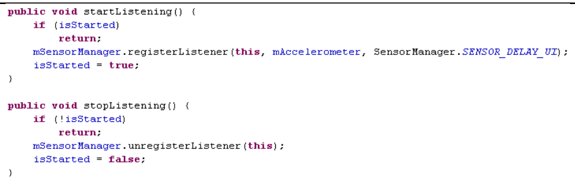


```
public void onSensorChanged(SensorEvent sensorEvent) {
    if (sensorEvent.sensor.getType() != Sensor.TYPE_ACCELEROMETER)
        return;

    float dx = sensorEvent.values[0];
    float dy = sensorEvent.values[1];
    float dz = sensorEvent.values[2];

    if (mHandler != null) {
        Message message = mHandler.obtainMessage();
        Bundle bundle = new Bundle();

        bundle.putFloat(TAG_VALUE_DX, dx);
        bundle.putFloat(TAG_VALUE_DY, dy);
        bundle.putFloat(TAG_VALUE_DZ, dz);

        message.setData(bundle);
        mHandler.sendMessage(message);
    }
}
```

Returning data back to the parent

08) Add two functions

1) startListening as a trigger for start
2) stopListening as a trigger for stop



```
public void startListening() {
    if (isStarted)
        return;
    mSensorManager.registerListener(this, mAccelerometer, SensorManager.SENSOR_DELAY_UI);
    isStarted = true;
}

public void stopListening() {
    if (!isStarted)
        return;
    mSensorManager.unregisterListener(this);
    isStarted = false;
}
```

Code:

```
public void startListening() {
        if (isStarted)
                return;
        mSensorManager.registerListener(this, mAccelerometer, SensorManager.SENSOR_DELAY_UI);
        isStarted = true;
}

public void stopListening() {
        if (!isStarted)
                return;
        mSensorManager.unregisterListener(this);
        isStarted = false;
}
```

Step 3

Update the main Activity

01) Add the useful constant and variable

Constant:

MAX_GRAVITY = 9.82f

Variable:

AccelerometerSensor mAccelerometerSensor;

02) Delete the mDisplayView Constructor

Delete:

mDisplayView = new DisplayView(this);

03) Set up the display

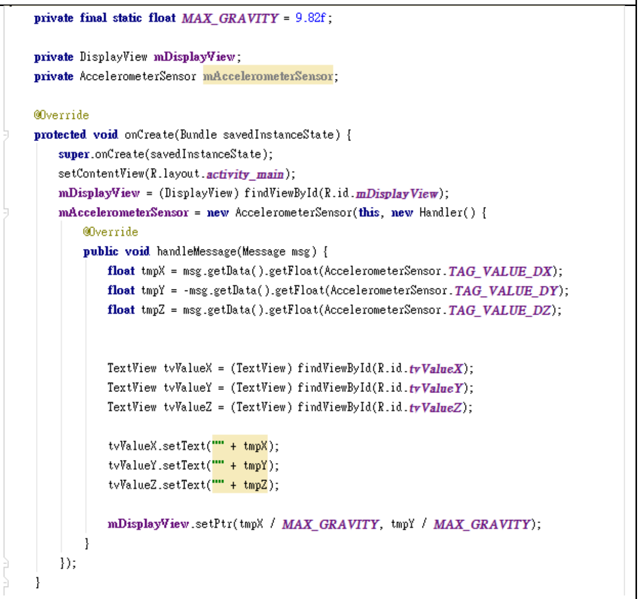
Code Changed:

setContentView(mDisplayView);

➔ setContentView(R.layout.main);

Code Add:

mDisplayView = (DisplayView)
findViewById(R.id.mDisplayView);



```
private final static float MAX_GRAVITY = 9.82f;

private DisplayView mDisplayView;
private AccelerometerSensor mAccelerometerSensor;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    mDisplayView = (DisplayView) findViewById(R.id.mDisplayView);
    mAccelerometerSensor = new AccelerometerSensor(this, new Handler() {
        @Override
        public void handleMessage(Message msg) {
            float tmpX = msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DX);
            float tmpY = -msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DY);
            float tmpZ = msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DZ);


            TextView tvValueX = (TextView) findViewById(R.id.tvValueX);
            TextView tvValueY = (TextView) findViewById(R.id.tvValueY);
            TextView tvValueZ = (TextView) findViewById(R.id.tvValueZ);

            tvValueX.setText("" + tmpX);
            tvValueY.setText("" + tmpY);
            tvValueZ.setText("" + tmpZ);

            mDisplayView.setPtr(tmpX / MAX_GRAVITY, tmpY / MAX_GRAVITY);
        }
    });
}
```

