Workshop

1. Create a simple Environment Sensors (*Page 1 – 6*)
   - Pressure Sensor
   - Ambient Temperature Sensor
   - Light Sensor
   - Relative Humidity Sensor

2. Create a simple Position Sensors (*Page 7 – 8*)
   - Proximity Sensor
   - Geomagnetic Field Sensor

3. Create a simple Motion Sensors (*Page 9 – 13*)
   - Rotation Vector Sensor
   - Acceleration Sensor

4. By using GPS service, find out your location by emulator and your mobile *(Page 14 – 19*)
   - Obtain the GPS Location
   - Send Mock Location to AVD

5. Store your data in shared preferences (*Page 20 – 22*).

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**", ensure that the id for textView1 is existed.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.mysensor.MainActivity" >

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

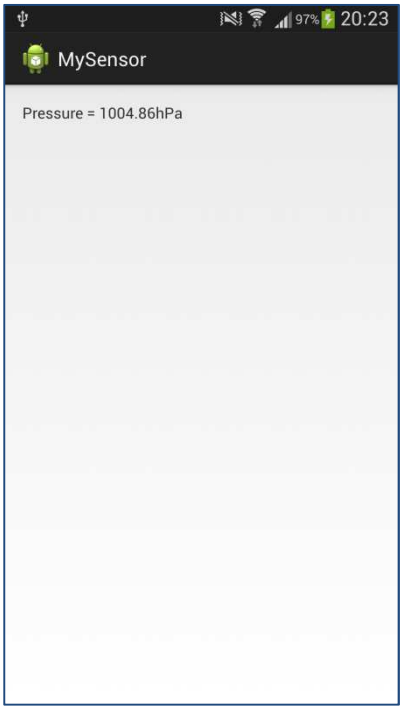
3. Modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysensor;


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

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

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

        // Display the value on screen
        mTextView.setText("Pressure = " + event.values[0] + "hPa");
    }

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

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

    @Override
    public 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 does not workable in emulator. Moreover, these sensors are hardware-based and are available only if a device manufacturer has built them into a device*).

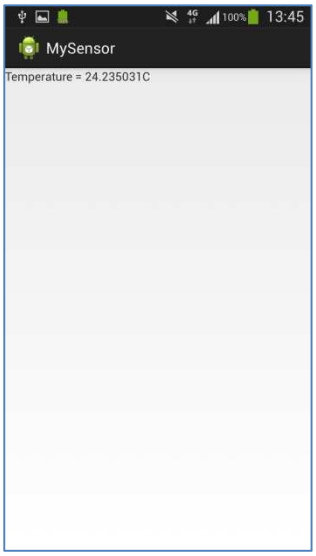


1.2 Ambient Temperature Sensor

1. Open the previous project and modify the sensor type for the sensor manager to **TYPE_AMBIENT_TEMPERATURE** in source file "**MainActivity.java**". You can also update the text for displaying temperature.

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

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

    // Display the value on screen
    mTextView.setText("Temperature = " + event.values[0] + "C");
}
```

2. Save and execute the app, you should able to obtain the ambient air temperature in your mobile.

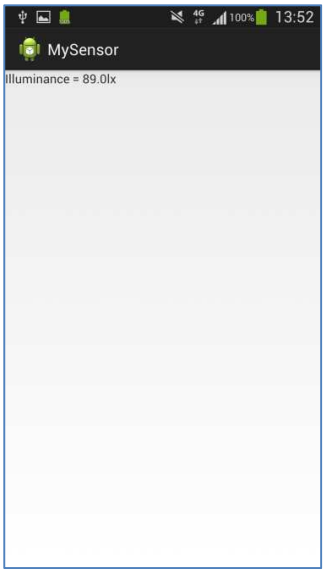


1.3 Light Sensor

1. Open the previous project and modify the sensor type for the sensor manager to **TYPE_LIGHT** in source file "**MainActivity.java**". You can also update the text for displaying illuminance.

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

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

    // Display the value on screen
    mTextView.setText("Illuminance = " + event.values[0] + "lx");
}
```

2. Save and execute the app, you should able to obtain the illuminance in your mobile.



1.4 Relative Humidity Sensor

1. Open the previous project and modify the sensor type for the sensor manager to **TYPE_RELATIVE_HUMIDITY** in source file "**MainActivity.java**". You can also update the text for displaying relative humidity.

