Android Basic Development Guide

Software Engineering – 1 (COMP-3350)

Hadi Hemmati
12/16/2015

Section 1

Setting up Environment

Fig 1. Start by Downloading Android Studio®

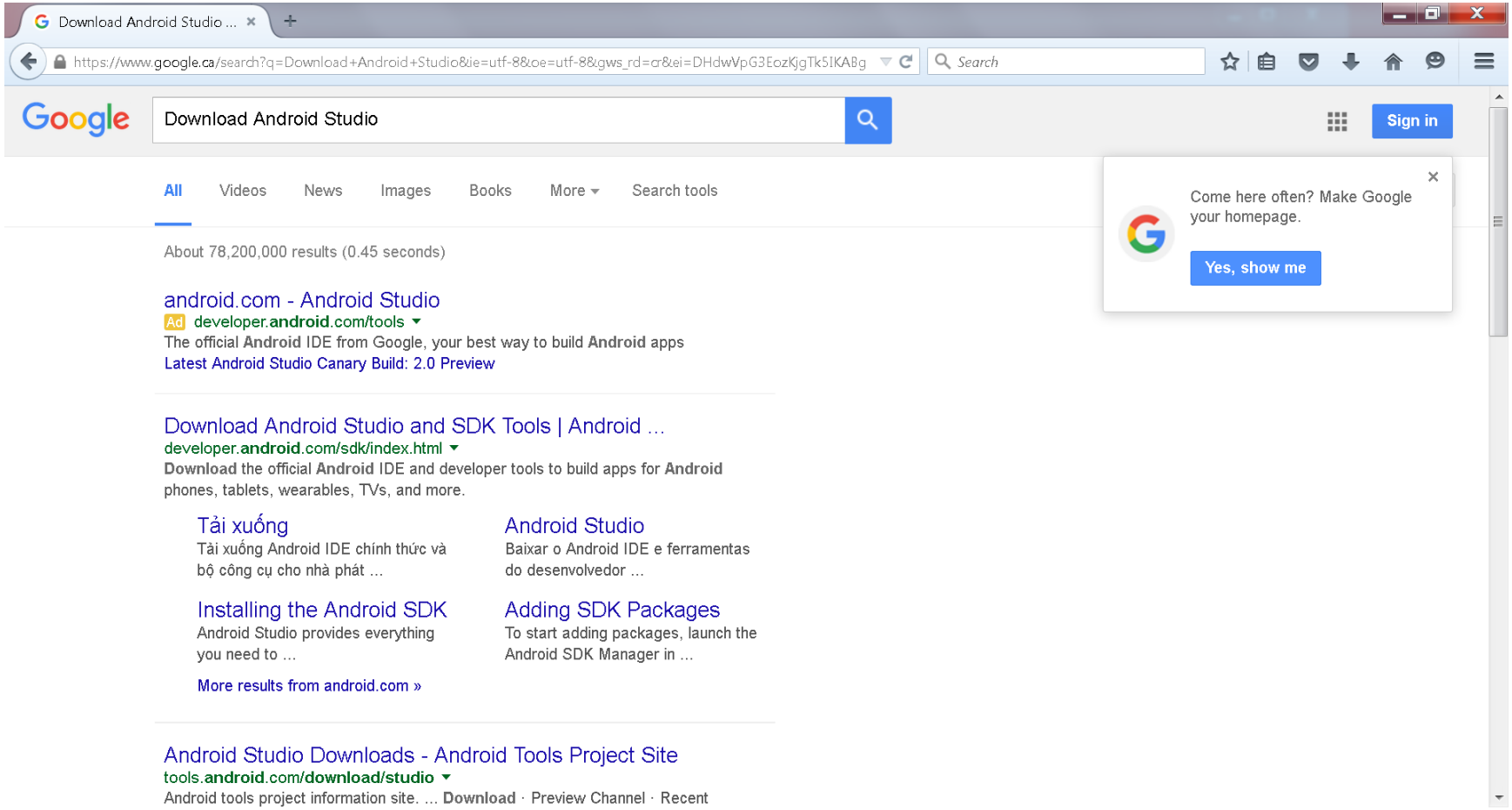


Fig 2. Downloading Android Studio®

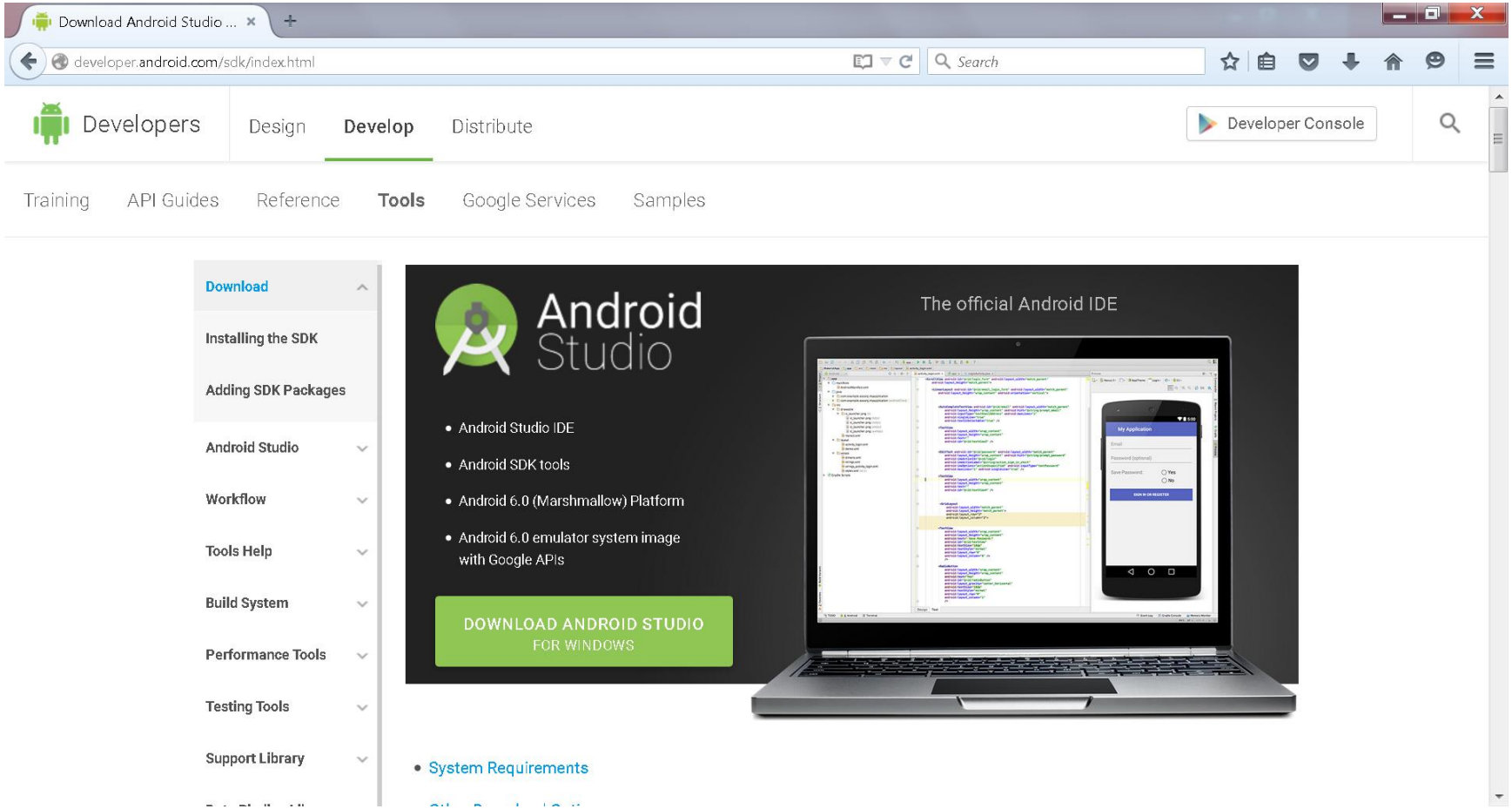


Fig 3. Downloading Android Studio®

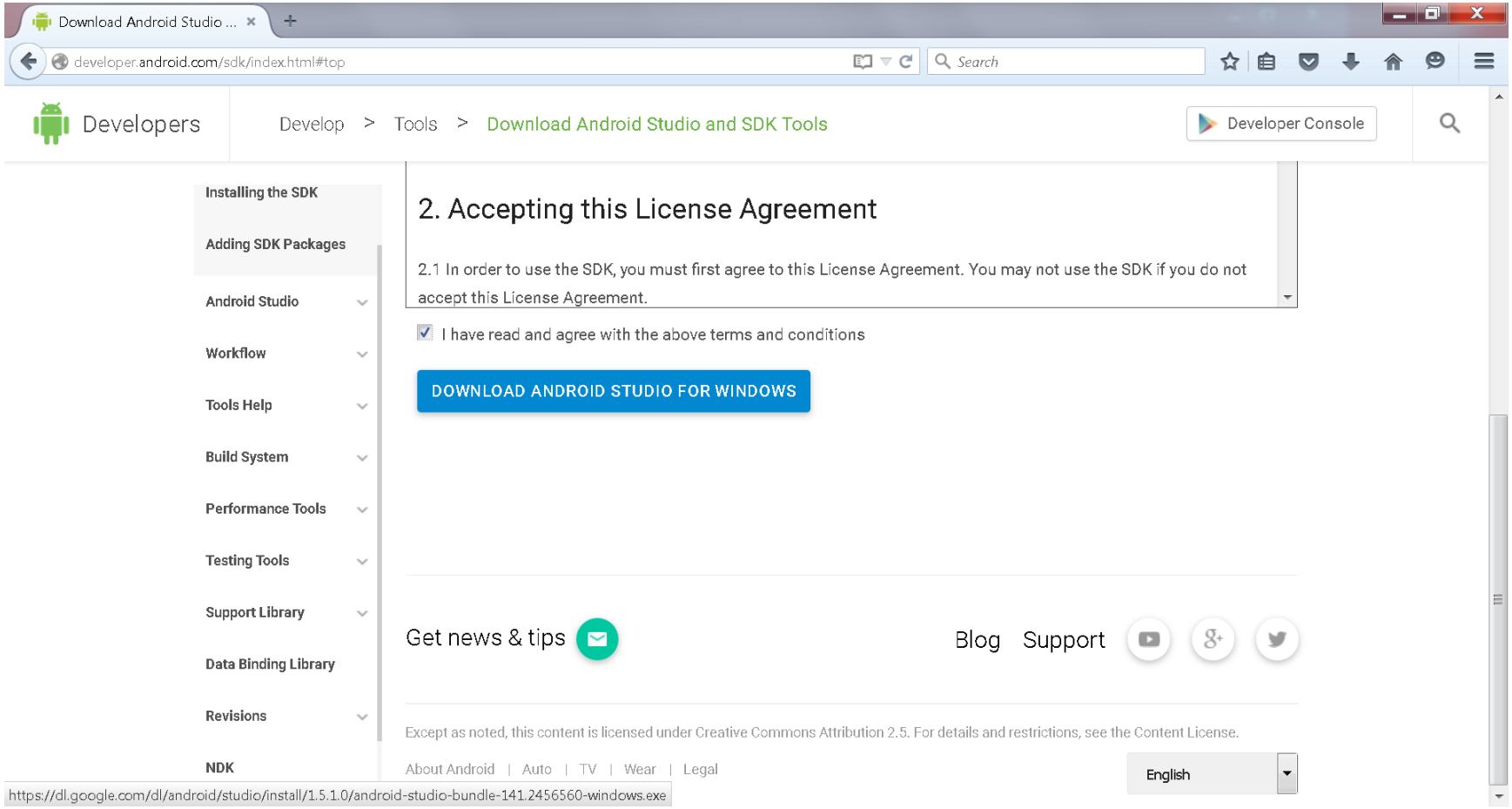


Fig 4. Run Android Studio® setup as Administrator

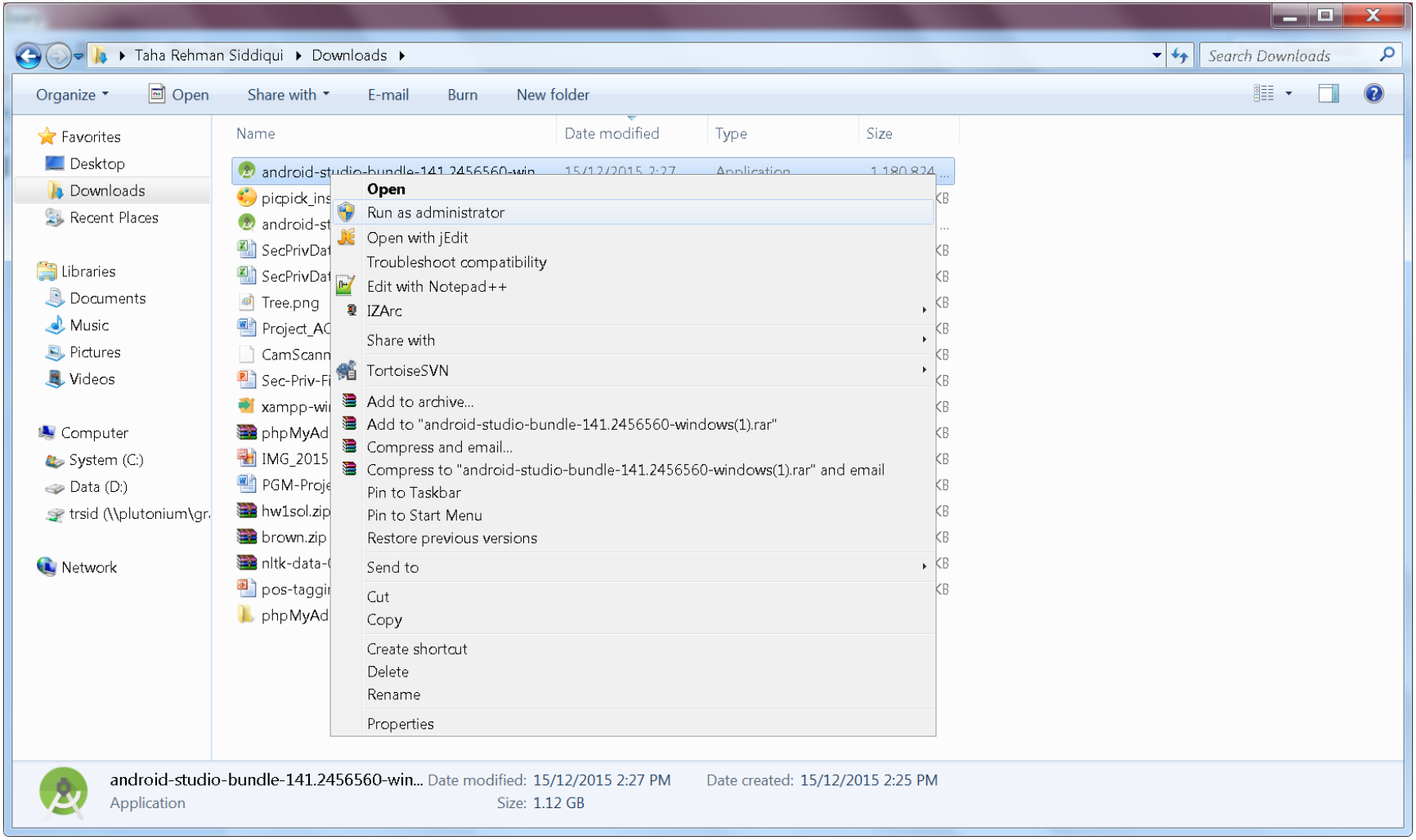


Fig 5. Set up Android Studio® (Keep clicking Next)

Fig 6. Set up Android Studio® Hardware Accelerator (The allowed Custom range depends upon the memory in your system. Allocating one quarter of the total memory is recommended)

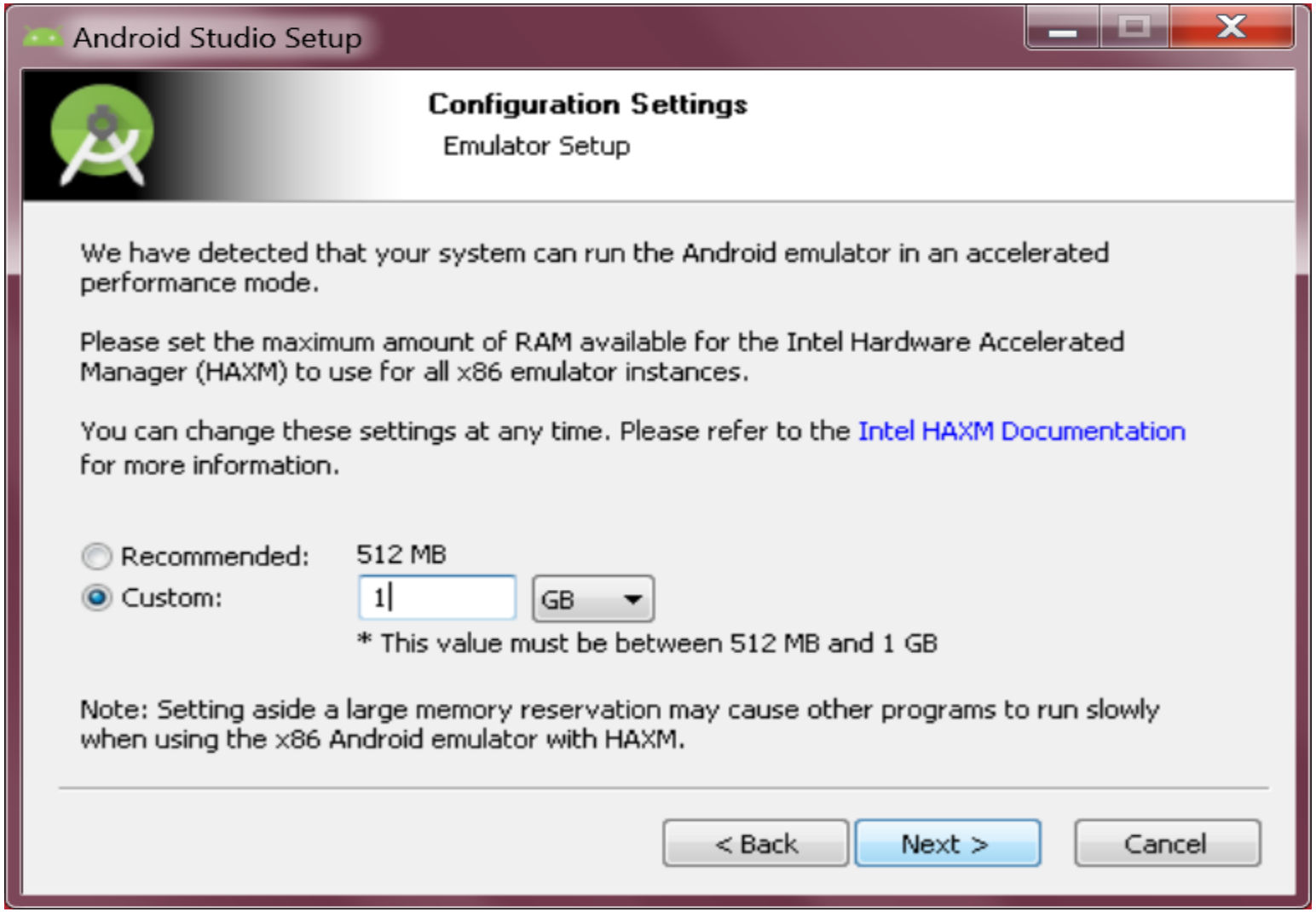


Fig 7. Start Android Studio® (Keep clicking Next)

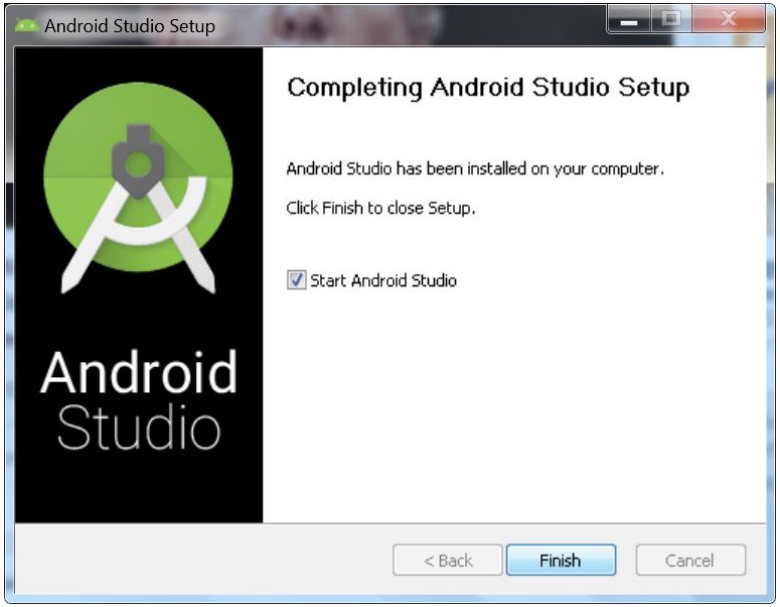


Fig 8. Select Do not Import Settings

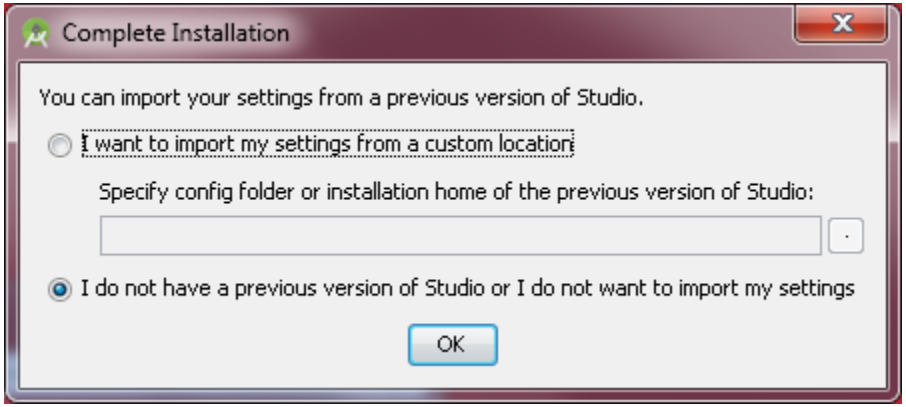


Fig 9. Add exception to firewall (as it will attempt to download a lot of resources itself)