04) Set up the Accelerometer

For the error of the handler, select import Handler (android.os)

Code:

Get back the data in the message

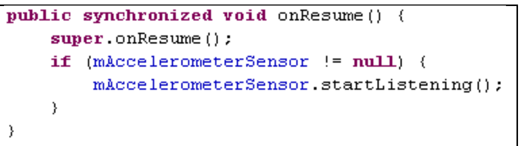
```
mAccelerometerSensor = new AccelerometerSensor(this, new Handler() {
        public void handleMessage(Message msg) {
                float tmpX = msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DX);
                float tmpY = -msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DY);
                float tmpZ = msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DZ);

                TextView tvValueX = (TextView) findViewById(R.id.tvValueX);
                TextView tvValueY = (TextView) findViewById(R.id.tvValueY);
                TextView tvValueZ = (TextView) findViewById(R.id.tvValueZ);

                tvValueX.setText("" + tmpX);
                tvValueY.setText("" + tmpY);
                tvValueZ.setText("" + tmpZ);

                mDisplayView.setPtr(tmpX / MAX_GRAVITY, tmpY / MAX_GRAVITY);
        }
});
```

05) Create the onResume function to handling start cases

Code: When the Apps started or resume from pause, it will call onResume

```
public synchronized void onResume() {
        super.onResume();
        if (mAccelerometerSensor != null) {
                mAccelerometerSensor.startListening();
        }
}
```

We should start listen the accelerometer, when it is started or resume



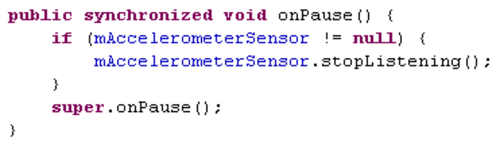
```
public synchronized void onResume() {
    super.onResume();
    if (mAccelerometerSensor != null) {
        mAccelerometerSensor.startListening();
    }
}
```

06) Create the onPause function to handling the stop cases

Code: When the Apps stopped or pause, it will call onPause

```
public synchronized void onPause() {
        if (mAccelerometerSensor != null) {
                mAccelerometerSensor.stopListening();
        }
        super.onPause();
}
```

In order to have good protection for the phone, we should stop listening to accelerometer when it is pause / stop

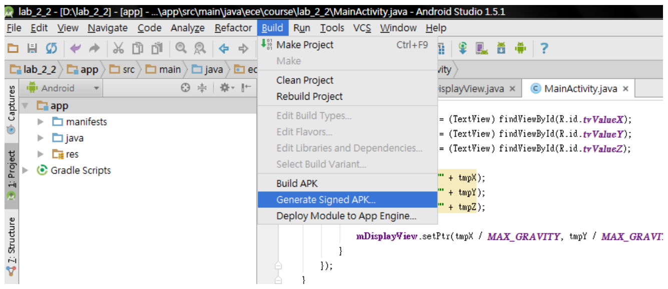


```
public synchronized void onPause() {
    if (mAccelerometerSensor != null) {
        mAccelerometerSensor.stopListening();
    }
    super.onPause();
}
```

Stpe 4

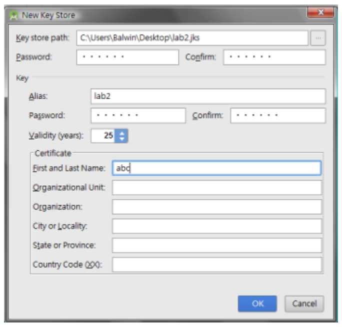
Export the apps

01) Highlight the <project folder>
02) Click "Build APK"
03) Click Generate Signed APK


lab_2_2 - [D:\lab_2_2] - [app] - ...\app\src\main\java\ece\course\lab_2_2\MainActivity.java - Android Studio 1.5.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Make Project Ctrl+F9
Make
Clean Project
Rebuild Project
Edit Build Types...
Edit Flavors...
Edit Libraries and Dependencies...
Select Build Variant...
Build APK
Generate Signed APK...
Deploy Module to App Engine...
lab_2_2 app src main java
Android
app
manifests
java
res
Gradle Scripts
Captures
1: Project
7: Structure
isplayView.java
MainActivity.java