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

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

    // Display the value on screen
    mTextView.setText("Relative humidity = " + event.values[0] + "%");
}
```

2. Save and execute the app, you should able to obtain the ambient relative humidity in your mobile.

2. Position Sensors

2.1 Proximity Sensor

1. Open the previous project and modify the sensor type for the sensor manager to **TYPE_PROXIMITY** in source file "**MainActivity.java**". You can also update the text for displaying distance.

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

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

    // Display the value on screen
    mTextView.setText("Distance = " + event.values[0] + "cm");
}
```

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.

2.2 Geomagnetic Field Sensor

1. Open the previous project and modify the sensor type for the sensor manager to **TYPE_MAGNETIC_FIELD** in source file "**MainActivity.java**". You can also update the text for displaying geomagnetic field strength.

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

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

    // Display the value on screen
    mTextView.setText("x axis = " + event.values[0] + "µT\n" +
                      "y axis = " + event.values[1] + "µT\n" +
                      "z axis = " + event.values[2] + "µT");
}
```

2. Save and execute the app, you should able to obtain the geomagnetic field strength along the x, y, z axis from the mobile.

3. Motion Sensors

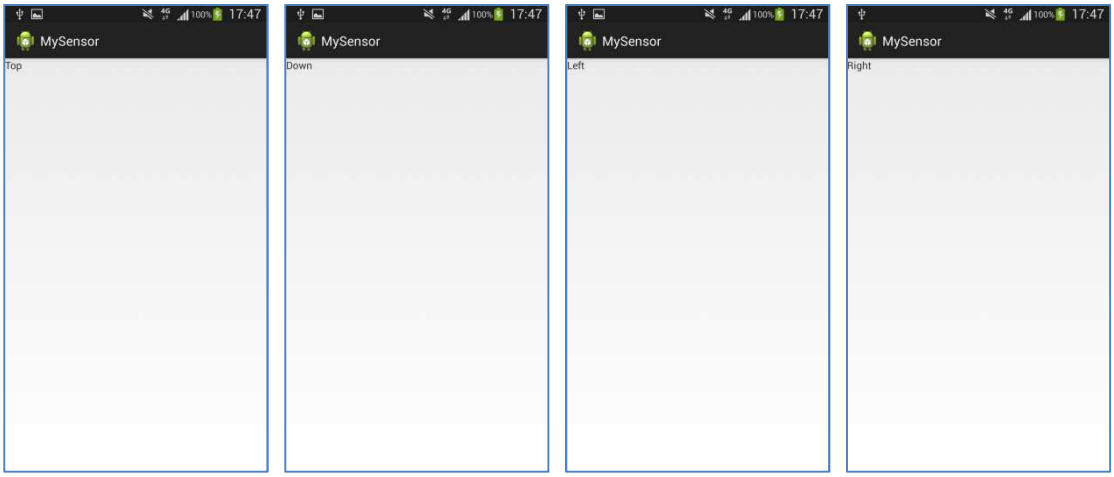
3.1 Rotation Vector Sensor

1. Open the previous project and modify the sensor type to **TYPE_ROTATION_VECTOR** and the sensor change event as follow in source file "**MainActivity.java**".

```
package com.example.mysensor;

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

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

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

        // Display the value on screen
```

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

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

        @Override
        public 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.



3.2 Acceleration Sensor

1. Modify the sensor type to **TYPE_ACCELEROMETER** and the sensor change event as follow in source file "**MainActivity.java**".