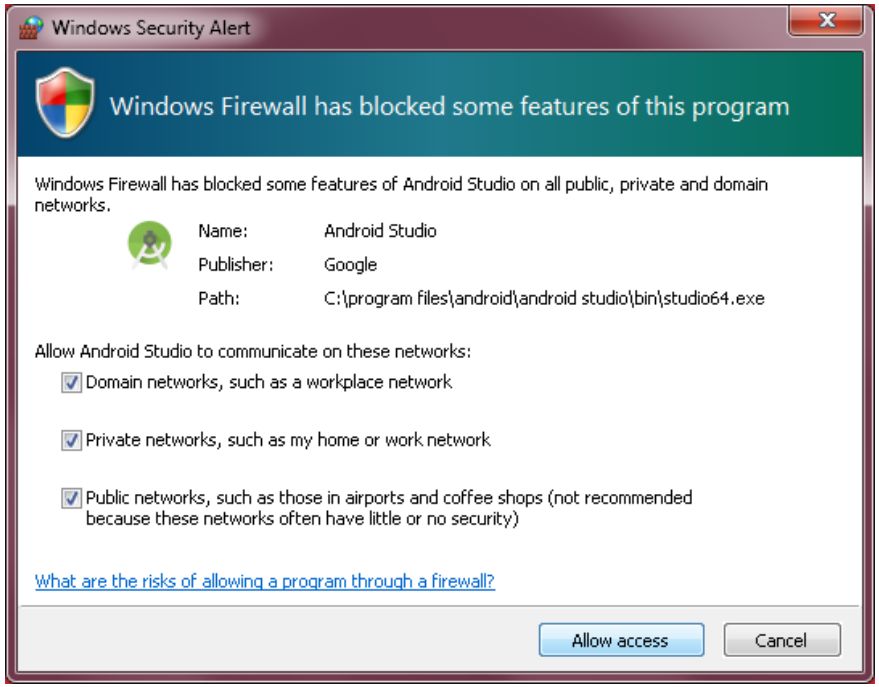


Fig 10. Auto-Downloading latest SDK

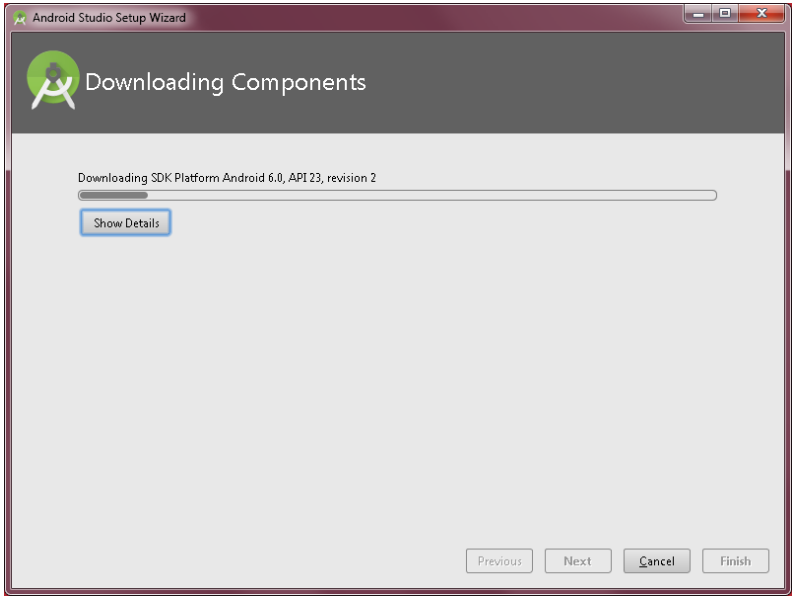


Fig 9. You may have to retry it (happened to me both times I installed Android Studio ® on different machines)

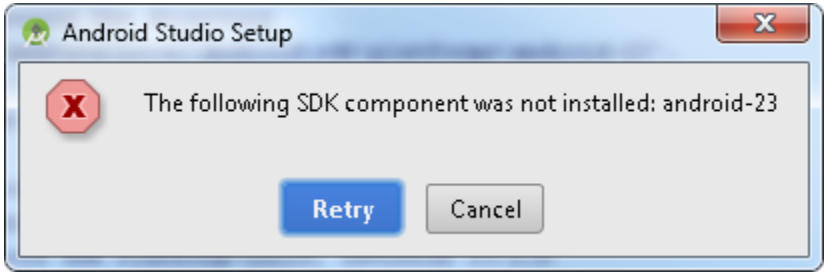


Fig 10. Android Studio ® is ready

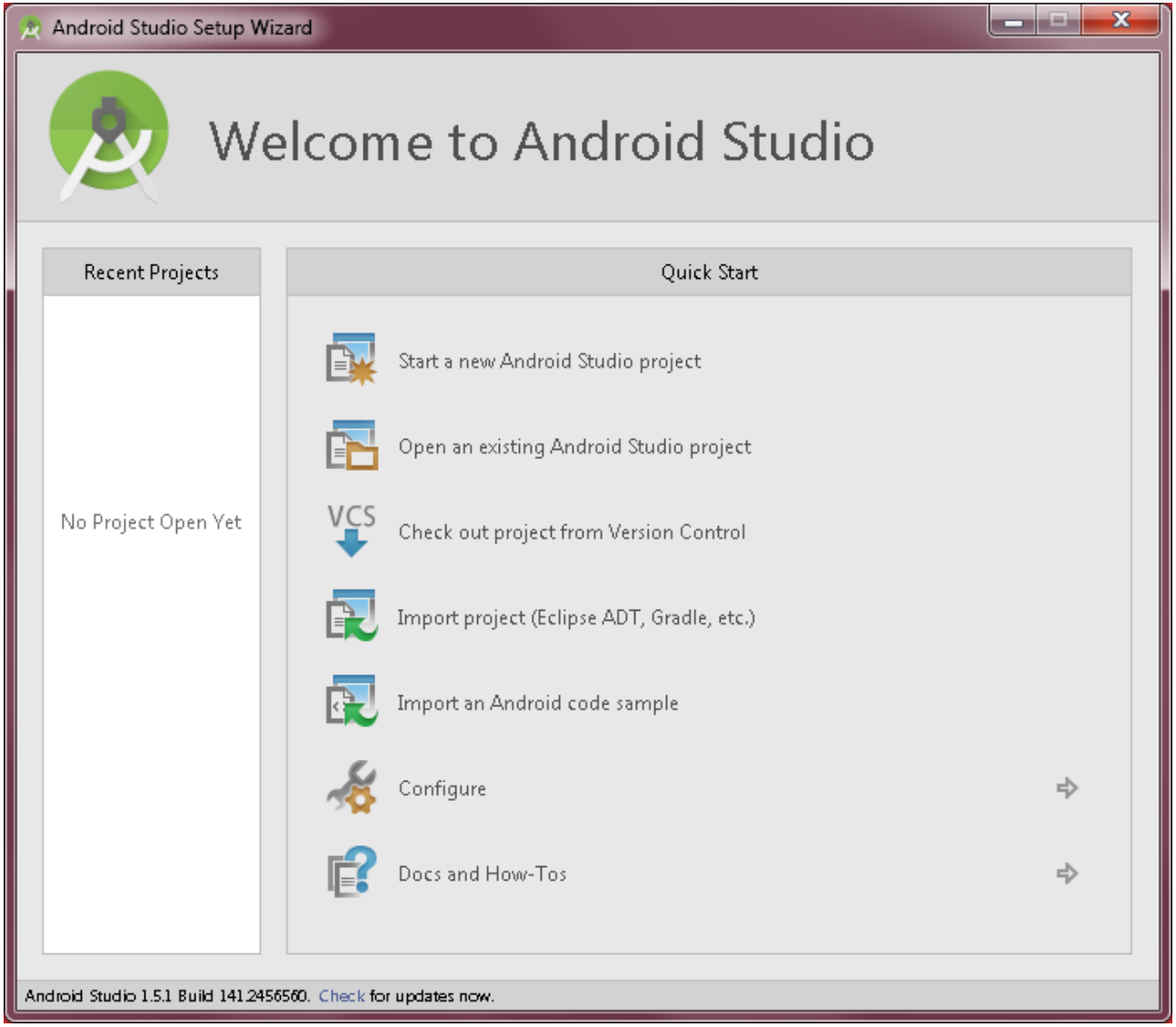
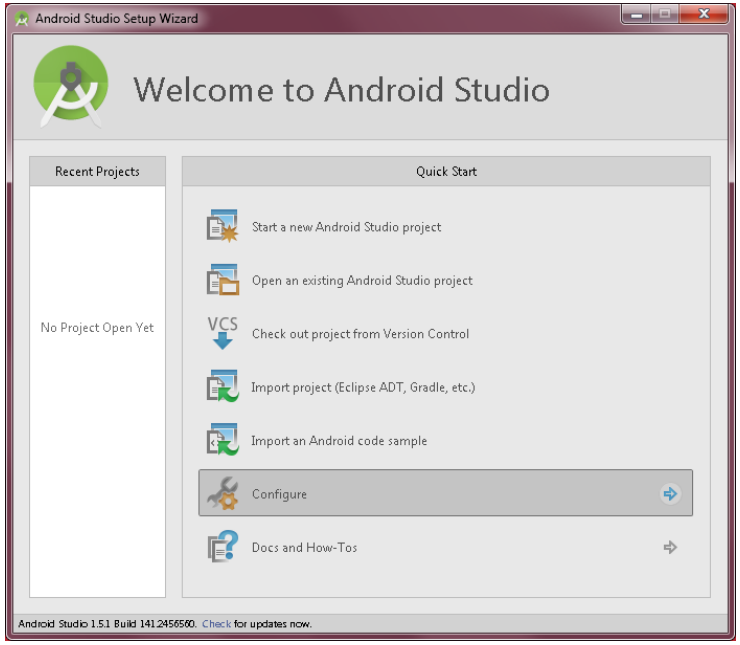
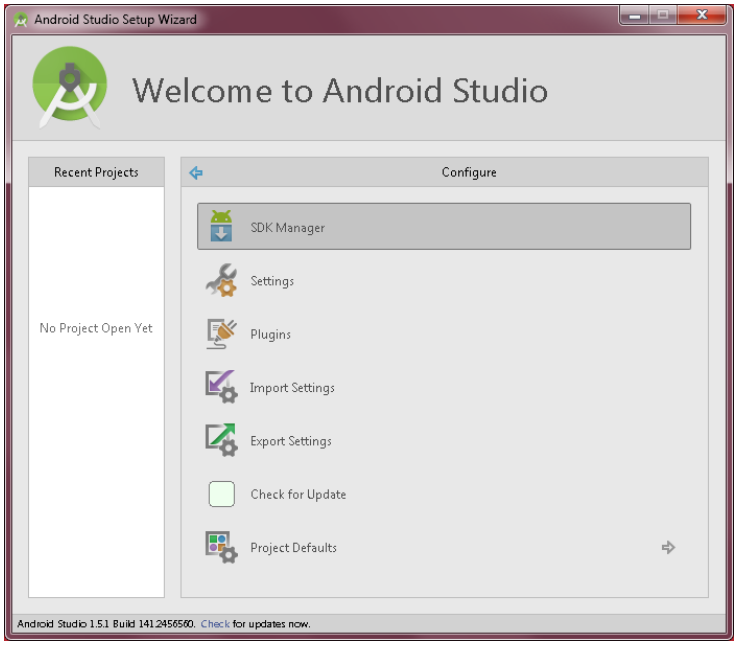
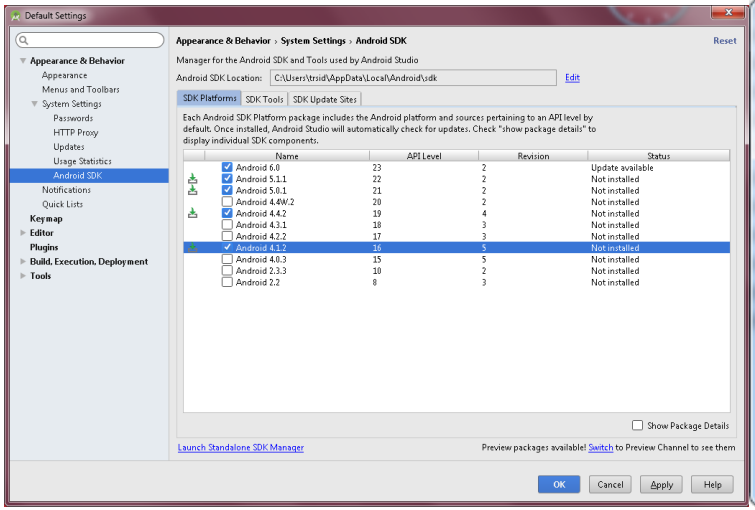


Fig 10. Add old SDK's (Android version 4.1, 4.2, 5.0). By default there is only 6.0







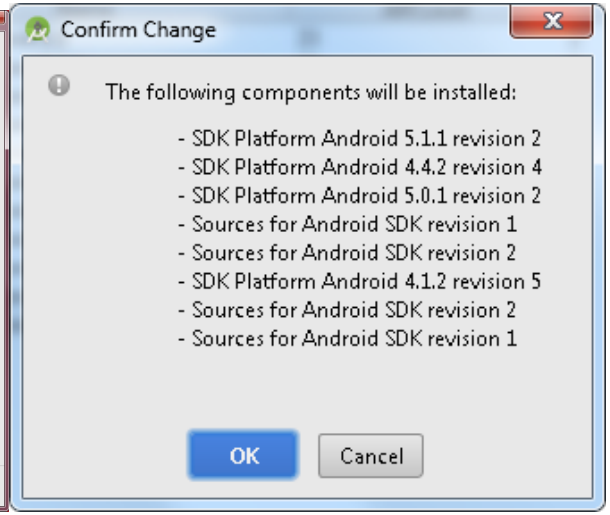
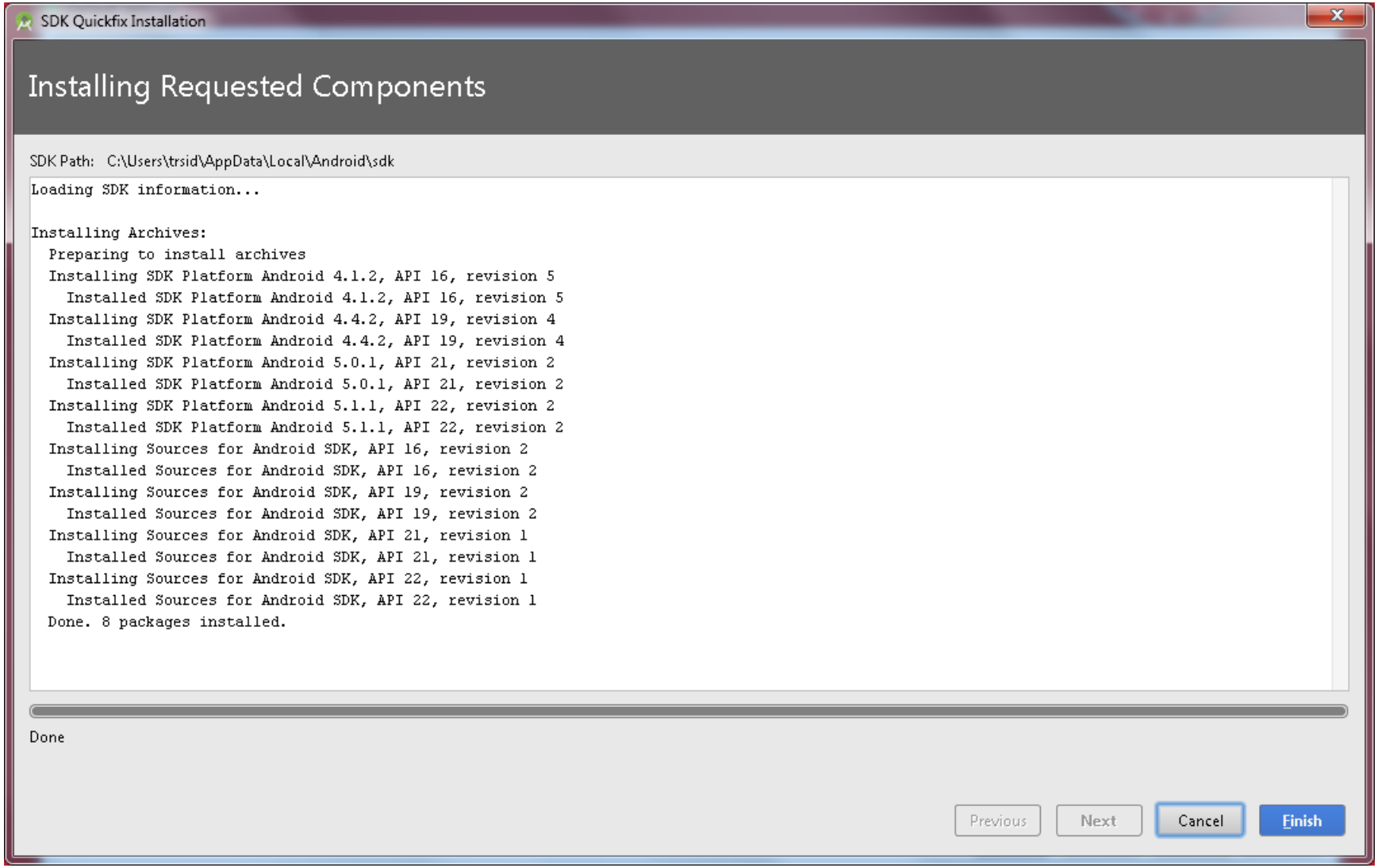