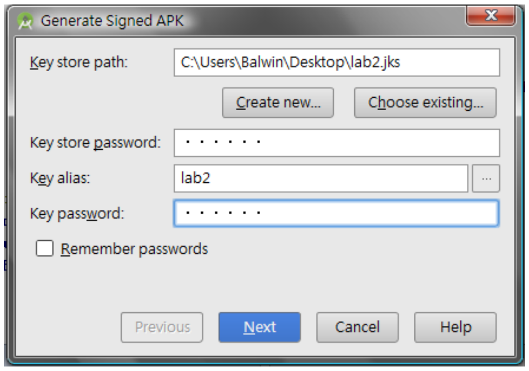
```
= (TextView) findViewById(R.id.tvValueX);
= (TextView) findViewById(R.id.tvValueY);
= (TextView) findViewById(R.id.tvValueZ);

"" + tmpX);
"" + tmpY);
"" + tmpZ);

            mDisplayView.setPtr(tmpX / MAX_GRAVITY, tmpY / MAX_GRAVI
        }
    });
}
```

04) You can select “Use existing keystore” or “Create new keystore”
- 1) For “Use existing keystore”
  - 1)Use “Browse…” to select a existing keystore or enter the path and file name in “Location”
  - 2)Enter the “Password” of keystore
- 2) For “Create new keystore”
  - 1)Use “Browse…” to select the creating path and name or enter the path and file name in “Location”
  - 2)Enter a “Password” for your keystore
  - 3)Re-enter the “Password” in “Confirm”
  - 4)Jump to Line 08 in this Step



New Key Store

Key store path: C:\Users\Balwin\Desktop\lab2.jks
Password: • • • • • • • Confirm: • • • • • • •
Key
Alias: lab2
Password: • • • • • • • Confirm: • • • • • • •
Validity (years): 25
Certificate
First and Last Name: abc
Organizational Unit:
Organization:
City or Locality:
State or Province:
Country Code (XX):



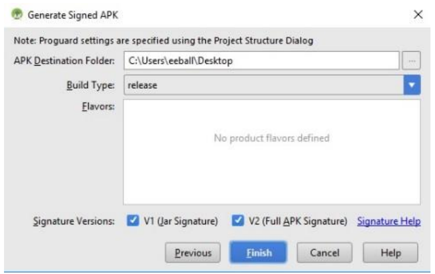
Generate Signed APK

Key store path: C:\Users\Balwin\Desktop\lab2.jks
Key store password: • • • • • • •
Key alias: lab2
Key password: • • • • • • •
Remember passwords

05) You can select “Use existing key” or “Create new key” for your apps
- 1) For “Use existing key”
  - 1)Select the key in “alias” of the key
  - 2)Enter the “Password” of the key
  - 3)Press “Next”
  - 4)Jump to Line 15 in this Step
- 2) For “Create new key”
  - 1)Press “Next”

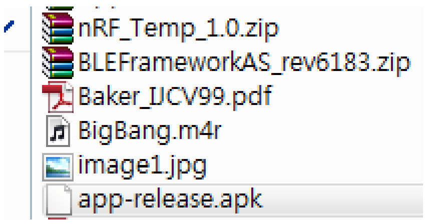
06) Enter the “alias” for indicating the key
07) Enter a “Password for the key
08) Re-enter the “Password” in “Confirm”
09) Enter the field, “Validity (years)”, recommend more than “25”
10) Enter the “First and Last Name”
11) For the other field, you can choose to “fill-in”
12) Press “OK”

13) Select the path of the APK, click both V1&V2 Box
14) Press “Finish”
15) The apk filename is app-release.apk by default



Generate Signed APK

Note: Proguard settings are specified using the Project Structure Dialog
APK Destination Folder: C:\Users\eeball\Desktop
Build Type: release
Flavors: No product flavors defined
Signature Versions: [✓ V1 (Jar Signature)] [✓ V2 (Full APK Signature)] Signature Help



nRF_Temp_1.0.zip
BLEFrameworkAS_rev6183.zip
Baker_IJCV99.pdf
BigBang.m4r
image1.jpg
app-release.apk

After testing the program in the Android phone, you will notices that althrough everything are working, still some minor problems happened.
For example
1) The layout will change if phone is in a landscape mode
2) The accelerometer value is fluctuating all the time
3) The moniter will shut down automatically after a while.