```
package com.example.mysensor;

import android.app.Activity;
import android.os.Bundle;
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;
    float x, init_x;
    long lastUpdateTime = 0;

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

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

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

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

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

        @Override
        public 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;
        }
}
```

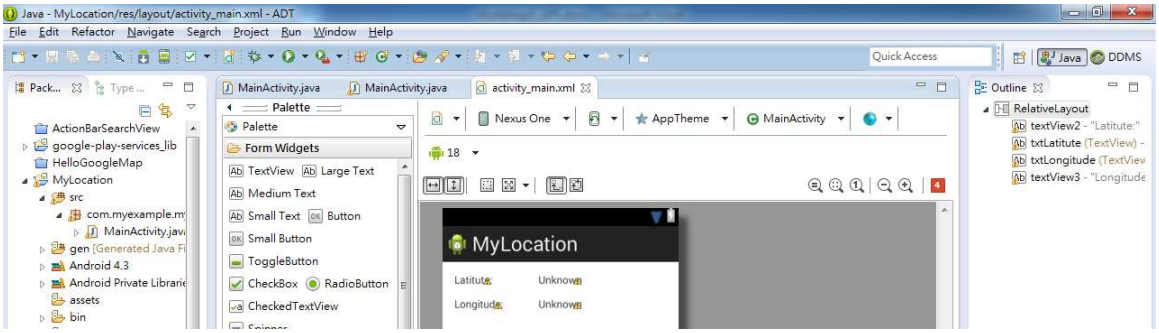
2. Save and execute the app, you should able to obtain the direction (left or right) by shaking your mobile.

4. Location Services

4.1 GPS Location

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

2. Put four text views into the layout as follow:



3. The XML code for the layout look like:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Latitude:" />
    <TextView
        android:id="@+id/txtLatitude"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/textView2"
        android:layout_marginLeft="19dp"
        android:layout_toRightOf="@+id/textView2"
        android:text="Unknown" />
    <TextView
        android:id="@+id/textView1"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView2"
        android:layout_below="@+id/textView2"
        android:layout_marginTop="18dp"
        android:text="Longitude:" />
    <TextView
        android:id="@+id/txtLongitude"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/textView1"
        android:layout_alignBottom="@+id/textView1"
        android:layout_alignLeft="@+id/txtLatitude"
        android:text="Unknown" />
</RelativeLayout>
```

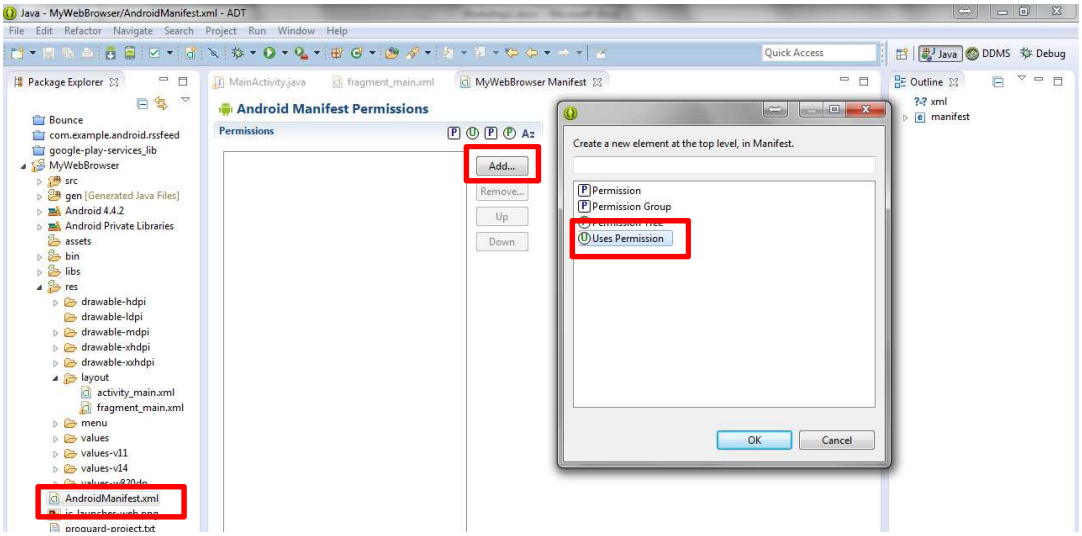
4. Modify the source file "**MainActivity.java**" as follow:

```
package com.example.mylocation;

import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.content.Context;
import android.widget.TextView;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;

public class MainActivity extends Activity implements LocationListener {
    private double latitude, longitude;
    private TextView txtLatitude;
    private TextView txtLongitude;
    private LocationManager locationManager;
    private String provider;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

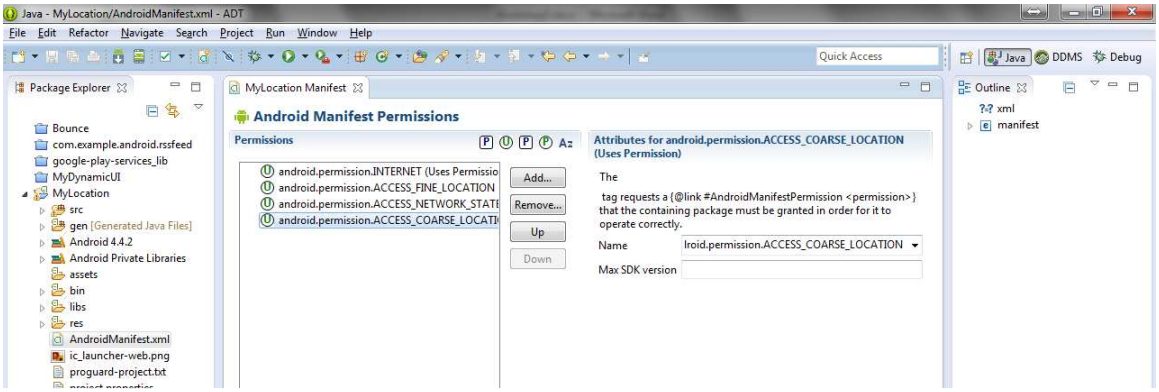
```
        setContentView(R.layout.activity_main);