Fig 12. Old SDK's installed



Section 2

Starting Coding

Fig 13. Start a new Project

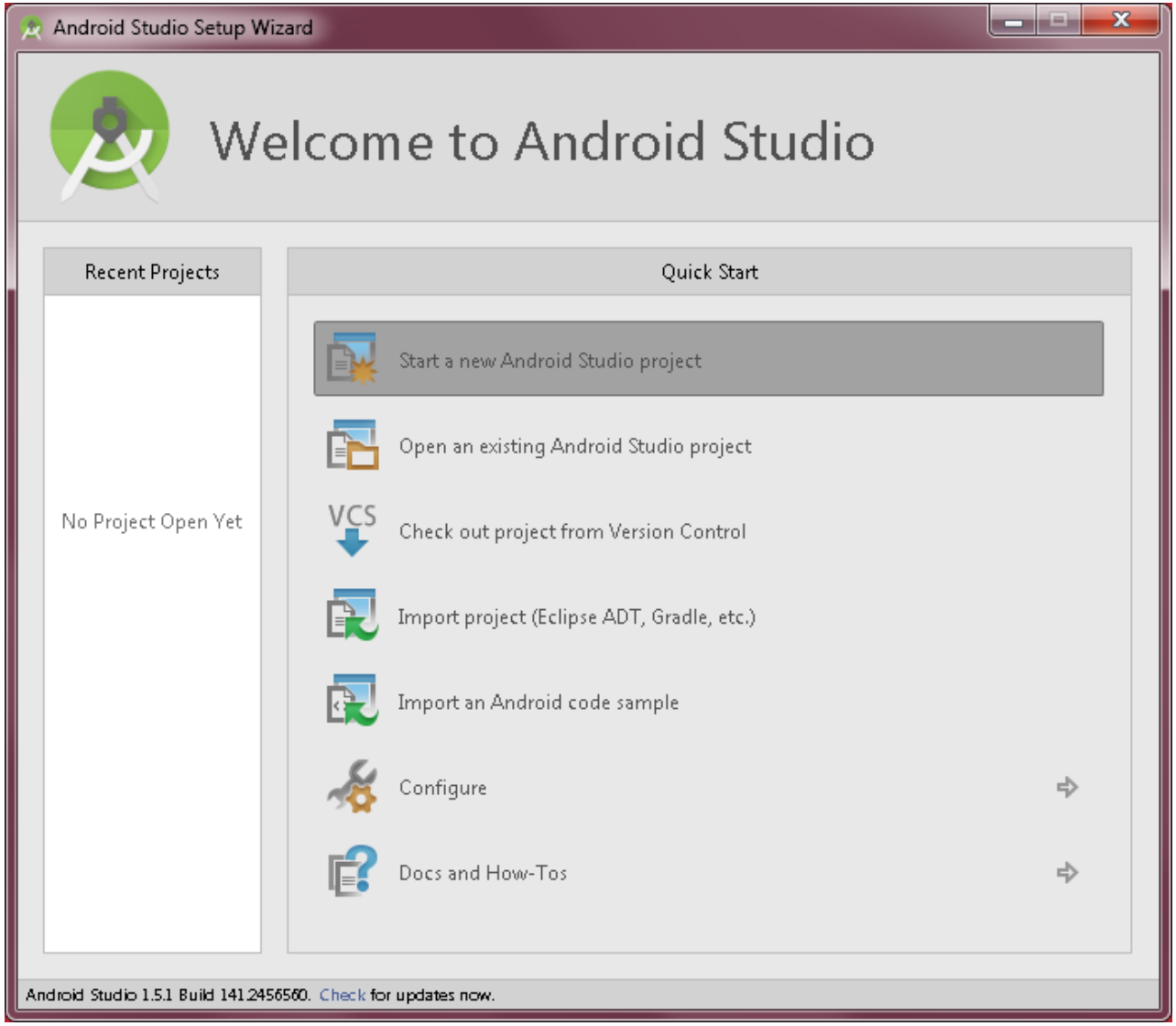


Fig 14. Name your project

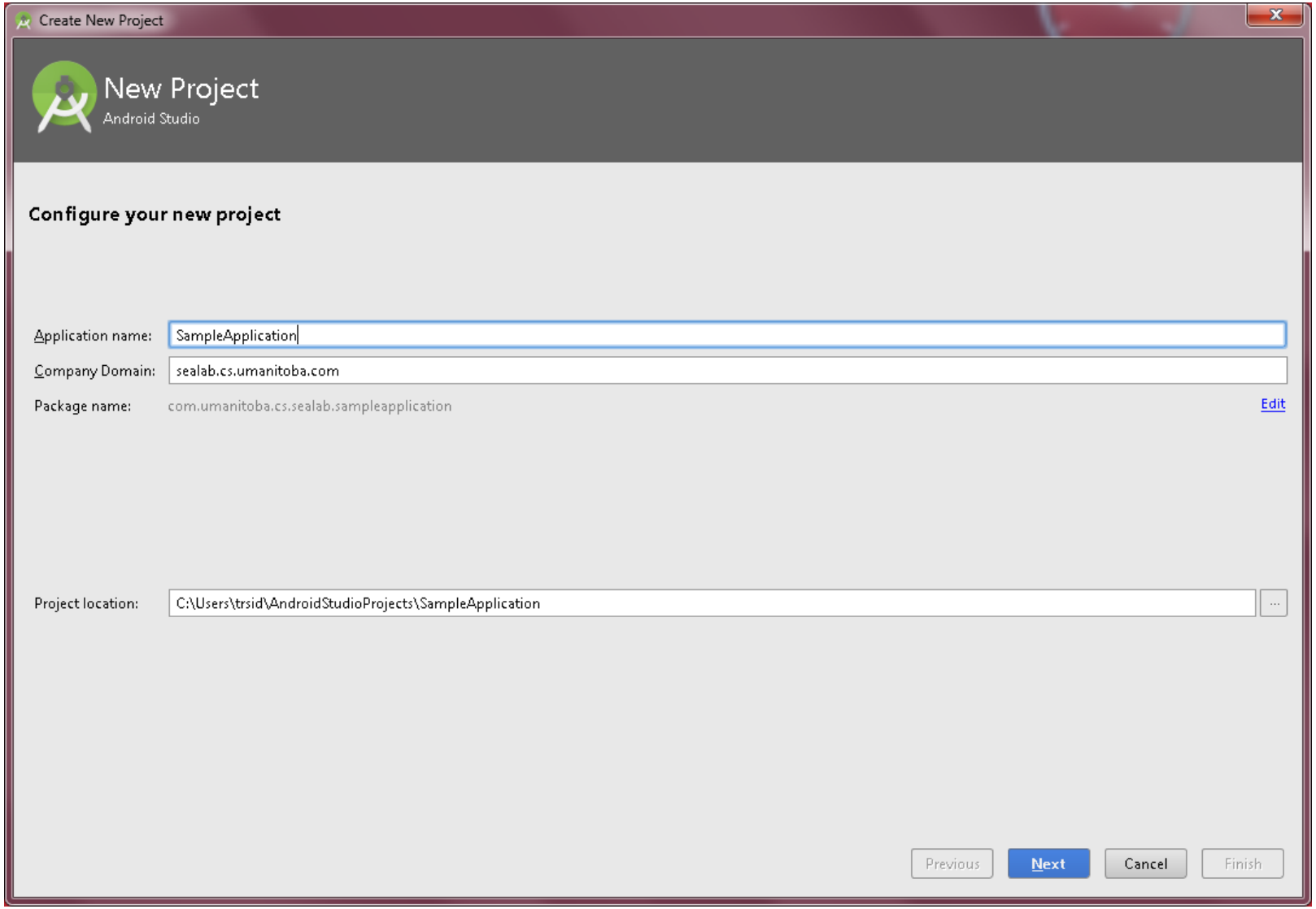


Fig 14. Select the minimum supported version (SDK)

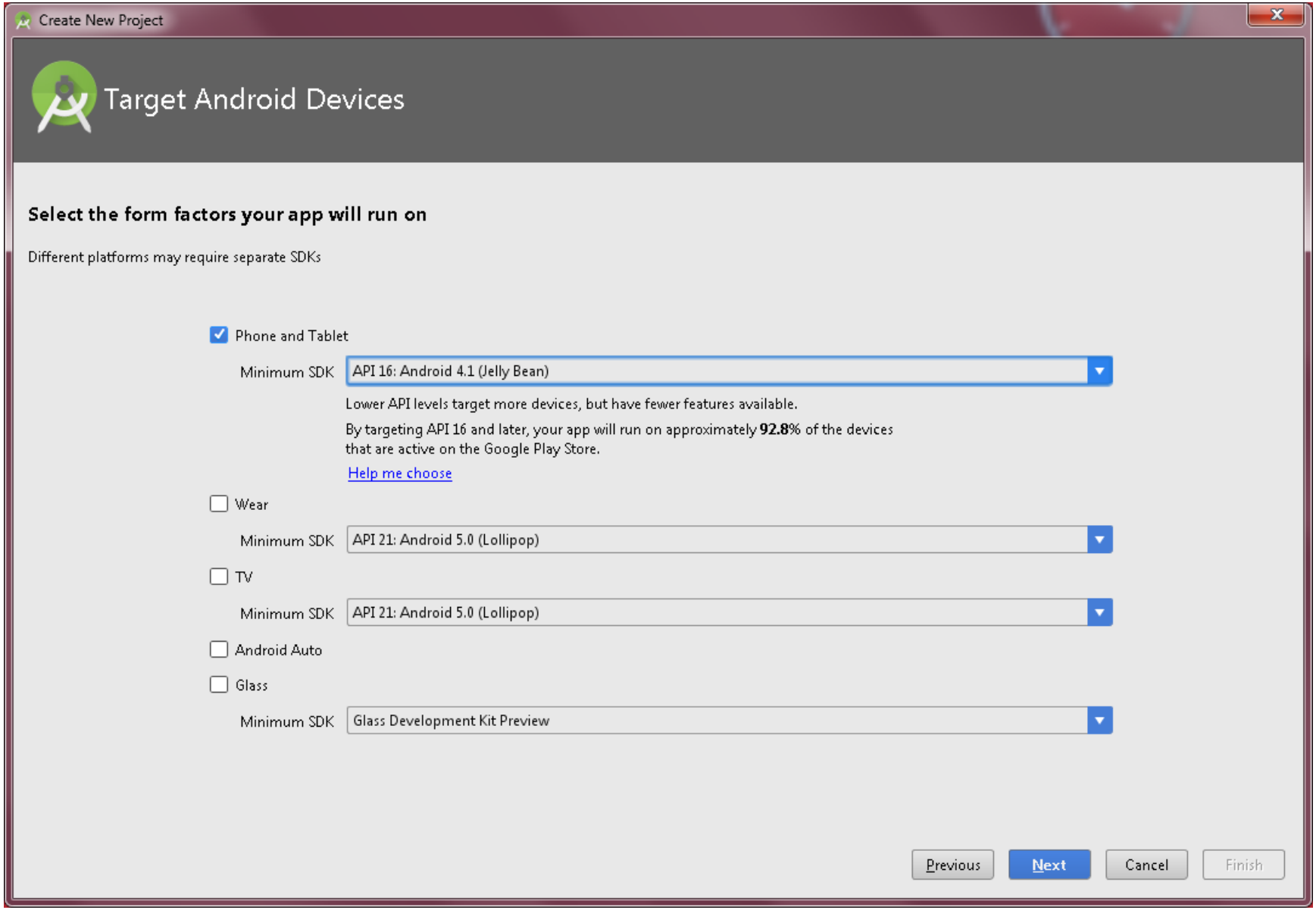


Fig 15. Select the main activity (The main view when your app opens)

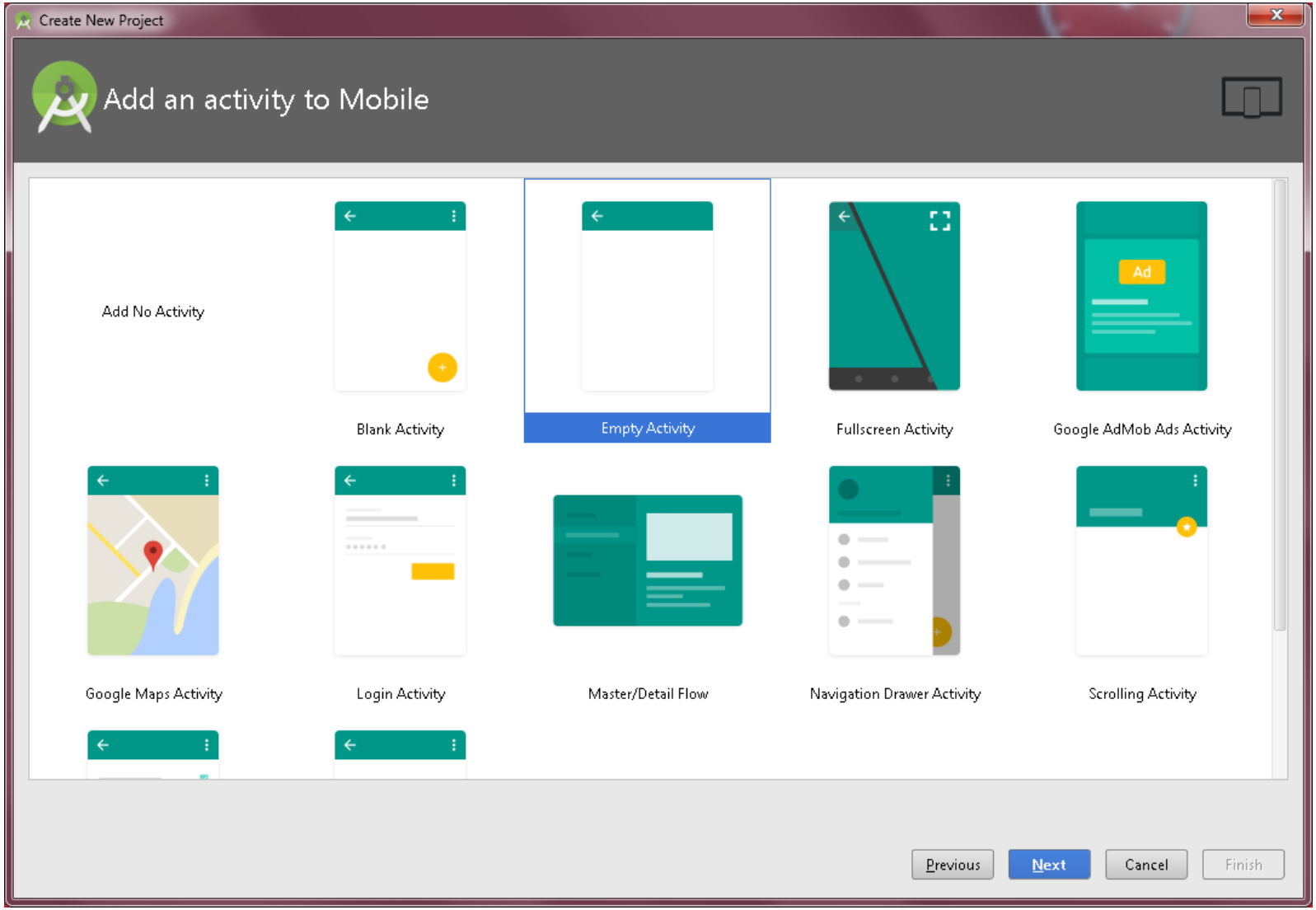


Fig 16. Lets call it MainActivity

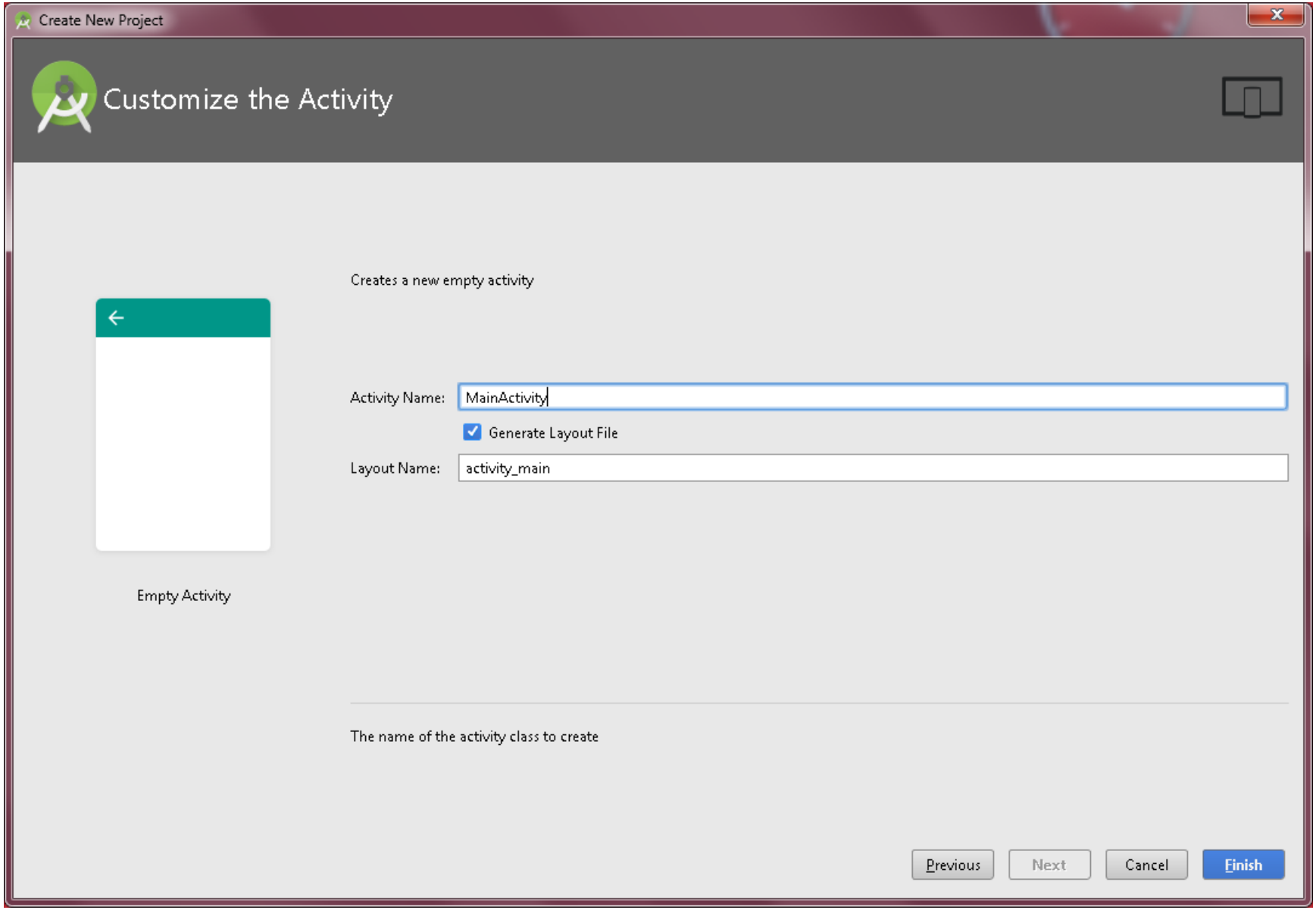
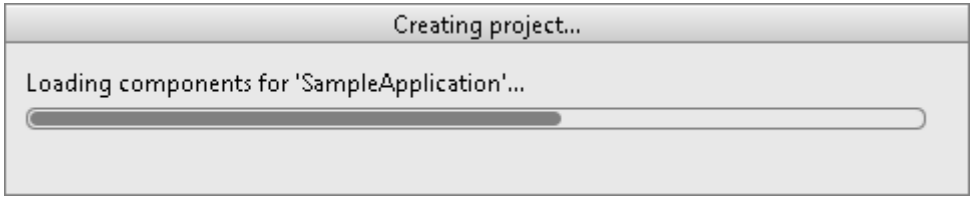


Fig 17. Project is Loading



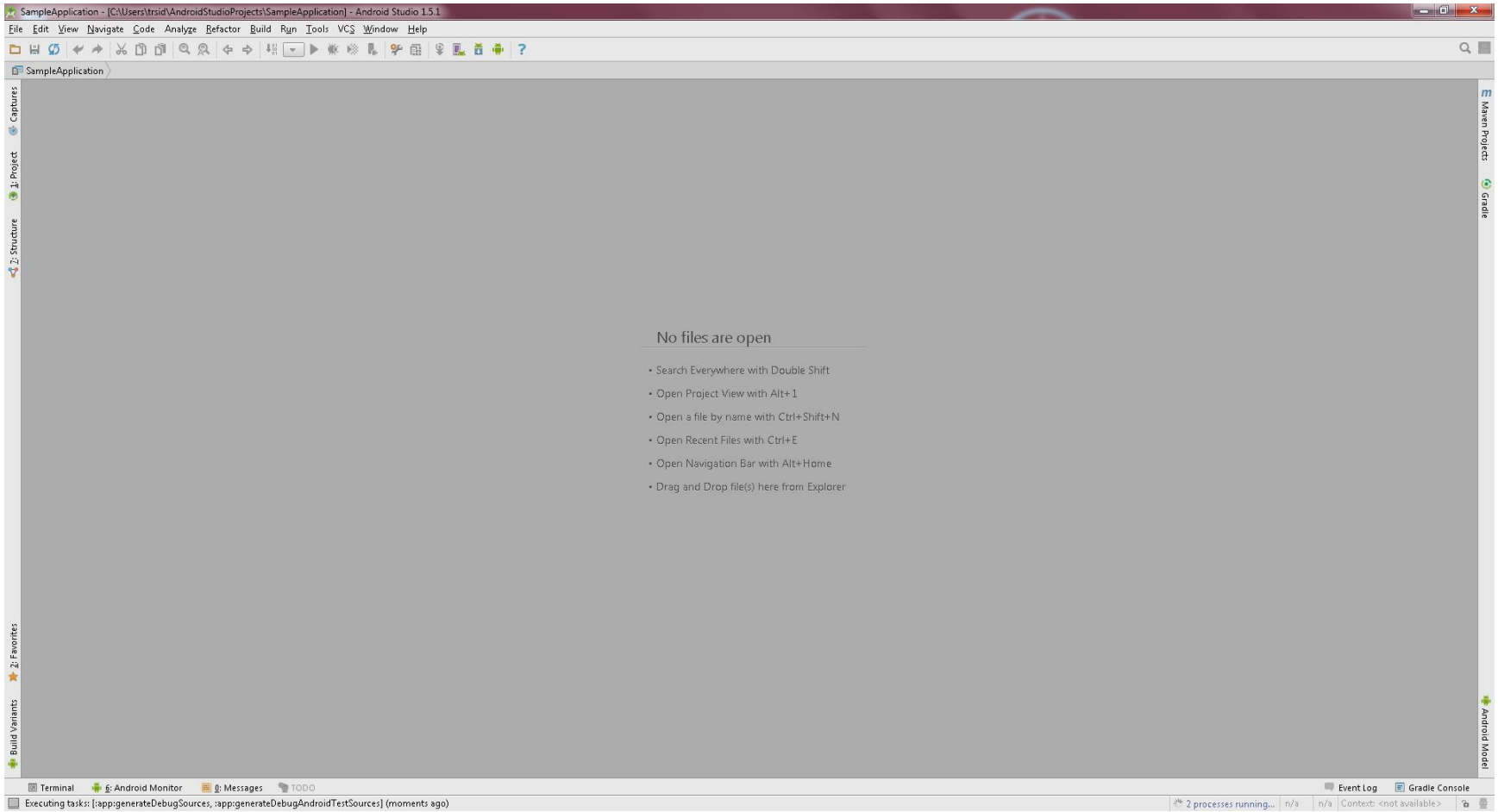


Fig 18. Project Ready. Notice the file structure on the left. Your java code is going to be in java folder, and their corresponding views in layout

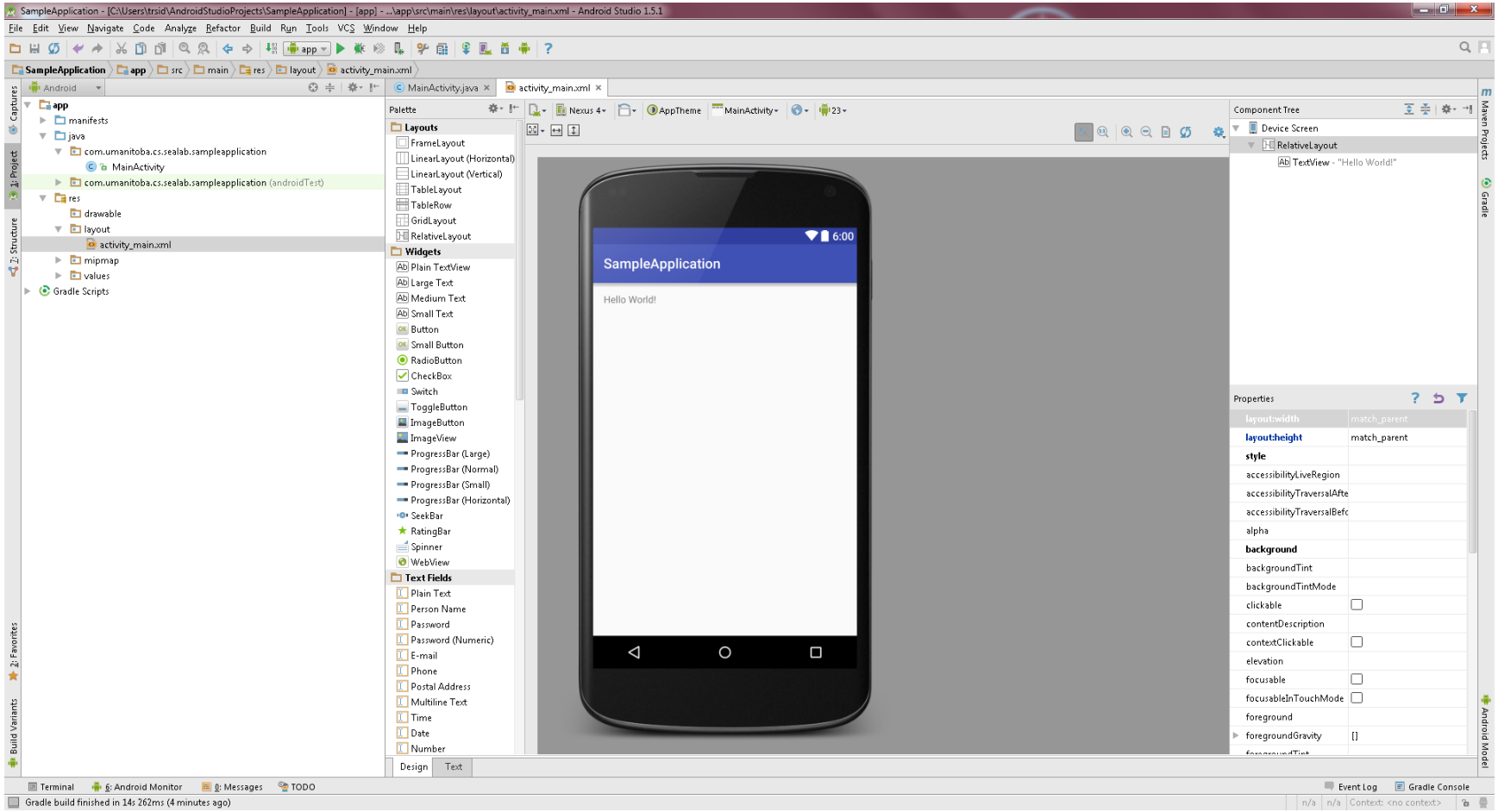


Fig 19. Edit the activity_main.xml under layout. This is the main page where we are showing the List of entries. Remove the Hello World Text View.