So there is a special task for the apps (Lab). In the task, we will do
1) Fix the orientation
2) Add threshold for the changing of accelerometer to prevent the fluctuating problem
3) Add a wakeLock to make the phone on all the time.

Task 4	Clean up the minor problems.

Knowledge learn in this task:

screenOrientation	1) Which is a setting under "Activity", in the AndroidManifest.xml 2) Use to fix the orientation of a "Activity"
wakeLock	1) Which can make the apps never go into sleep mode 2) When using it, we need to ask for permission.

Procedure of the task:

Step 1
Update the main activity

01) Adding the variable and constant for "Threshold"
Code:
```
private final static float MAX_GRAVITY = 9.82f
private float mX = -100.0f;
private float mY = -100.0f;
private float mZ = -100.0f;
```

02) Changing the code in the function handleMessage in the function onCreate for "Threshold"
Code changed to:
```
float tmpX = msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DX);
float tmpY = -msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DY);
float tmpZ = msg.getData().getFloat(AccelerometerSensor.TAG_VALUE_DZ);
if (tmpX - mX > THRESHOLD || tmpX - mX < -THRESHOLD ||
        tmpY - mY > THRESHOLD || tmpY - mY < -THRESHOLD ||
        tmpZ - mZ > THRESHOLD || tmpZ - mZ < -THRESHOLD) {

        mX = tmpX; mY = tmpY; mZ = tmpZ;
        TextView tvValueX = (TextView) findViewById(R.id.tvValueX);
        TextView tvValueY = (TextView) findViewById(R.id.tvValueY);
        TextView tvValueZ = (TextView) findViewById(R.id.tvValueZ);
        tvValueX.setText("" + mX);
        tvValueY.setText("" + mY);
        tvValueZ.setText("" + mZ);
        mDisplayView.setPtr(mX / MAX_GRAVITY, mY / MAX_GRAVITY);
}
```

If the changes bigger than threshold, we can think that it is not the noise

03) Adding the variable for “wakeLock”

Code:

```
private PowerManager mPowerManager;
private WakeLock mWakeLock;
```

04) Adding the code in the **onCreate** function for the “wakeLock”

Code:

```
mPowerManager = (PowerManager) getSystemService(POWER_SERVICE);
mWakeLock = mPowerManager.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK,
                getClass().getName());
```

05) Adding the following code in onResume()

Code:

```
mWakeLock.acquire();
```

06) Adding the following code in onPause()

Code:

```
mWakeLock.release();
```

Step 2

Update the “AndroidManifest.xml”

01) Open the “AndroidManifest.xml”

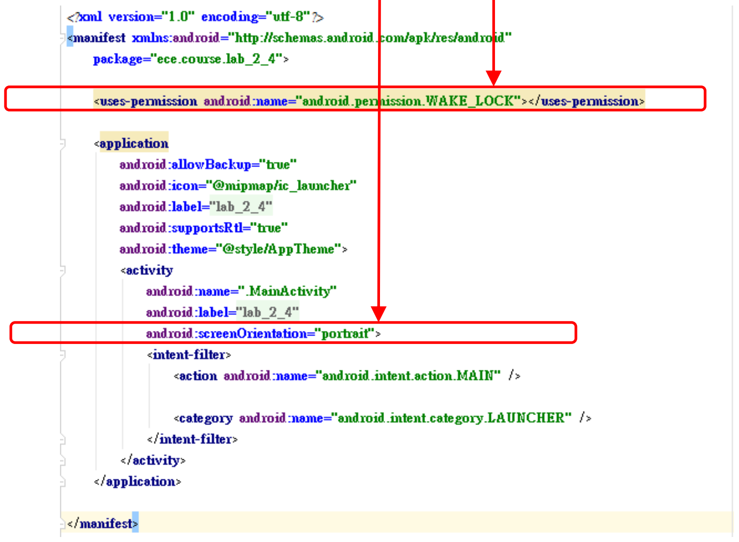
02) Open the xml editor by clicking “AndroidManifest.xml” on the bottom of eclipse

03) Adding the permission

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

04) Adding fix orientation

Add: android:screenOrientation="portrait"



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ece.course.lab_2_4">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="lab_2_4"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="lab_2_4"
            android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Step 3

Export the apps and test it And demonstrates to TA / IA