        txtLatitude = (TextView) findViewById(R.id.txtLatitude);
        txtLongitude = (TextView) findViewById(R.id.txtLongitude);

        // Get the location manager
       locationManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);

       // Define the criteria how to select the location provider
       Criteria criteria = new Criteria();
       provider = locationManager.getBestProvider(criteria, false);
       Location location = locationManager.getLastKnownLocation(provider);

       // Initialize the location fields
       if (location != null) {
           // Provider selected
            onLocationChanged(location);
       } else {
            txtLatitude.setText("Unknown");
            txtLongitude.setText("Unknown");
       }
   }

   @Override
   public void onLocationChanged(Location location) {
        latitude = (double) (location.getLatitude());
        longitude = (double) (location.getLongitude());
        txtLatitude.setText(String.valueOf(latitude));
        txtLongitude.setText(String.valueOf(longitude));
   }

   @Override
   public void onStatusChanged(String provider, int status, Bundle extras) {
   }

   @Override
   public void onProviderEnabled(String provider) {
   }
```

```
    @Override
    public void onProviderDisabled(String provider) {
    }

    @Override
    public void onResume() {
        // Request updates at startup
        super.onResume();
        locationManager.requestLocationUpdates(provider, 400, 1, this);
    }

    @Override
    public void onPause() {
        // Remove the location listener updates when Activity is paused
        super.onPause();
        locationManager.removeUpdates(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;
    }
}
```

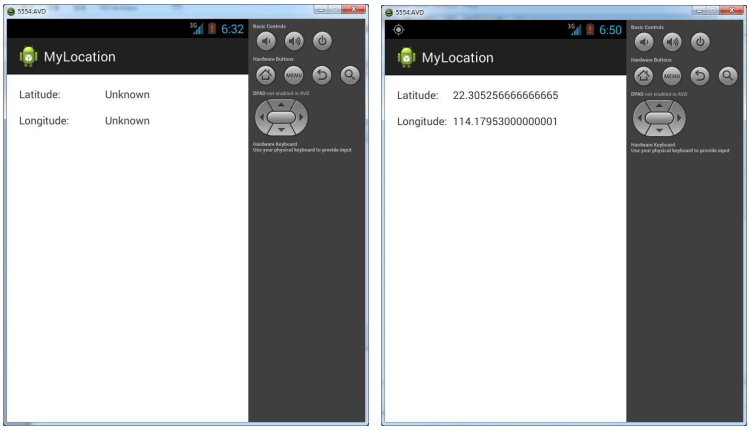
5. Open the manifest file "**AndroidManifest.xml**", and press **[Add]** in the "**Permission**" tab. Then select "**Uses Permission**" and press **[OK]**.



6. Add the following user permission in "**AndroidManifest.xml**":
   - android.permission.INTERNET
   - android.permission.ACCESS_FINE_LOCATION