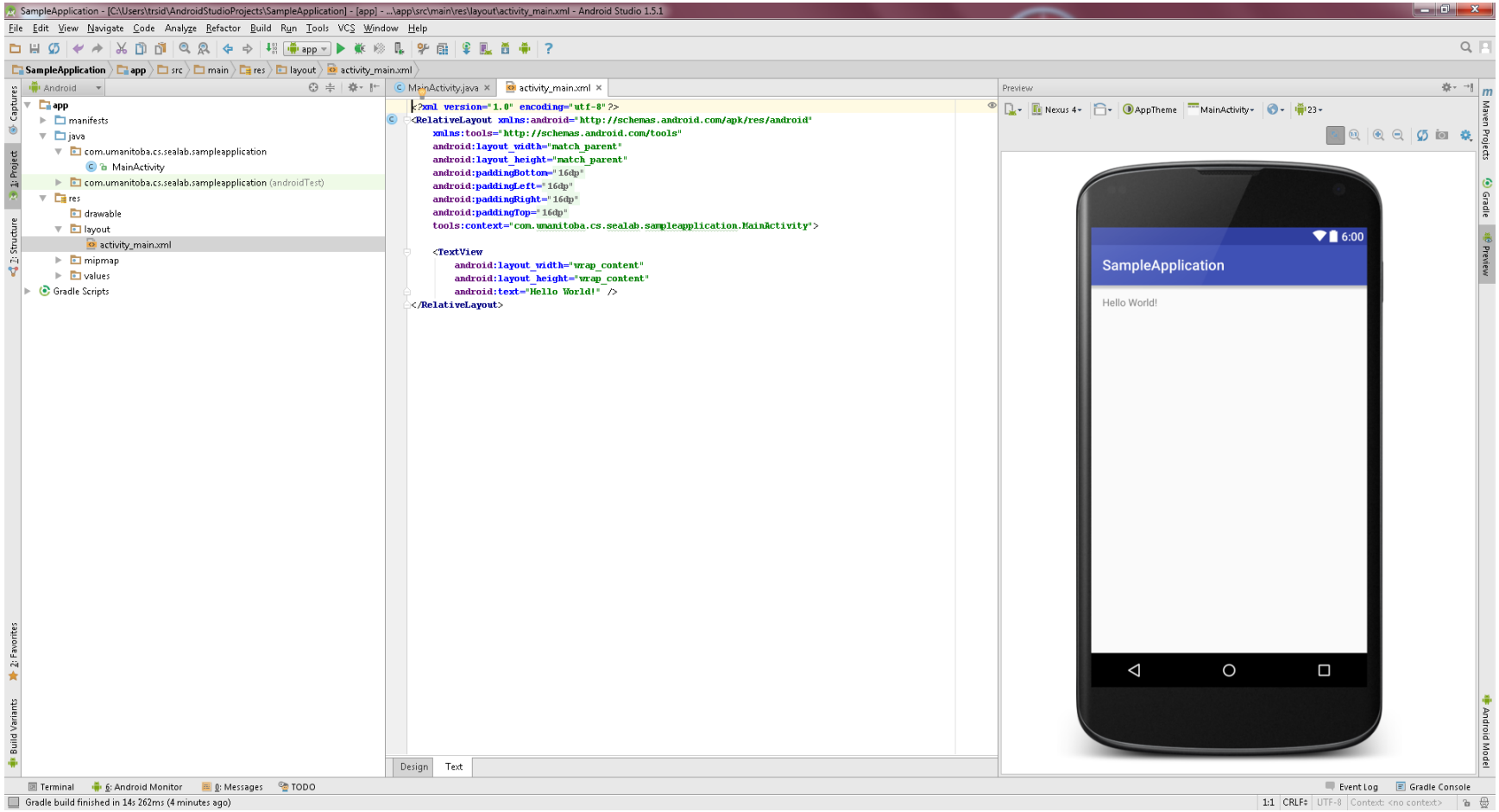
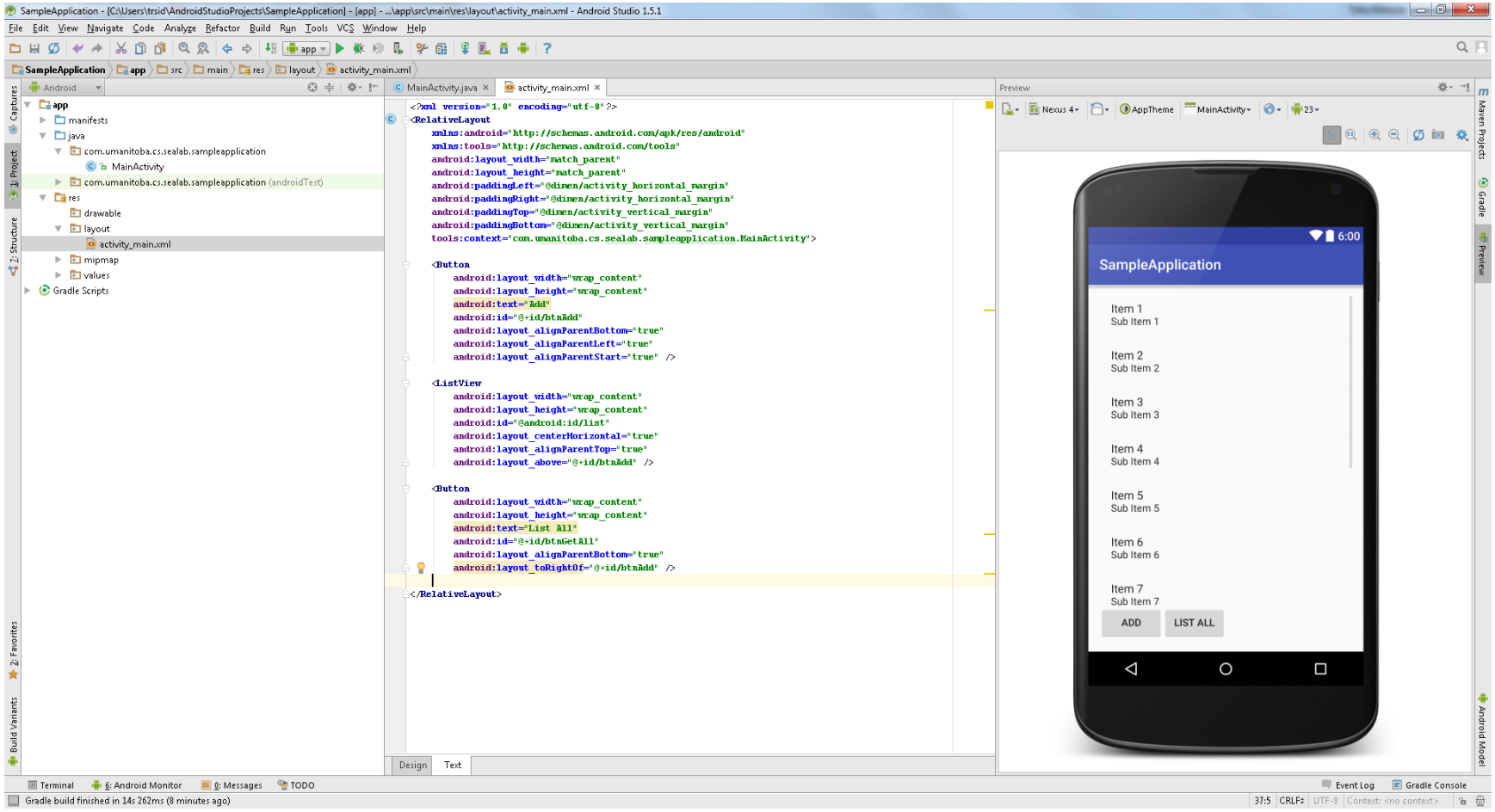


Fig 20. Add the following controls. 1) Button to Insert and item 2) Button to refresh the List 3) List of items inserted.



Code 1. activity_main.xml

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Add"
    android:id="@+id/btnAdd"
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />

<ListView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@android:id/list"
    android:layout_centerHorizontal="true"
    android:layout_alignParentTop="true"
    android:layout_above="@+id/btnAdd" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="List All"
    android:id="@+id/btnGetAll"
    android:layout_alignParentBottom="true"
    android:layout_toRightOf="@+id/btnAdd" />
```

Fig 21. Add a new activity for showing the detail of each list item.

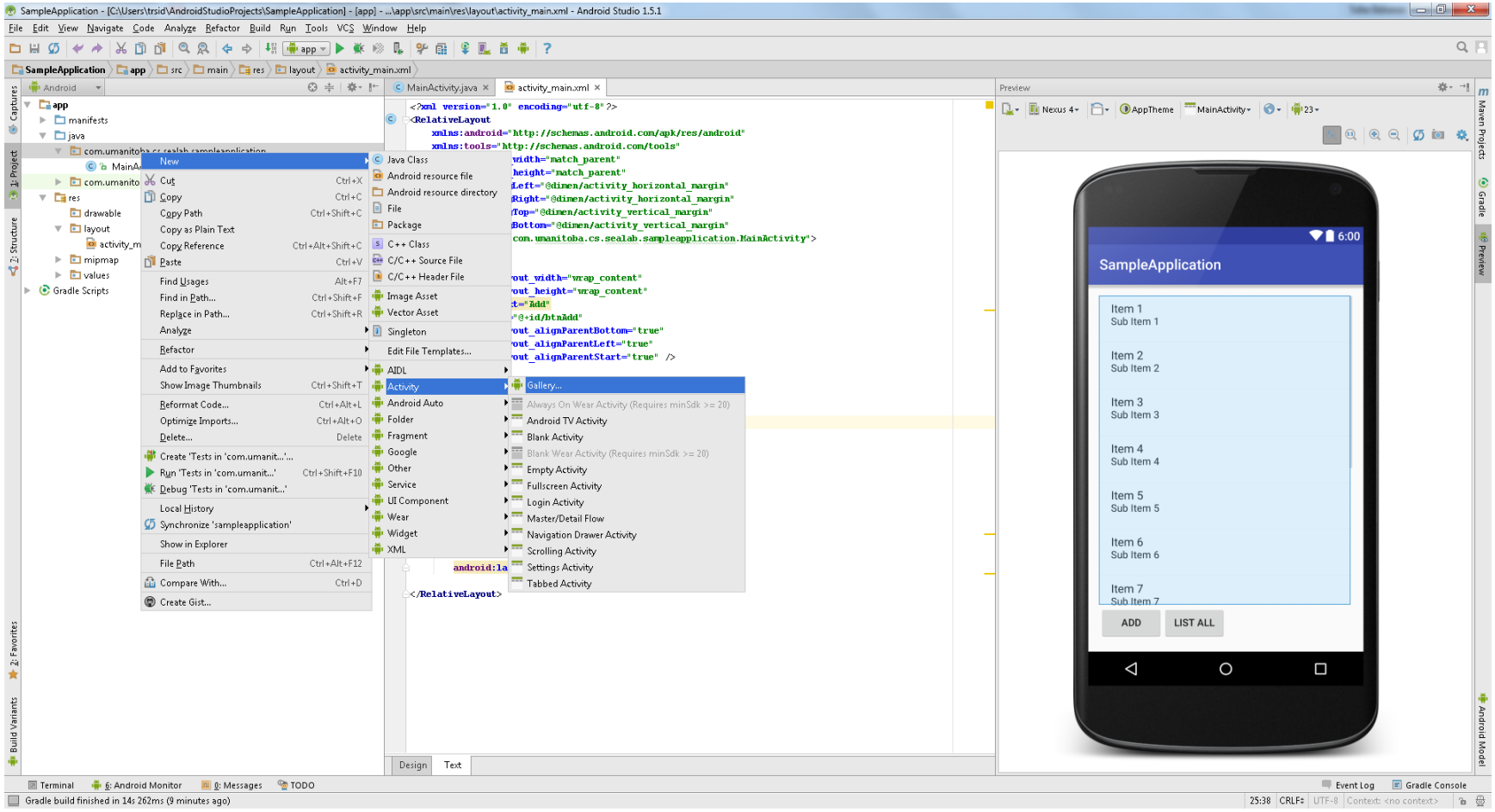


Fig 22. Add a new Empty activity for showing the detail of each list item (cont.).

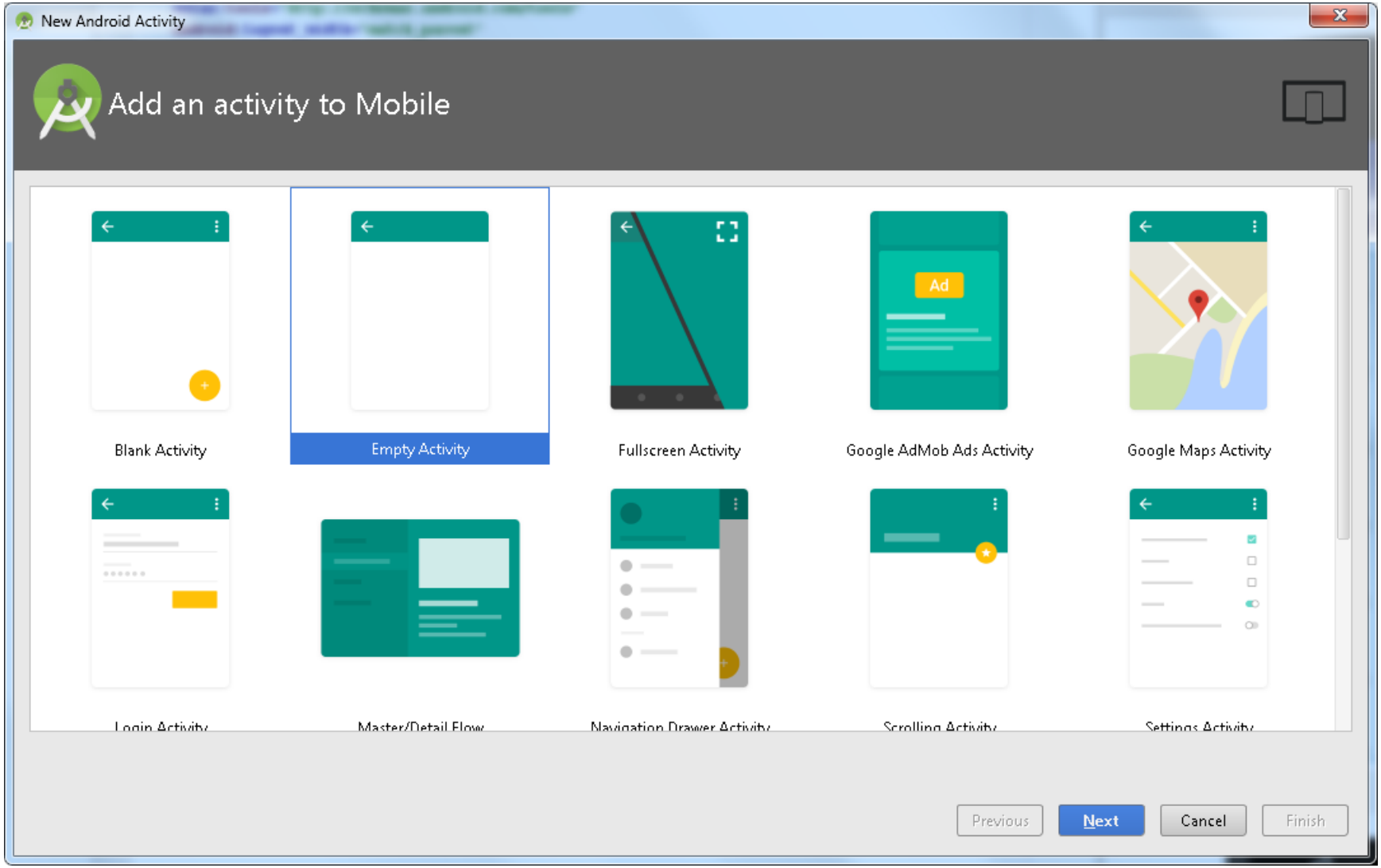


Fig 23. Name the new activity StudentDetail.

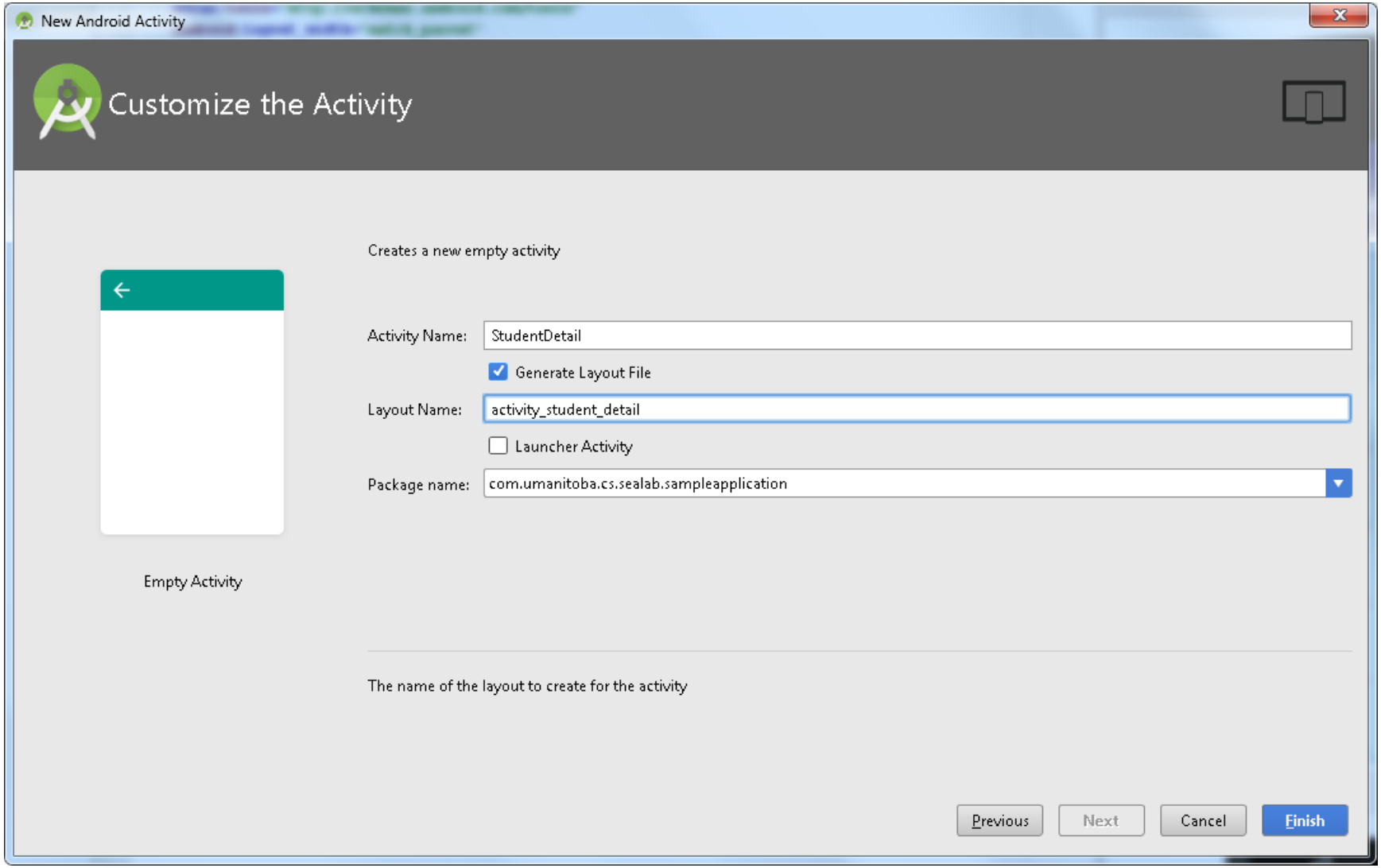
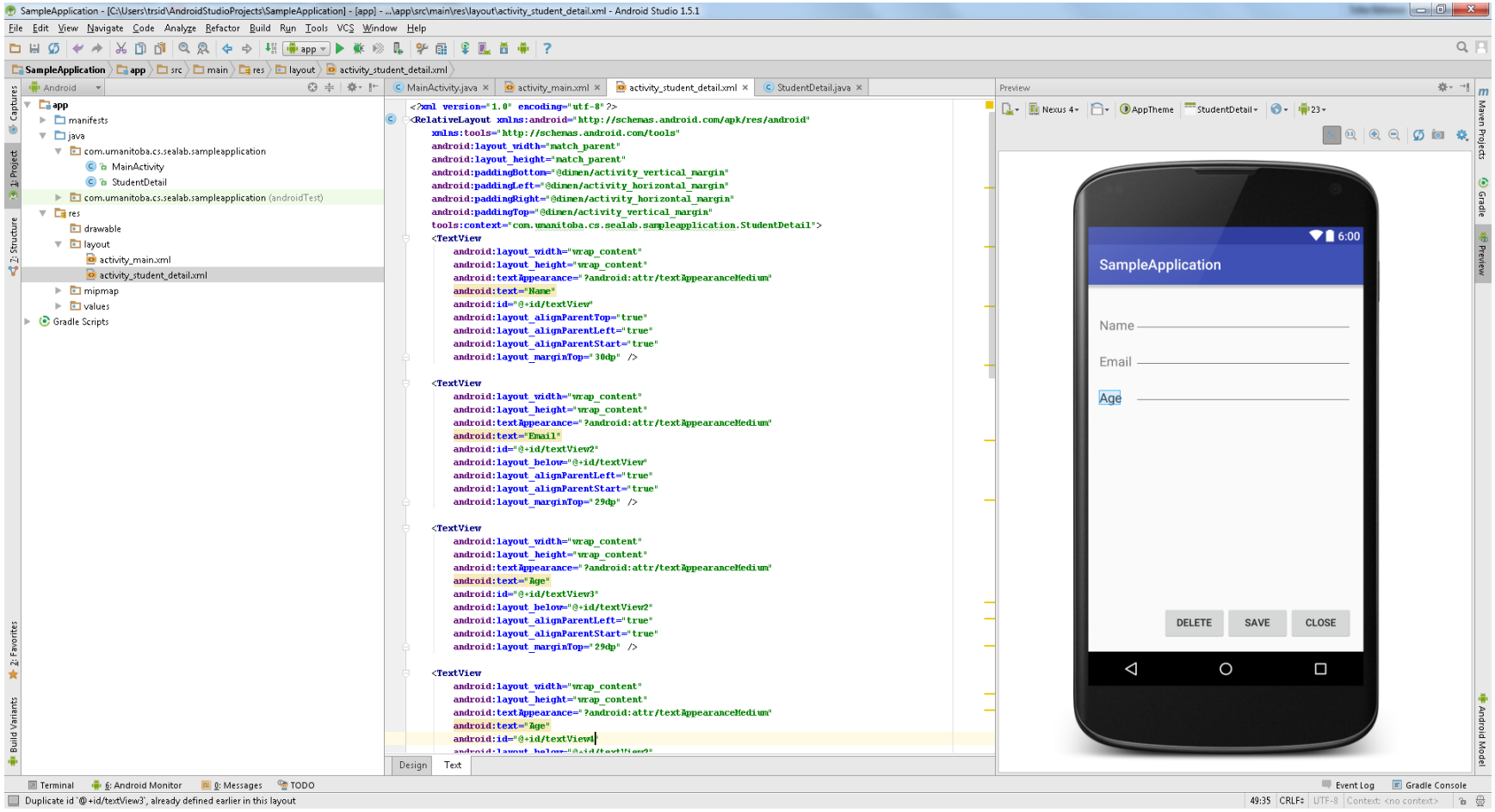


Fig 24. Add Controls according to your fields in the activity_student_detail.xml just created above. Text views for Name, Email and Age.



Code 2. activity_student_detail.xml

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="Name"
    android:id="@+id/textView"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginTop="30dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="Email"
    android:id="@+id/textView2"
    android:layout_below="@+id/textView"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginTop="29dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="Age"
    android:id="@+id/textView3"
    android:layout_below="@+id/textView2"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginTop="29dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="Age"
    android:id="@+id/textView4"
    android:layout_below="@+id/textView2"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginTop="29dp" />

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:inputType="textPersonName"
    android:ems="10"
    android:id="@+id/editTextName"
    android:layout_above="@+id/textView2"
    android:layout_toRightOf="@+id/textView"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />
```

```
<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:inputType="textEmailAddress"
    android:ems="10"
    android:id="@+id/editTextEmail"
    android:layout_above="@+id/textView3"
    android:layout_toRightOf="@+id/textView"
    android:layout_alignRight="@+id/editTextName"
    android:layout_alignEnd="@+id/editTextName" />

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:inputType="number"
    android:ems="10"
    android:id="@+id/editTextAge"
    android:layout_alignBottom="@+id/textView3"
    android:layout_alignLeft="@+id/editTextEmail"
    android:layout_alignStart="@+id/editTextEmail"
    android:layout_alignRight="@+id/editTextEmail"
    android:layout_alignEnd="@+id/editTextEmail" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Save"
    android:id="@+id/btnSave"
    android:layout_alignParentBottom="true"
    android:layout_toLeftOf="@+id/btnClose" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Close"
    android:id="@+id/btnClose"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Delete"
    android:id="@+id/btnDelete"
    android:layout_alignTop="@+id/btnSave"
    android:layout_toLeftOf="@+id/btnSave" />
```