   - android.permission.ACCESS_NETWORK_STATE
   - android.permission.ACCESS_COARSE_LOCATION

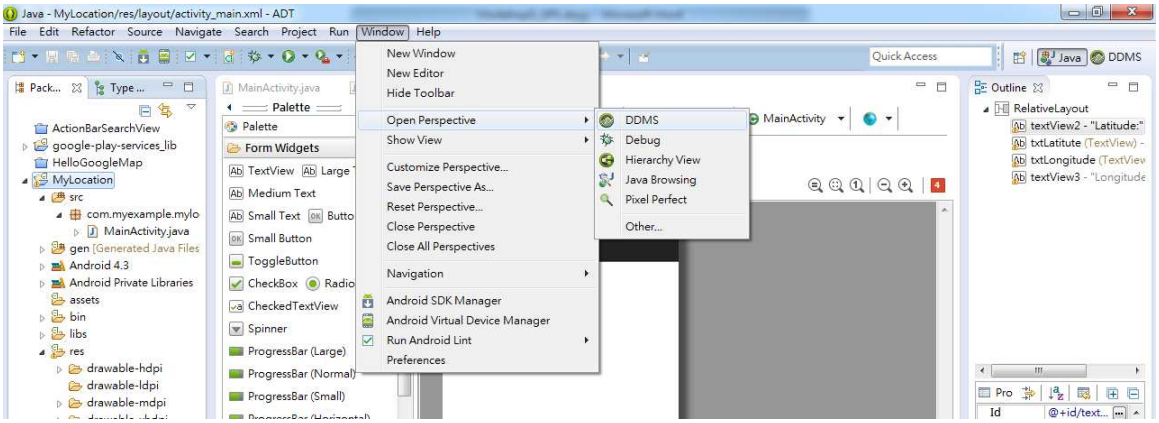


7. Save and execute the app, can you see the location information in emulator? Install this app into real mobile, and turn on GPS, can you see the location information correctly?

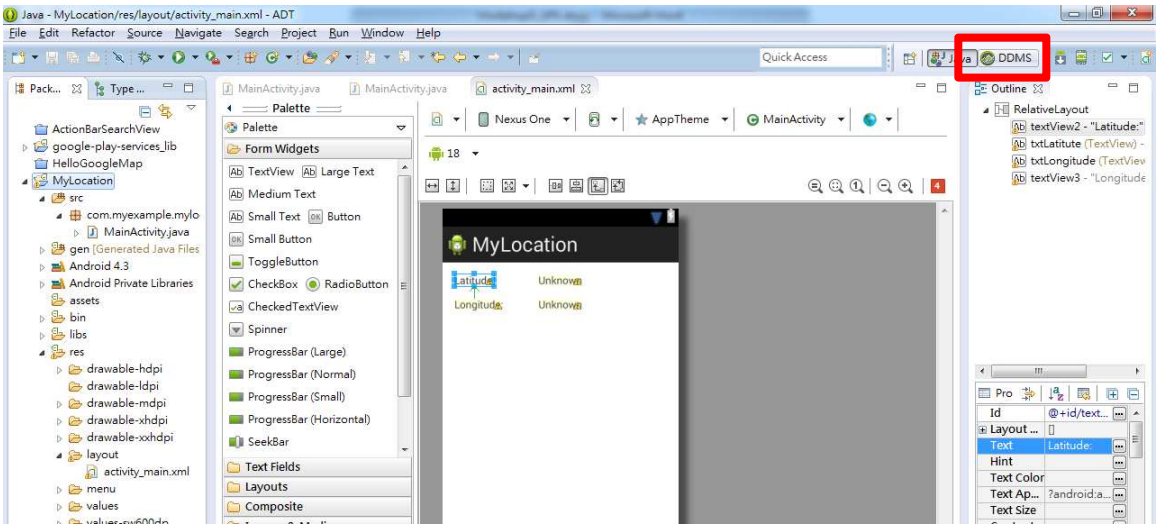


4.2 Mock Location in AVD

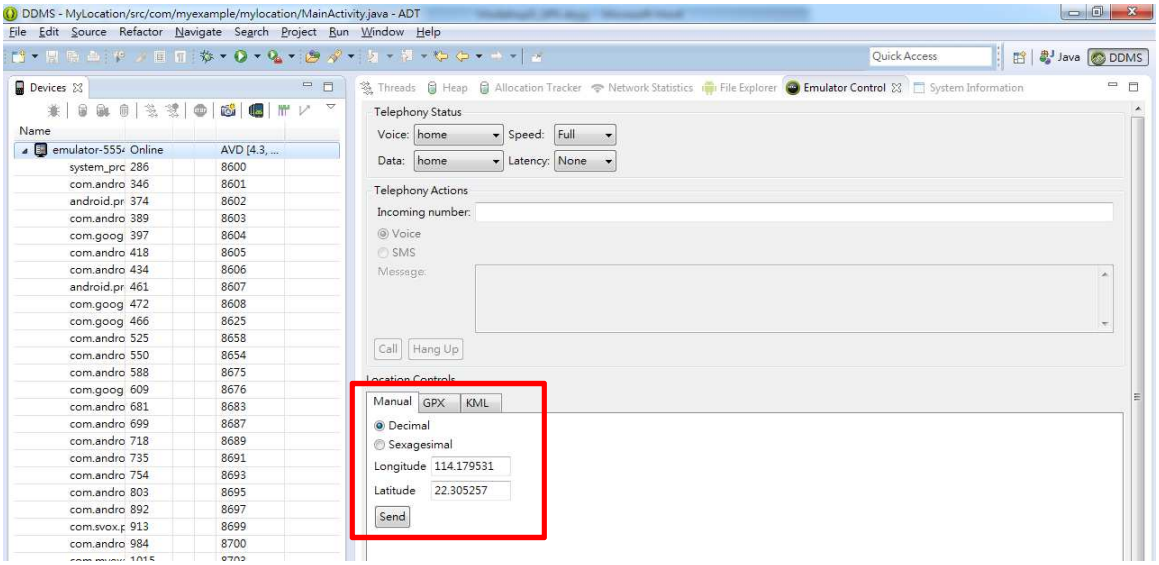
1. Select **Windows ➔ Open Perspective ➔ DDMS**.



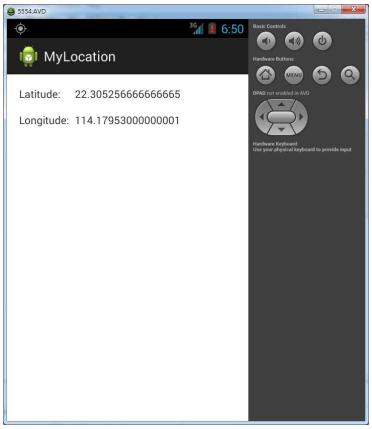
2. Select the **DDMS** tab.



3. Input "**114.179531**" and "**22.305257**" as the longitude and latitude, and then press "**Send**".



4. Switch to the app again, the location should be display correctly.

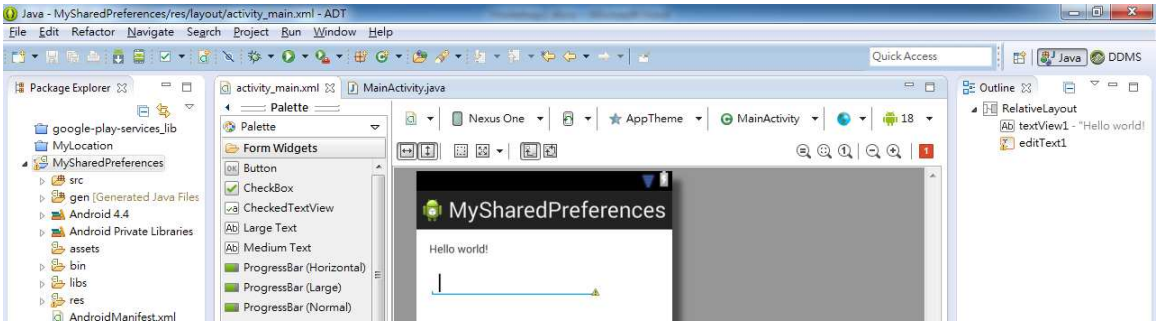


5. Shared Preferences

5.1 Shared Preferences

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

2. Drag a Plain Text to the layout.



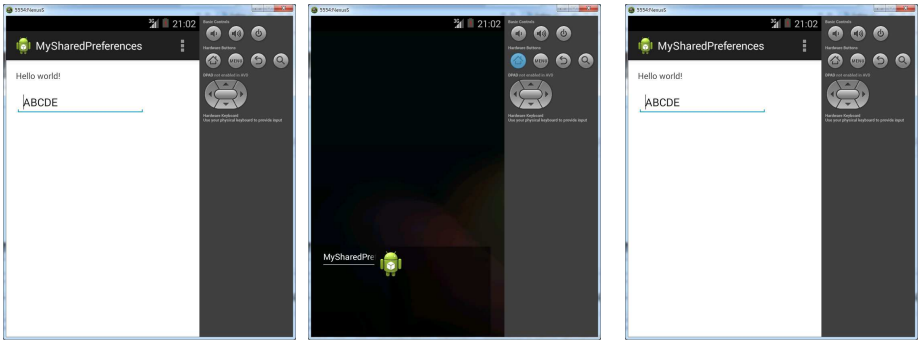
3. Modify the source file "**MainActivity.java**" as follow:

```
package com.example.mysharedpreferences;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.content.SharedPreferences;
import android.content.Context;
import android.widget.EditText;

public class MainActivity extends Activity {
    private final String PREF_FILE =
            "com.example.mysharedpreferences.PREF_FILE";
    private final String PREF_KEY = "MY_SAVE_DATA";
    private EditText mEditText;
    private String mDataString;

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

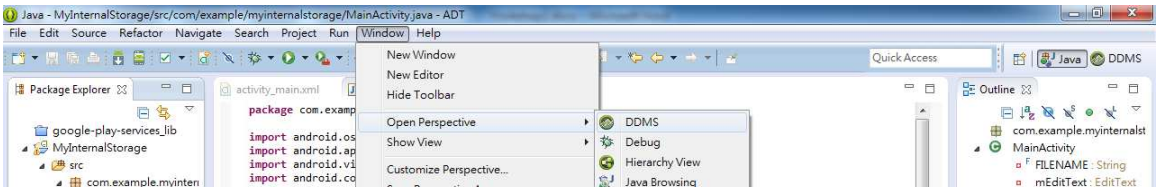
```
        // Restore the data from SharedPreferences
        ReadSharedPreferences();

        // Finds a view that was identified by the ID attribute from the XML
        mEditText = (EditText) findViewById(R.id.editText1);

        // Display the data on EditText
        mEditText.setText(mDataString);
    }

    @Override
    public void onPause() {
        super.onPause();

        // Read the user input text and save
        mDataString = mEditText.getText().toString();

        // Save the data to SharedPreferences
        SaveSharedPreferences();
    }

    public void ReadSharedPreferences() {
        // Retrieve and hold the contents of the preferences file
        SharedPreferences preferences =
            getSharedPreferences(PREF_FILE, Context.MODE_PRIVATE);

        // Retrieve the String from the preferences
        mDataString = preferences.getString(PREF_KEY, "");
    }

    public void SaveSharedPreferences() {
        // Retrieve and hold the contents of the preferences file
        SharedPreferences preferences =
        getSharedPreferences(PREF_FILE, Context.MODE_PRIVATE);

        // Create a new Editor for these preferences
        SharedPreferences.Editor editor = preferences.edit();
```

```
            // Set the String value in the preferences editor
            editor.putString(PREF_KEY, mDataString);

            // Commit your preferences changes
            editor.commit();
        }

        @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. Input something in text field, and close the AVD. Then execute the app again, can you restore your input value?



5. Select **Windows ➔ Open Perspective ➔ DDMS**



6. You can find your sharedpreference under **data➔data➔ [your package] ➔ shared_prefs**.