Fig 25. Add a new view (layout) for individual list item to be shown in the list in Fig 20.

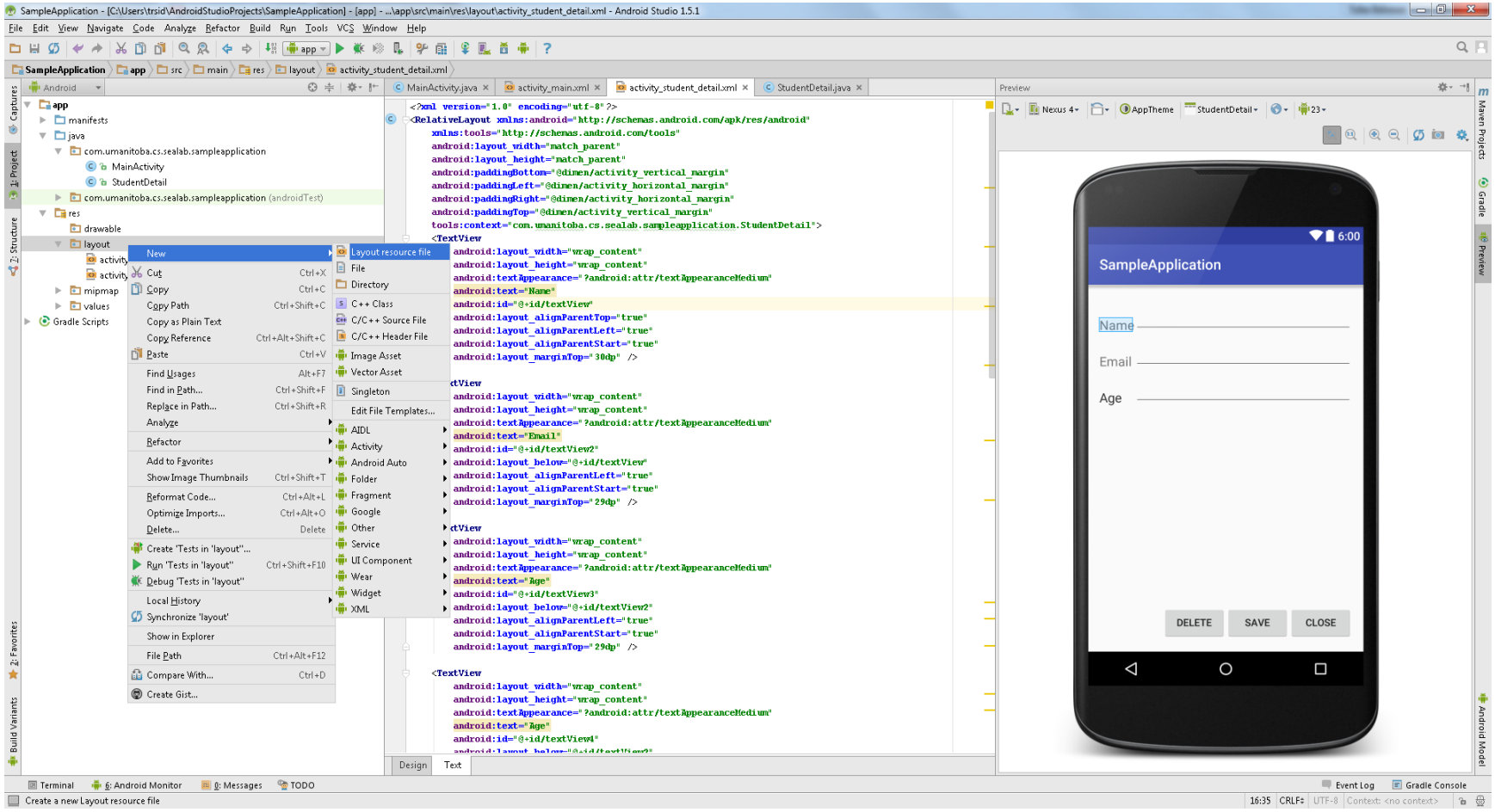
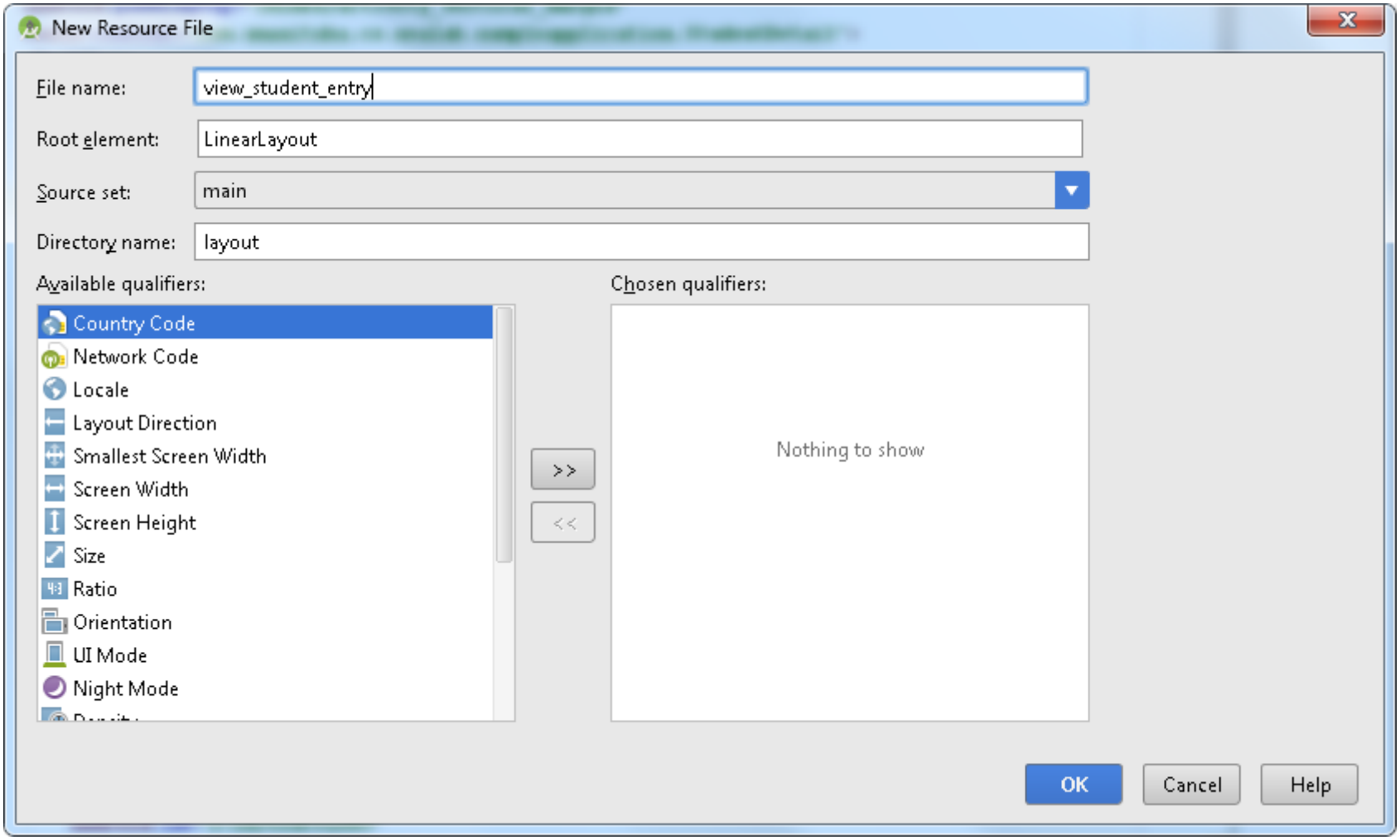


Fig 26. Name the view view_student_entry and add the control in the Code 3 below



Code 3. view_student_entry.xml

```
<TextView
    android:id="@+id/student_name"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:paddingLeft="6dip"
    android:paddingTop="6dip"
    android:textSize="22sp"
    android:textStyle="bold" />
```

Fig 27. Add a new class for the items in the database

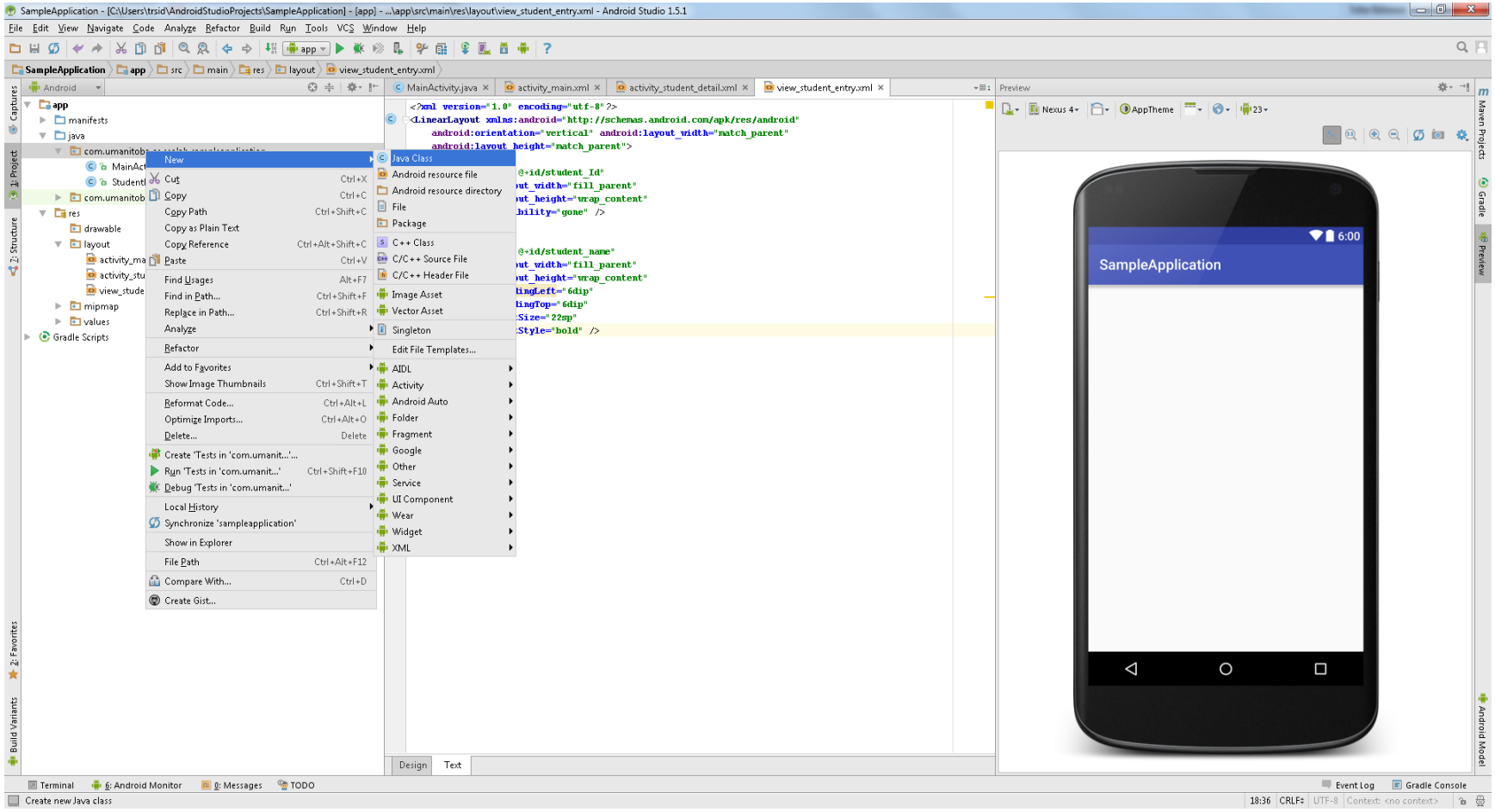
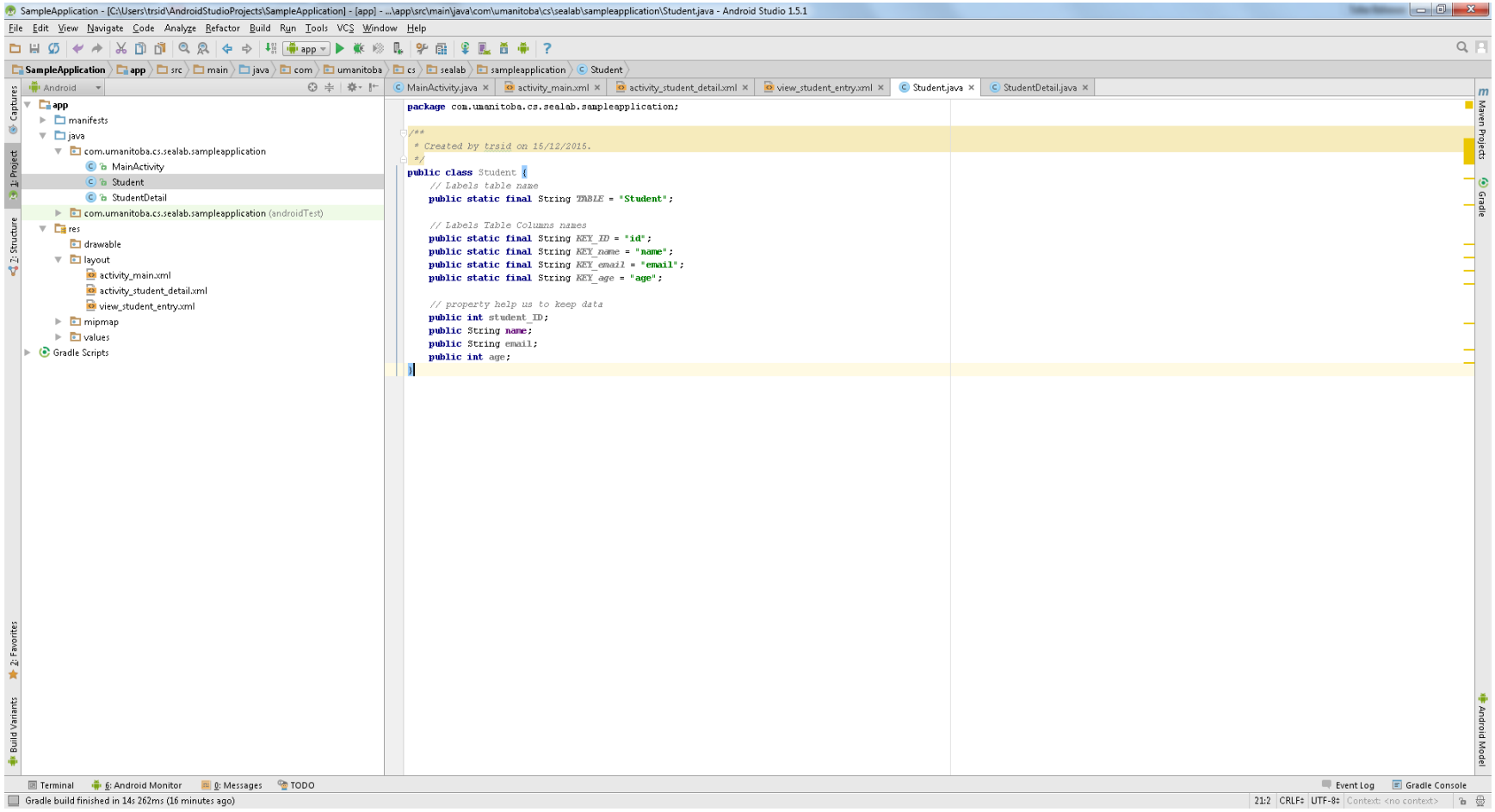


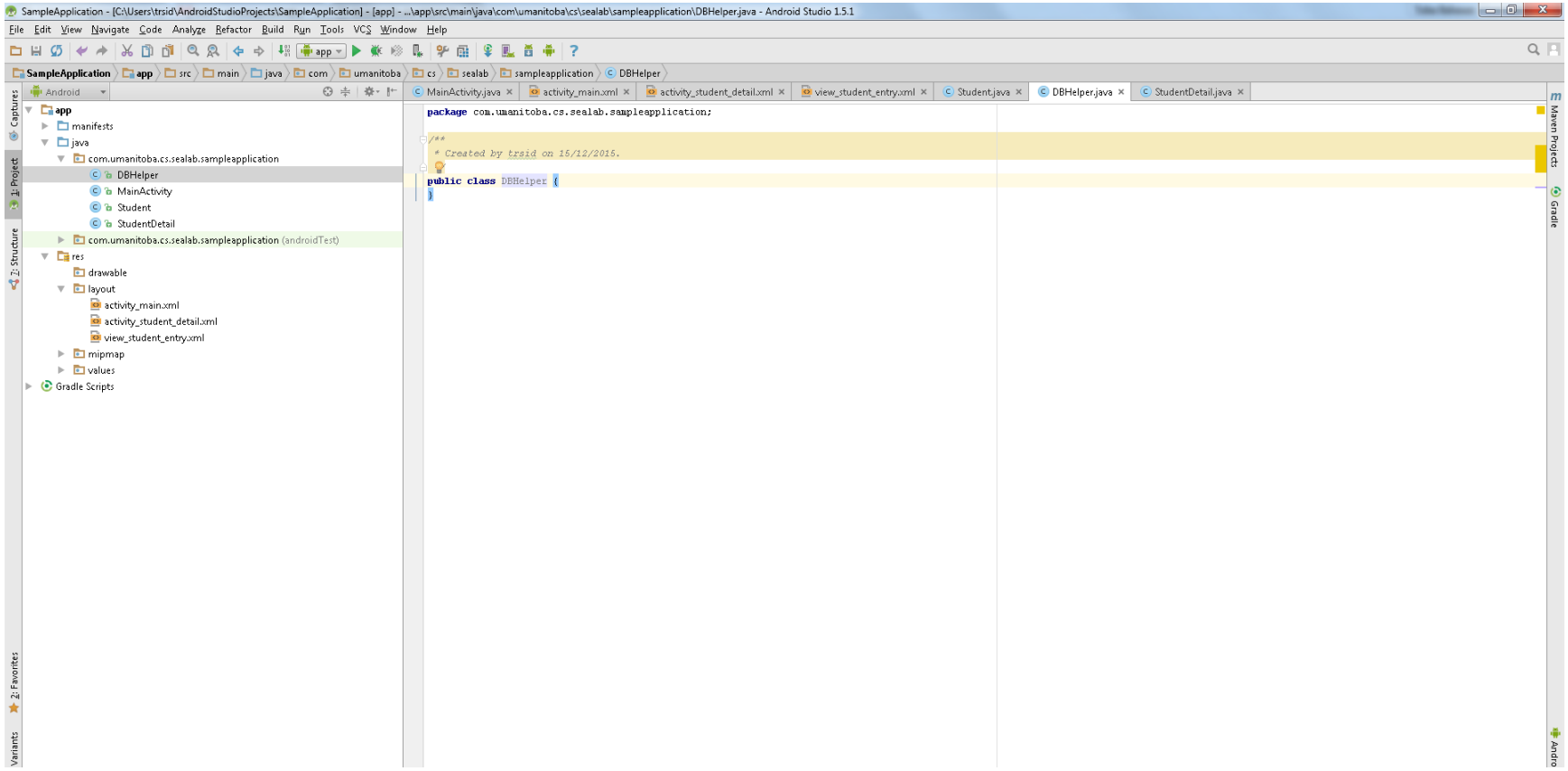
Fig 28. Add properties to your class



Code 4. student.java

```
public class Student {
    // Labels table name
    public static final String TABLE = "Student";

    // Labels Table Columns names
    public static final String KEY_ID = "id";
    public static final String KEY_name = "name";
    public static final String KEY_email = "email";
    public static final String KEY_age = "age";

    // property help us to keep data
    public int student_ID;
    public String name;
    public String email;
    public int age;
}
```

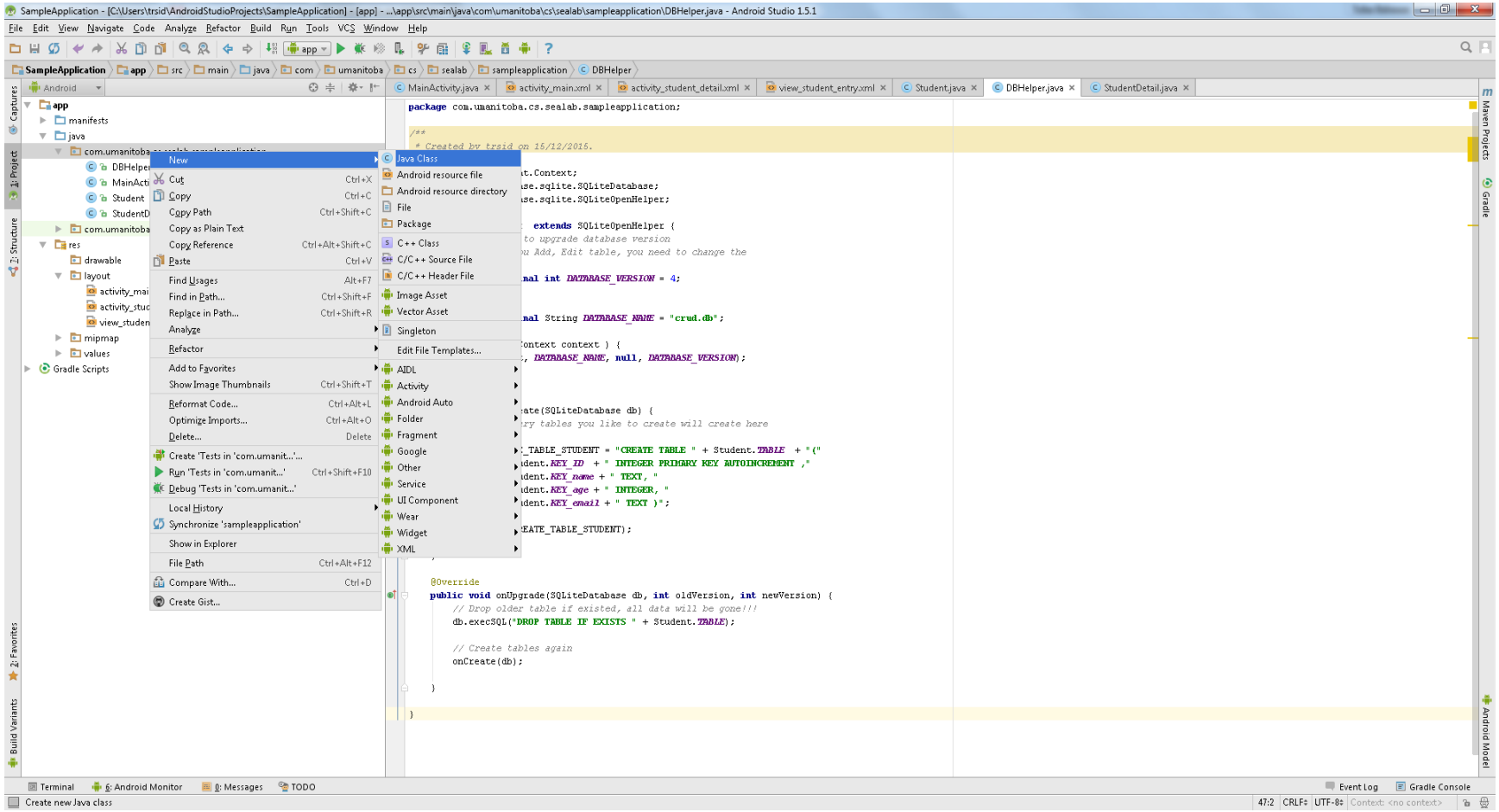
Fig 29. Create a new class for handling the Database, DBHelper and add Code 5



Code 5. DBHelper.java

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBHelper  extends SQLiteOpenHelper {
    //version number to upgrade database version
    //each time if you Add, Edit table, you need to change the
    //version number.
    private static final int DATABASE_VERSION = 4;

    // Database Name
    private static final String DATABASE_NAME = "crud.db";

    public DBHelper(Context context ) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        //All necessary tables you like to create will create here

        String CREATE_TABLE_STUDENT = "CREATE TABLE " + Student.TABLE  + "("
                + Student.KEY_ID  + " INTEGER PRIMARY KEY AUTOINCREMENT ,"
                + Student.KEY_name + " TEXT, "
                + Student.KEY_age + " INTEGER, "
                + Student.KEY_email + " TEXT )";

        db.execSQL(CREATE_TABLE_STUDENT);

    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // Drop older table if existed, all data will be gone!!!
        db.execSQL("DROP TABLE IF EXISTS " + Student.TABLE);

        // Create tables again
        onCreate(db);

    }

}
```

Fig 30. Create a new class StudentRepo for handling the database operations (Create, Read, Update, Delete) for the Student.java class and add Code 6



Code 6. StudentRepo.java

```
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import java.util.ArrayList;
import java.util.HashMap;

public class StudentRepo {
    private DBHelper dbHelper;

    public StudentRepo(Context context) {
        dbHelper = new DBHelper(context);
    }

    public int insert(Student student) {

        //Open connection to write data
        SQLiteDatabase db = dbHelper.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(Student.KEY_age, student.age);
        values.put(Student.KEY_email,student.email);
        values.put(Student.KEY_name, student.name);
        // Inserting Row
        long student_Id = db.insert(Student.TABLE, null, values);
        db.close(); // Closing database connection
        return (int) student_Id;
    }

    public void delete(int student_Id) {

        SQLiteDatabase db = dbHelper.getWritableDatabase();
        // It's a good practice to use parameter ?, instead of concatenate string
        db.delete(Student.TABLE, Student.KEY_ID + "= ?", new String[] { String.valueOf(student_Id) });
        db.close(); // Closing database connection
    }

    public void update(Student student) {

        SQLiteDatabase db = dbHelper.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(Student.KEY_age, student.age);
        values.put(Student.KEY_email,student.email);
        values.put(Student.KEY_name, student.name);
        // It's a good practice to use parameter ?, instead of concatenate string
        db.update(Student.TABLE, values, Student.KEY_ID + "= ?", new String[] { String.valueOf(student.student_ID) });
        db.close(); // Closing database connection
    }

    public ArrayList<HashMap<String, String>>  getStudentList() {
        //Open connection to read only
        SQLiteDatabase db = dbHelper.getReadableDatabase();
        String selectQuery =  "SELECT  " +
                Student.KEY_ID + "," +
```

```
                Student.KEY_name + "," +
                Student.KEY_email + "," +
                Student.KEY_age +
                " FROM " + Student.TABLE;

        //Student student = new Student();
        ArrayList<HashMap<String, String>> studentList = new ArrayList<HashMap<String, String>>();

        Cursor cursor = db.rawQuery(selectQuery, null);
        // looping through all rows and adding to list

        if (cursor.moveToFirst()) {
            do {
                HashMap<String, String> student = new HashMap<String, String>();
                student.put("id", cursor.getString(cursor.getColumnIndex(Student.KEY_ID)));
                student.put("name", cursor.getString(cursor.getColumnIndex(Student.KEY_name)));
                studentList.add(student);

            } while (cursor.moveToNext());
        }

        cursor.close();
        db.close();
        return studentList;

    }

    public Student getStudentById(int Id){
        SQLiteDatabase db = dbHelper.getReadableDatabase();
        String selectQuery =  "SELECT  " +
                Student.KEY_ID + "," +
                Student.KEY_name + "," +
                Student.KEY_email + "," +
                Student.KEY_age +
                " FROM " + Student.TABLE
                + " WHERE " +
                Student.KEY_ID + "=?";// It's a good practice to use parameter ?, instead of concatenate string

        int iCount =0;
        Student student = new Student();

        Cursor cursor = db.rawQuery(selectQuery, new String[] { String.valueOf(Id) } );

        if (cursor.moveToFirst()) {
            do {
                student.student_ID =cursor.getInt(cursor.getColumnIndex(Student.KEY_ID));
                student.name =cursor.getString(cursor.getColumnIndex(Student.KEY_name));
                student.email  =cursor.getString(cursor.getColumnIndex(Student.KEY_email));
                student.age =cursor.getInt(cursor.getColumnIndex(Student.KEY_age));

            } while (cursor.moveToNext());
        }

        cursor.close();
        db.close();
        return student;
    }

}
```

Fig 31. This is the final structure of the project (on the left).

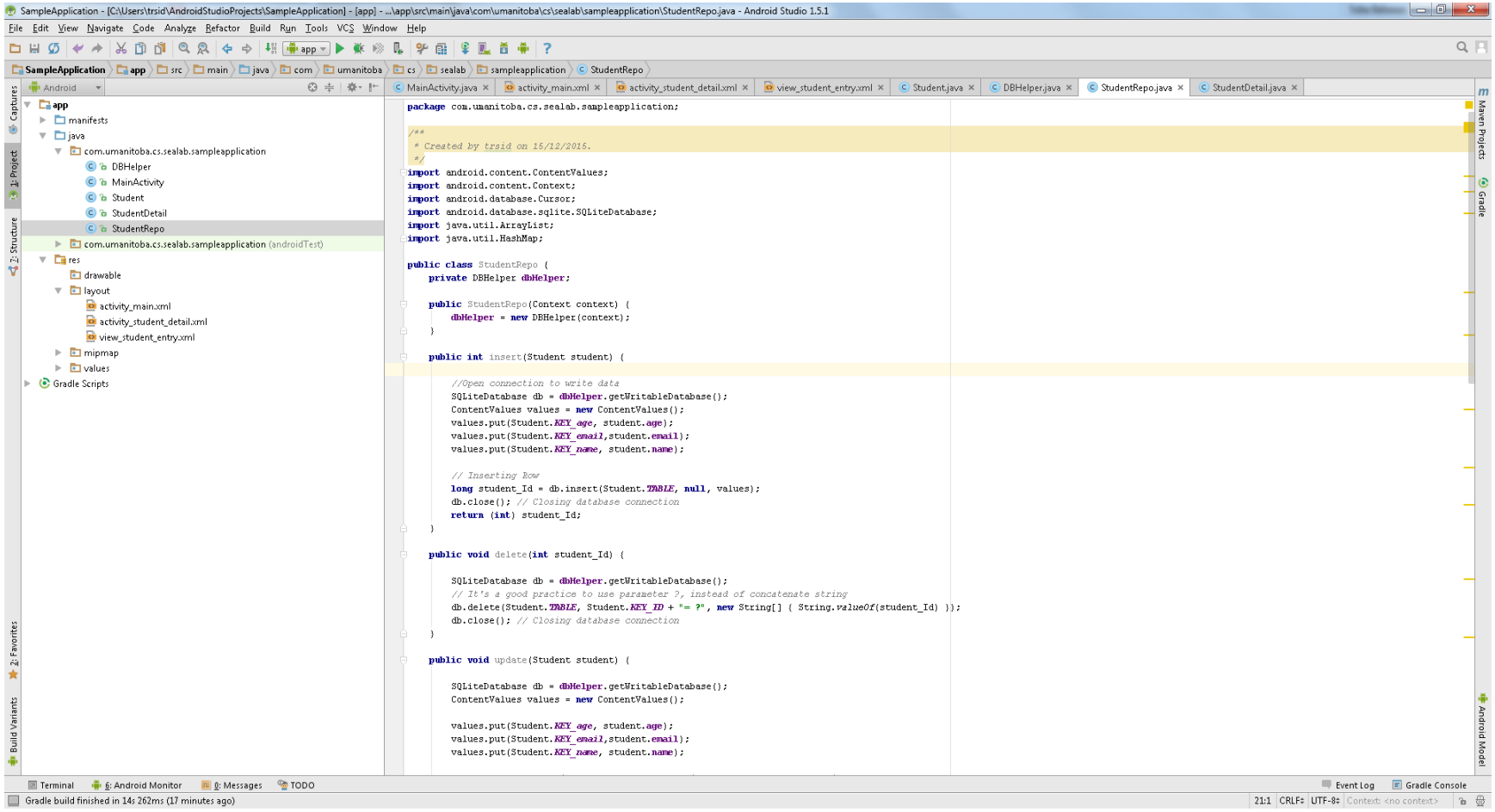
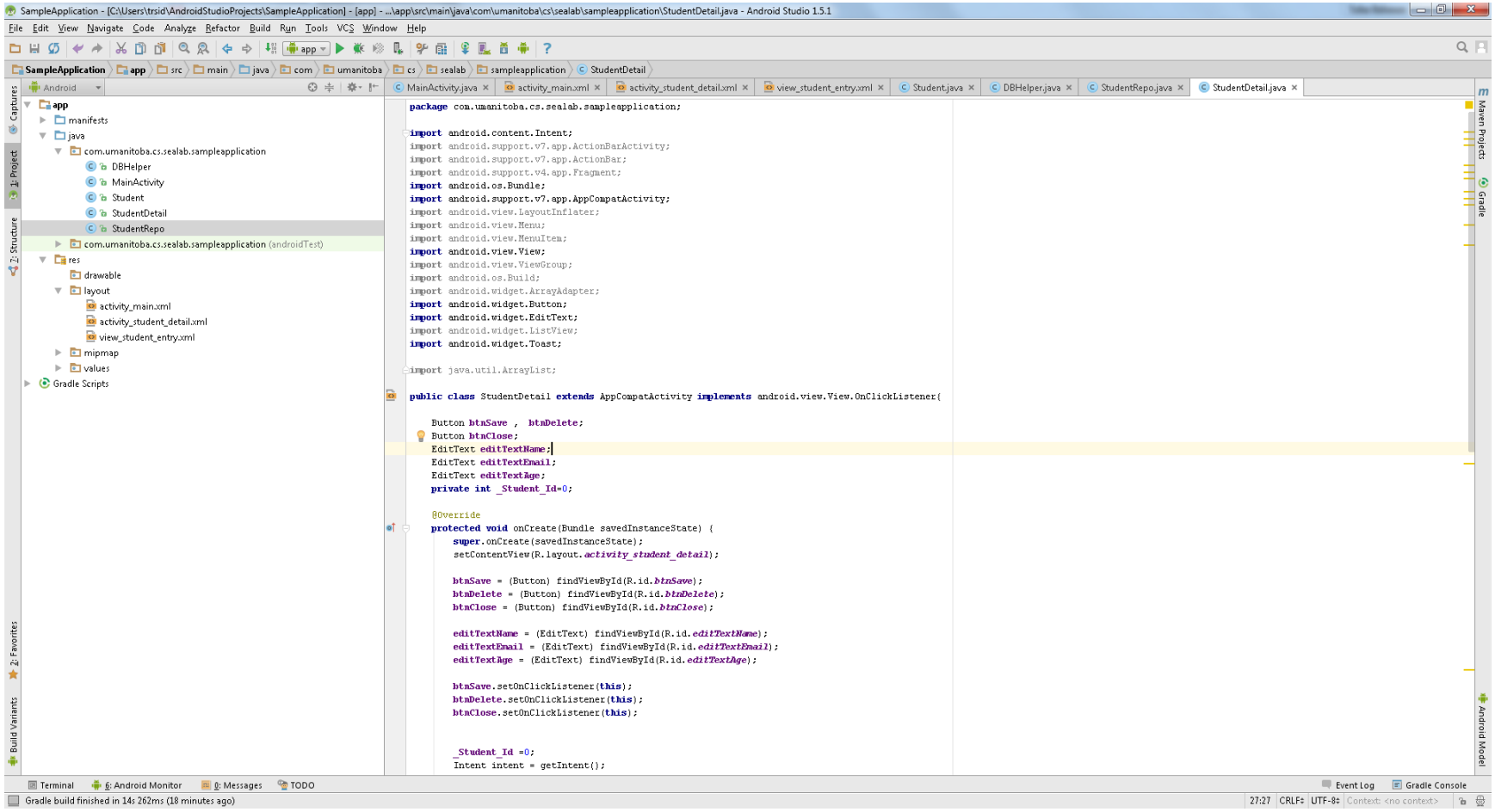


Fig 32. Copy Code 7 to the already created StudentDetail class

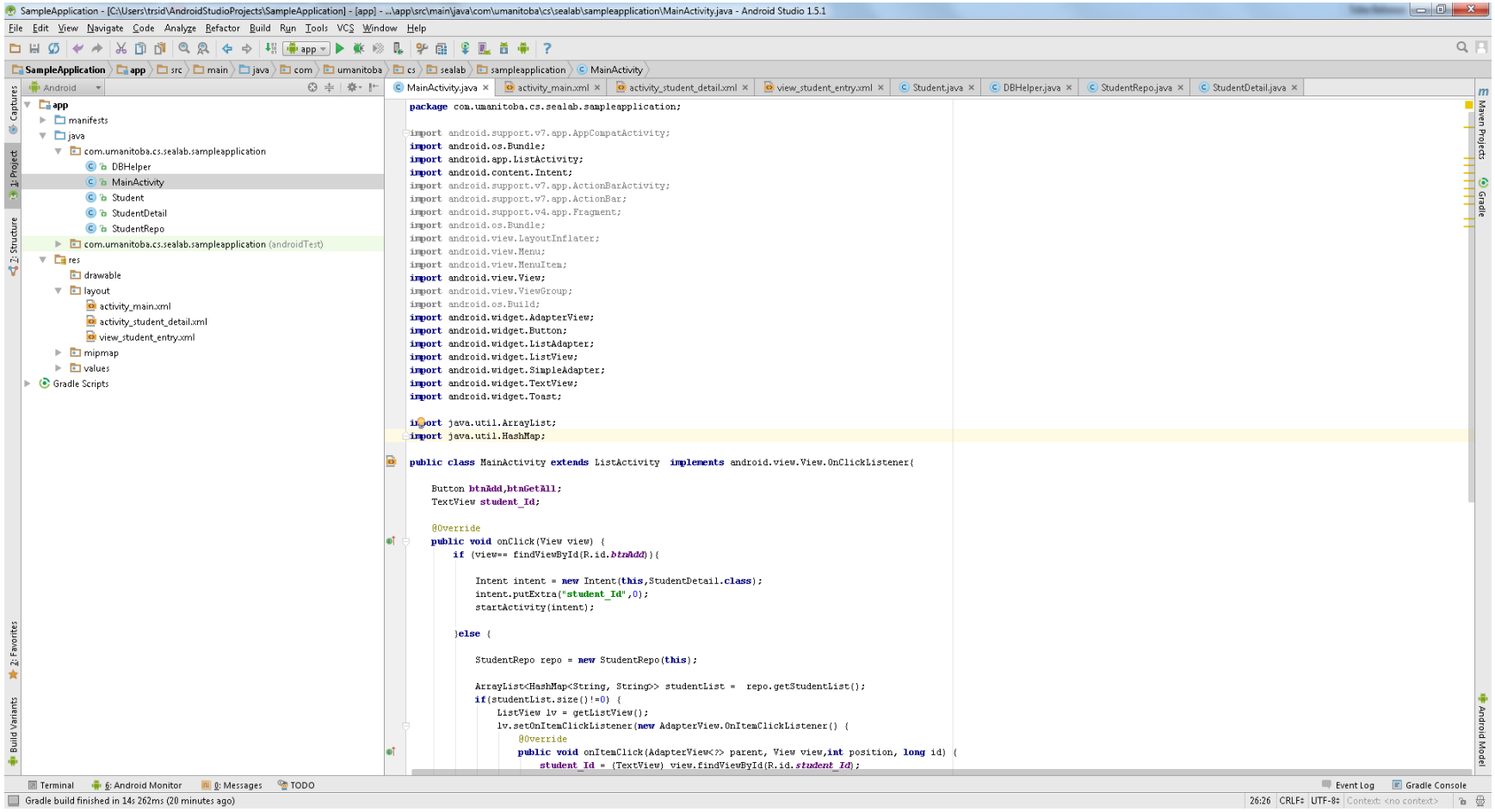


Code 7. StudentDetail.java

```
import android.content.Intent;
import android.support.v7.app.ActionBarActivity;
import android.support.v7.app.ActionBar;
import android.support.v4.app.Fragment;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.os.Build;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;

public class StudentDetail extends AppCompatActivity implements android.view.View.OnClickListener{

    Button btnSave ,  btnDelete;
    Button btnClose;
    EditText editTextName;
    EditText editTextEmail;
    EditText editTextAge;
    private int _Student_Id=0;

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

        btnSave = (Button) findViewById(R.id.btnSave);
        btnDelete = (Button) findViewById(R.id.btnDelete);
        btnClose = (Button) findViewById(R.id.btnClose);

        editTextName = (EditText) findViewById(R.id.editTextName);
        editTextEmail = (EditText) findViewById(R.id.editTextEmail);
        editTextAge = (EditText) findViewById(R.id.editTextAge);

        btnSave.setOnClickListener(this);
        btnDelete.setOnClickListener(this);
        btnClose.setOnClickListener(this);


        _Student_Id =0;
```

```
        Intent intent = getIntent();
        _Student_Id =intent.getIntExtra("student_Id", 0);
        StudentRepo repo = new StudentRepo(this);
        Student student = new Student();
        student = repo.getStudentById(_Student_Id);

        editTextAge.setText(String.valueOf(student.age));
        editTextName.setText(student.name);
        editTextEmail.setText(student.email);
    }



    public void onClick(View view) {
        if (view == findViewById(R.id.btnSave)){
            StudentRepo repo = new StudentRepo(this);
            Student student = new Student();
            student.age= Integer.parseInt(editTextAge.getText().toString());
            student.email=editTextEmail.getText().toString();
            student.name=editTextName.getText().toString();
            student.student_ID=_Student_Id;

            if (_Student_Id==0){
                _Student_Id = repo.insert(student);

                Toast.makeText(this,"New Student Insert",Toast.LENGTH_SHORT).show();
            }else{

                repo.update(student);
                Toast.makeText(this,"Student Record updated",Toast.LENGTH_SHORT).show();
            }
        }else if (view== findViewById(R.id.btnDelete)){
            StudentRepo repo = new StudentRepo(this);
            repo.delete(_Student_Id);
            Toast.makeText(this, "Student Record Deleted", Toast.LENGTH_SHORT);
            finish();
        }else if (view== findViewById(R.id.btnClose)){
            finish();
        }


    }

}
```

Fig 33. Copy Code 8 to MainActivity.java



Code 8. MainActivity.java

```
package com.umanitoba.cs.sealab.sampleapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.app.ListActivity;
import android.content.Intent;
import android.support.v7.app.ActionBarActivity;
import android.support.v7.app.ActionBar;
import android.support.v4.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.os.Build;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.HashMap;

public class MainActivity extends ListActivity  implements android.view.View.OnClickListener{

    Button btnAdd,btnGetAll;
    TextView student_Id;

    @Override
    public void onClick(View view) {
        if (view== findViewById(R.id.btnAdd)){

            Intent intent = new Intent(this,StudentDetail.class);
            intent.putExtra("student_Id",0);
            startActivity(intent);

        }else {

            StudentRepo repo = new StudentRepo(this);
```

```
            ArrayList<HashMap<String, String>> studentList =  repo.getStudentList();
            if(studentList.size()!=0) {
                ListView lv = getListView();
                lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
                    @Override
                    public void onItemClick(AdapterView<?> parent, View view,int position, long id) {
                        student_Id = (TextView) view.findViewById(R.id.student_Id);
                        String studentId = student_Id.getText().toString();
                        Intent objIndent = new Intent(getApplicationContext(),StudentDetail.class);
                        objIndent.putExtra("student_Id", Integer.parseInt( studentId));
                        startActivity(objIndent);
                    }
                });
                ListAdapter adapter = new SimpleAdapter( MainActivity.this,studentList, R.layout.view_student_entry,
new String[] { "id","name"}, new int[] {R.id.student_Id, R.id.student_name});
                setListAdapter(adapter);
            }else{
                Toast.makeText(this,"No student!",Toast.LENGTH_SHORT).show();
            }

        }
    }

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

        btnAdd = (Button) findViewById(R.id.btnAdd);
        btnAdd.setOnClickListener(this);

        btnGetAll = (Button) findViewById(R.id.btnGetAll);
        btnGetAll.setOnClickListener(this);

    }


}
```

Section 3

Setting up Emulator and running the app

Fig 34. We are done with the application. Now downloading a faster emulator for testing our app. Google “Download Genymotion”

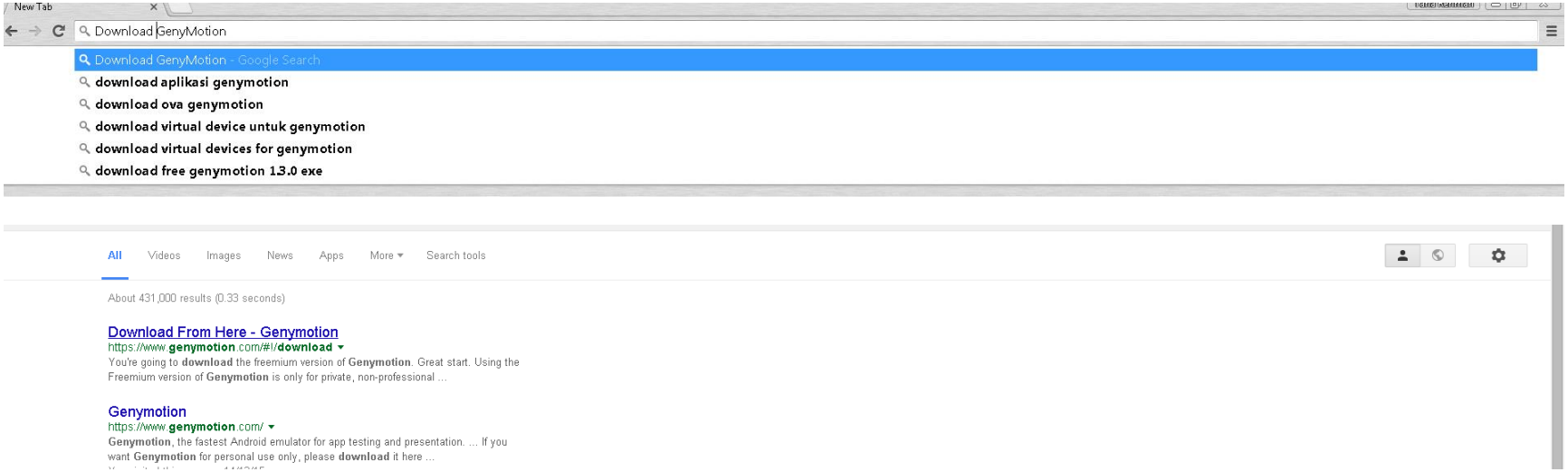


Fig 35. Download Genymotion for free (Personal Use)

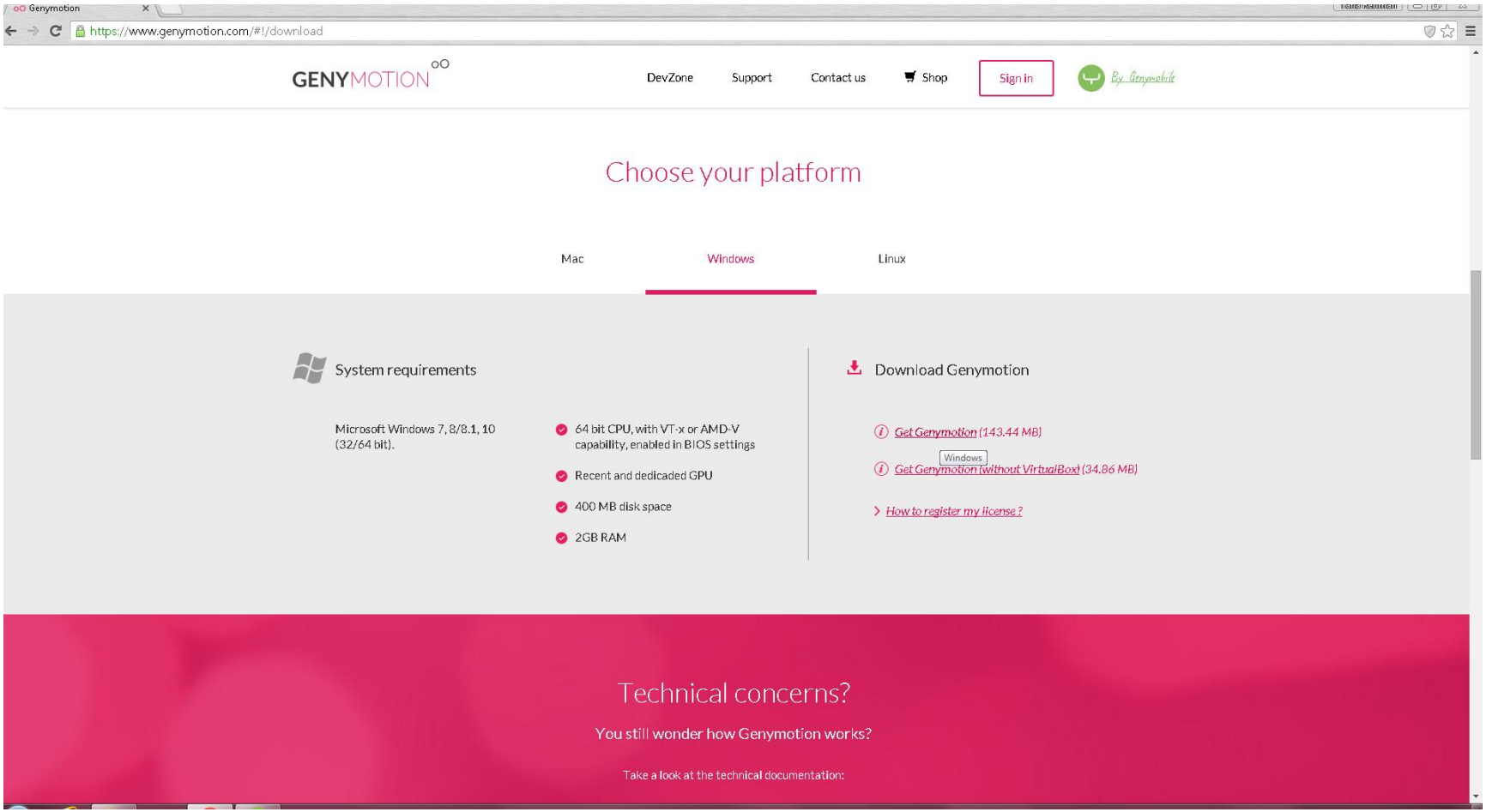
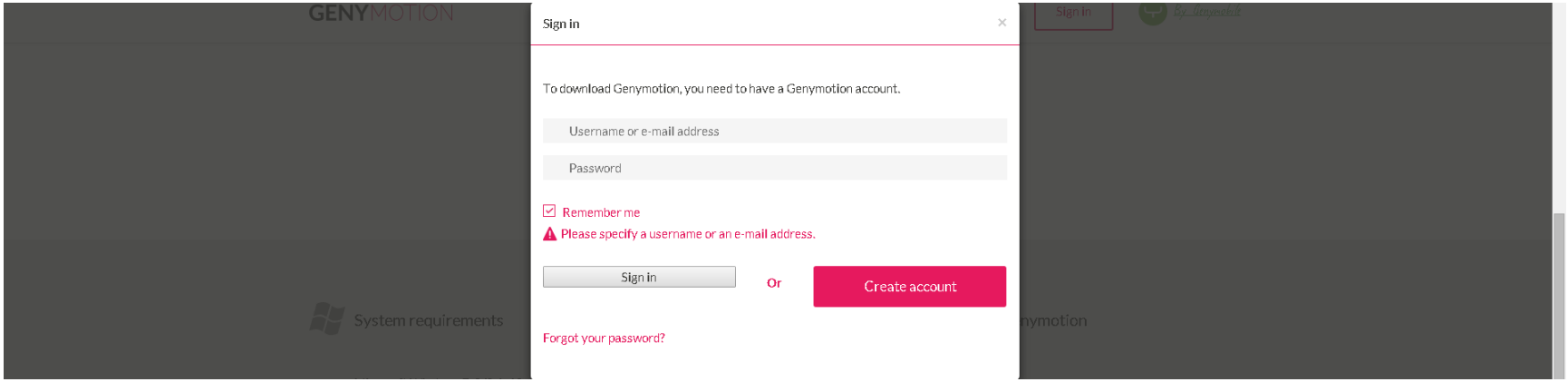


Fig 36. On login screen, login or create a new account. Remember the credentials as we are going to need them later



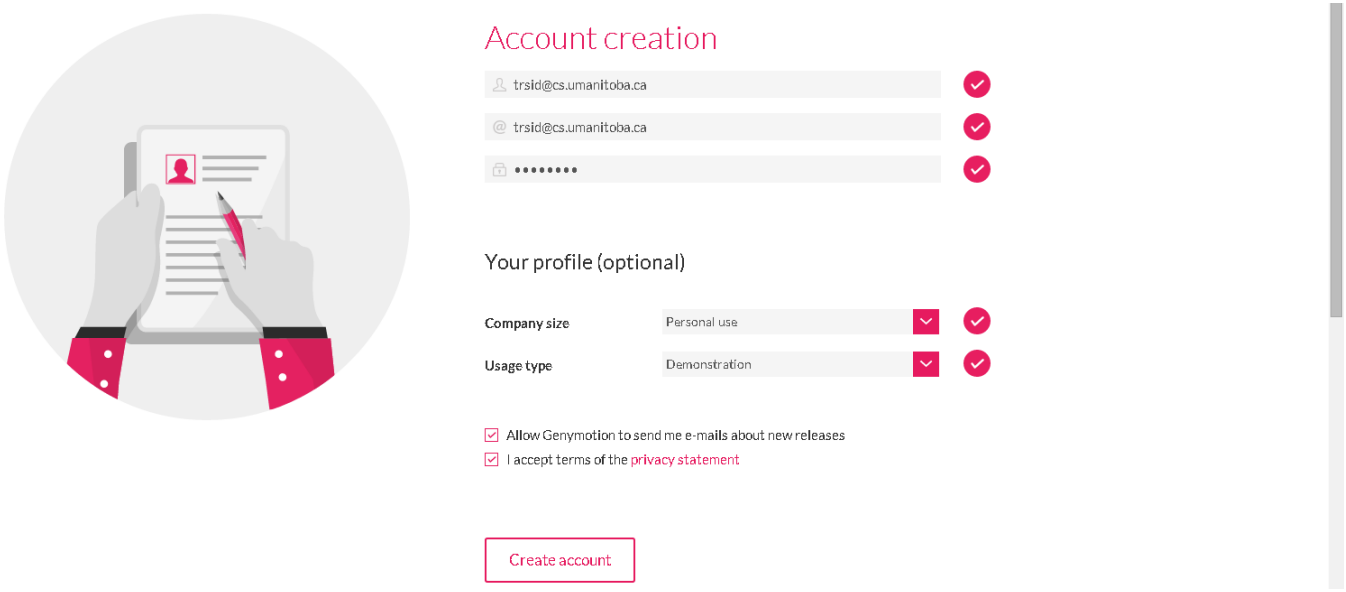


Fig 37. Download after logging in

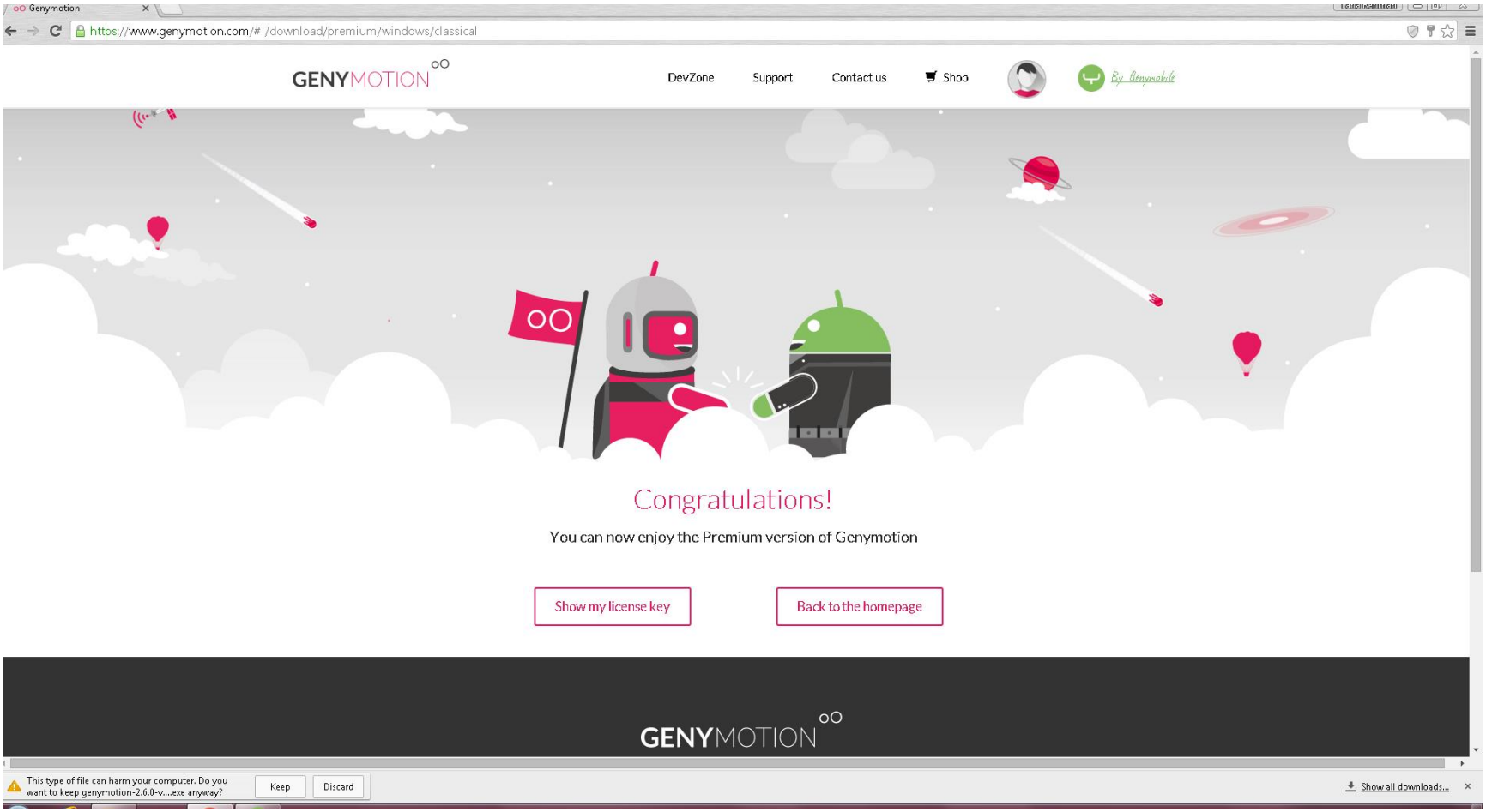


Fig 38. Run the setup. It needs VirtualBox® to run, so it will install that as well. Keep pressing Next or Install as shown below.

Fig 39. Done with the installation. Press finish to launch Genymotion only. After accepting the usage notice add a new virtual device (auto popup) using the same credentials that we used to download Genymotion.

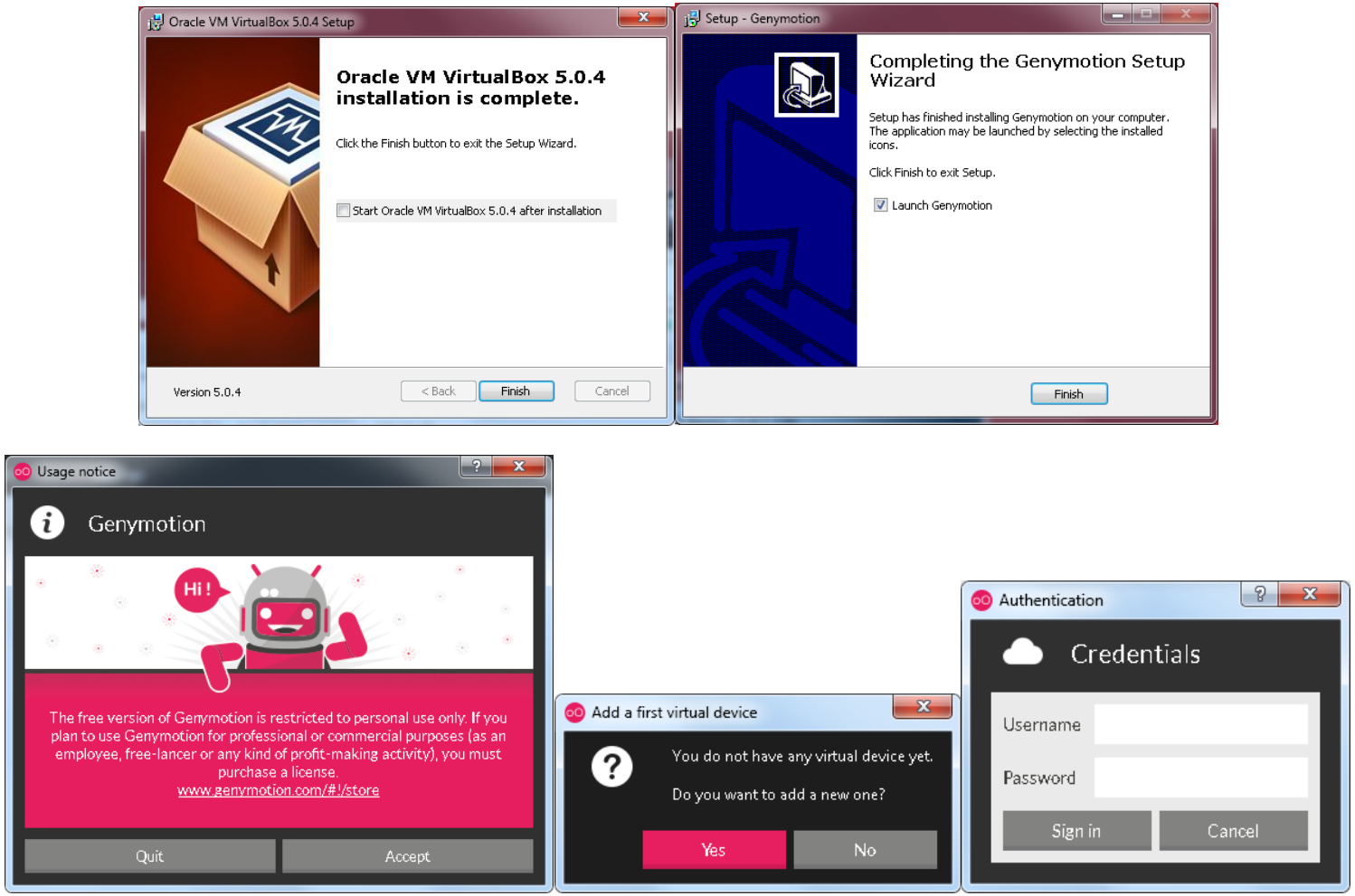


Fig 40. Add a new virtual device; Nexus 7. Lower the resolution of the device, (slightly) better the performance. After finishing, start the device from the application menu.

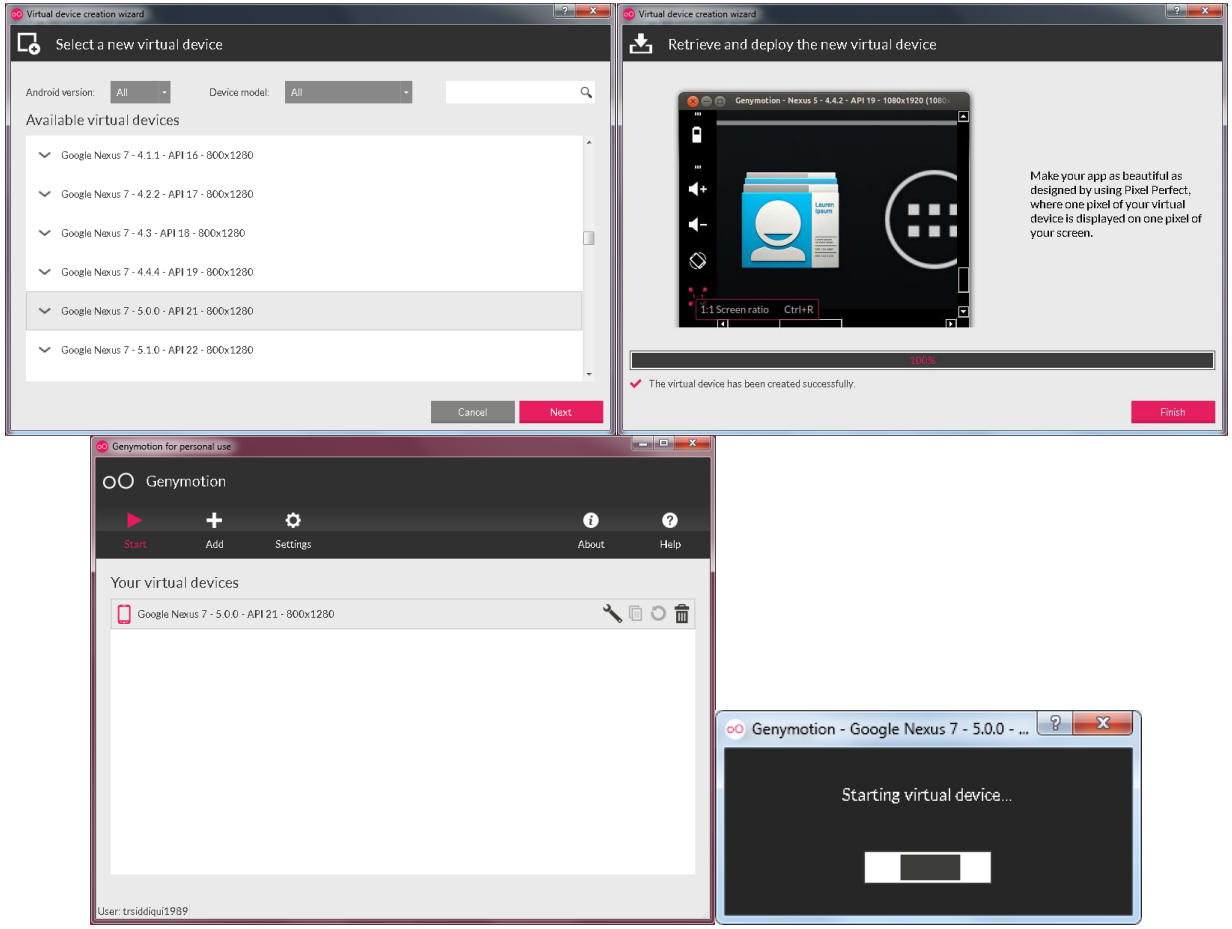


Fig 41. This is what our new device looks like. Once it finishes loading, it will work like any other android phone/tablet.

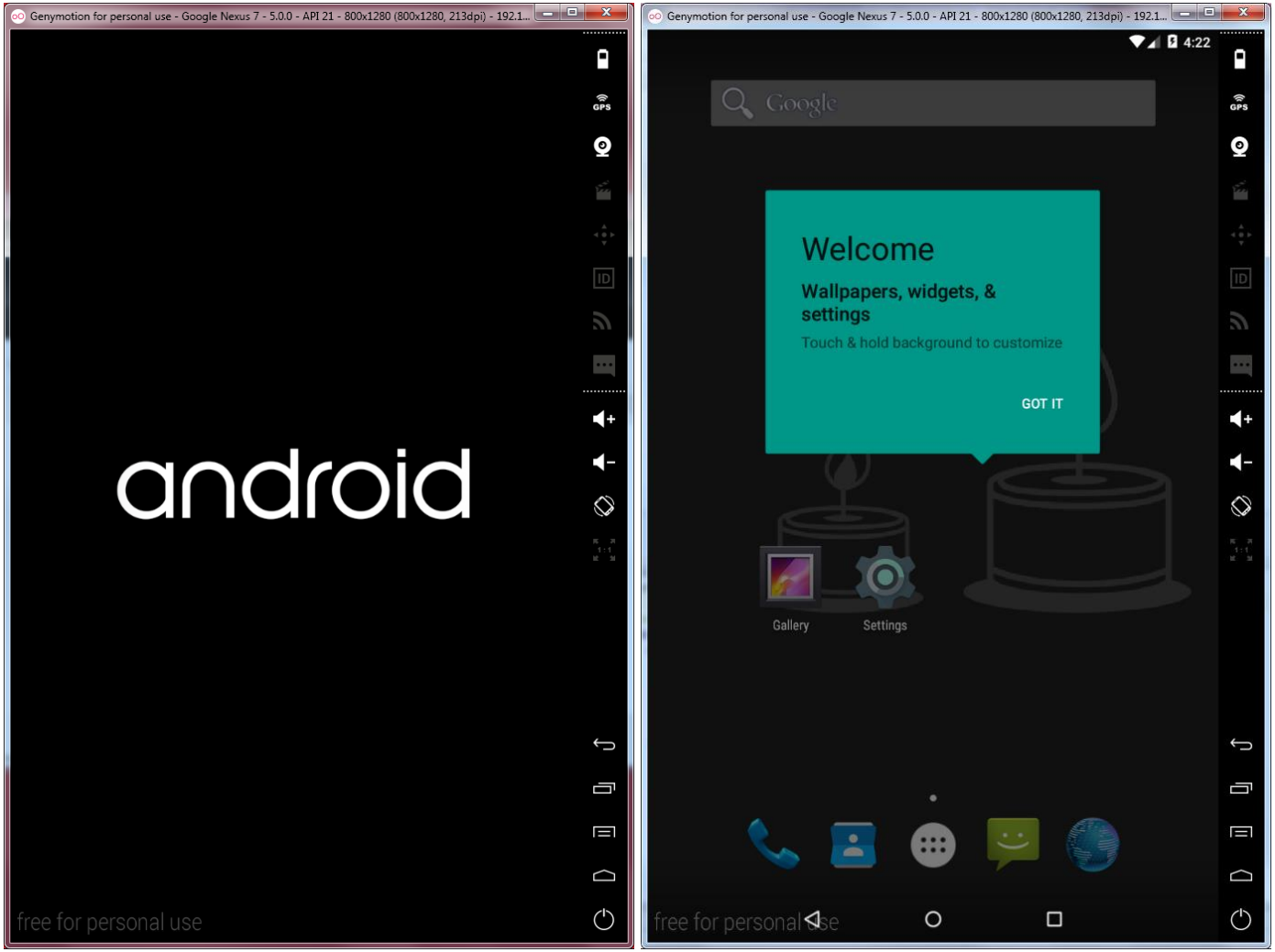


Fig 42. Coming back to Android Studio®. Go to menu Build and click on Build APK.

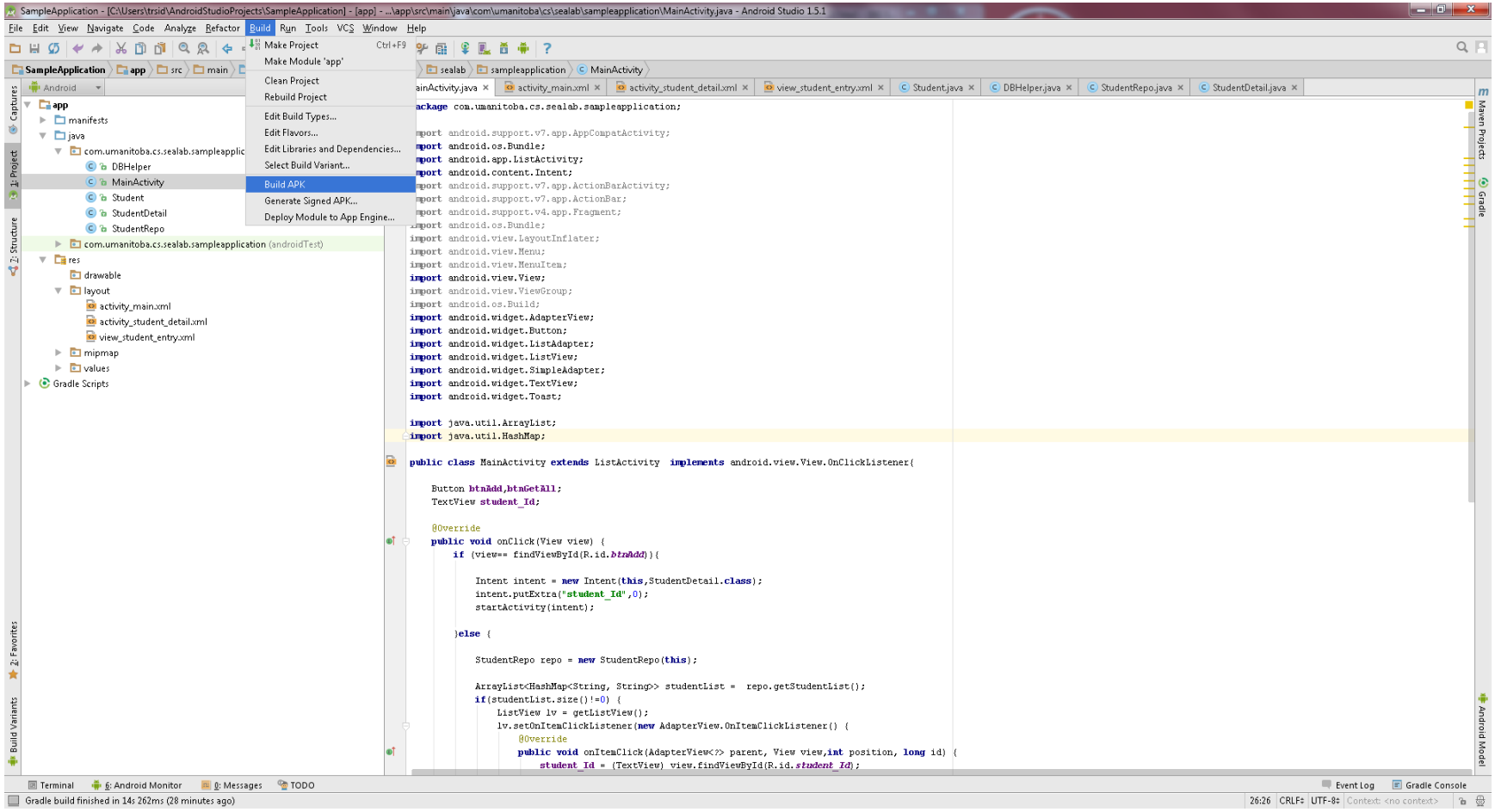
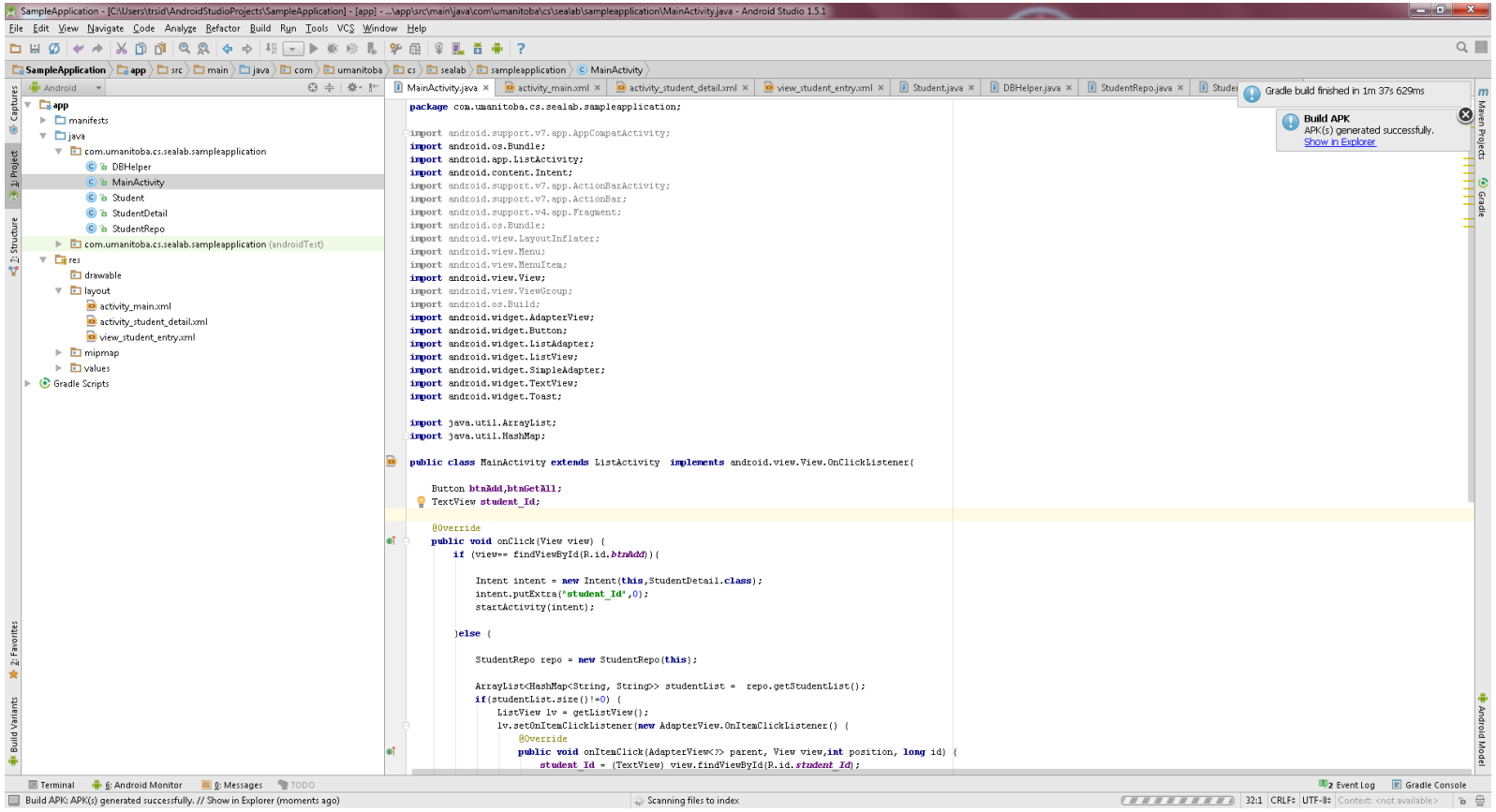


Fig 43. After building, it will show a link to the APK file generated. Click on the link on the top right of the screen to open the folder.



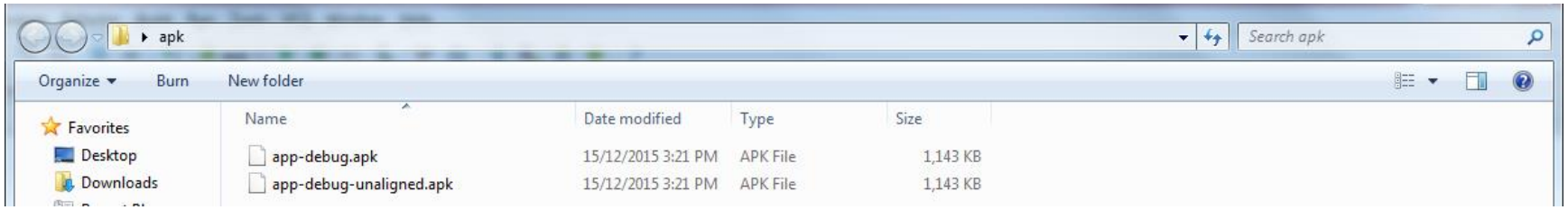


Fig 44. Now drag the APK file and drop it to our emulator Genymotion®. It will show a "File Transfer" window and open the application.

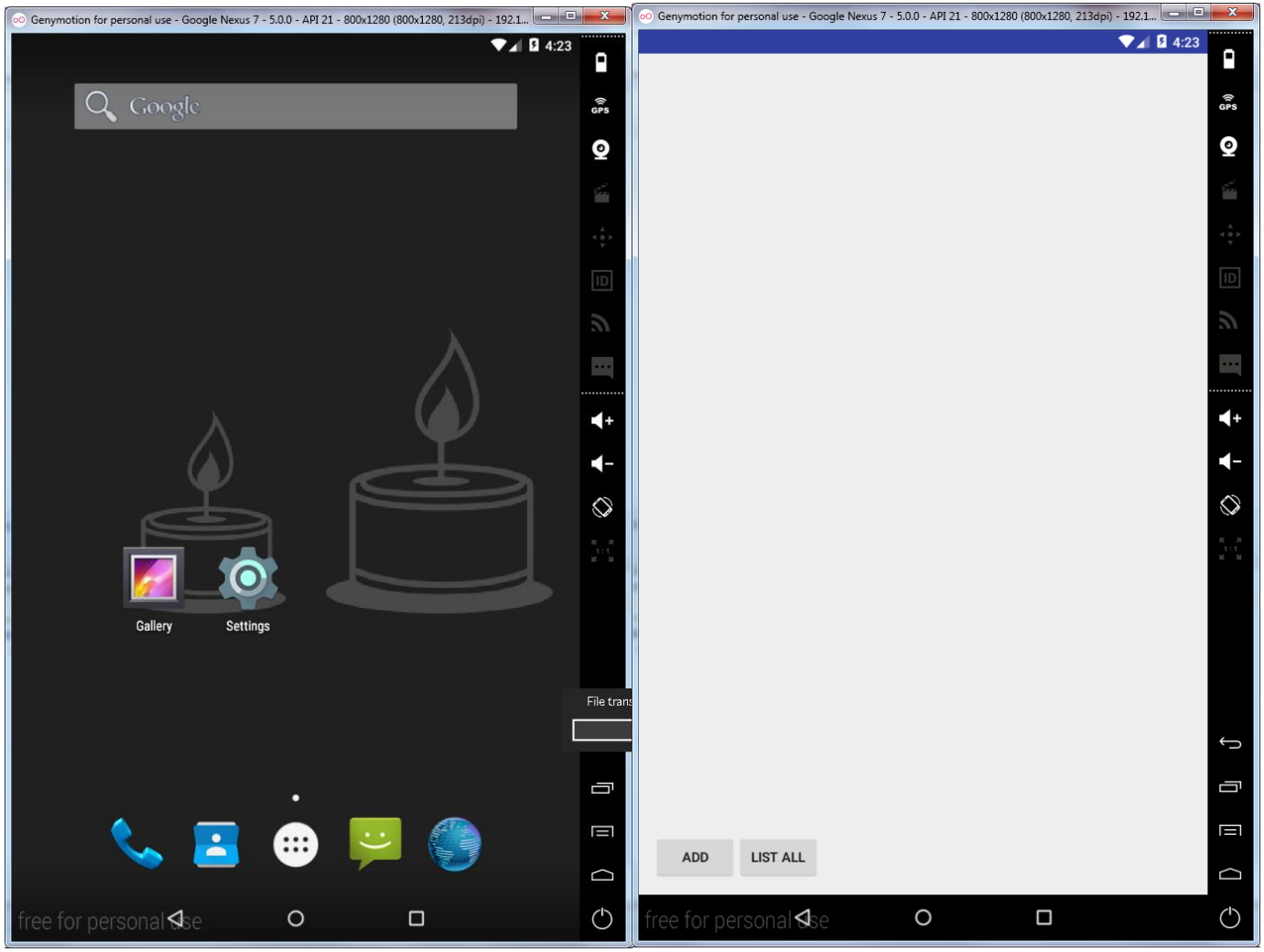
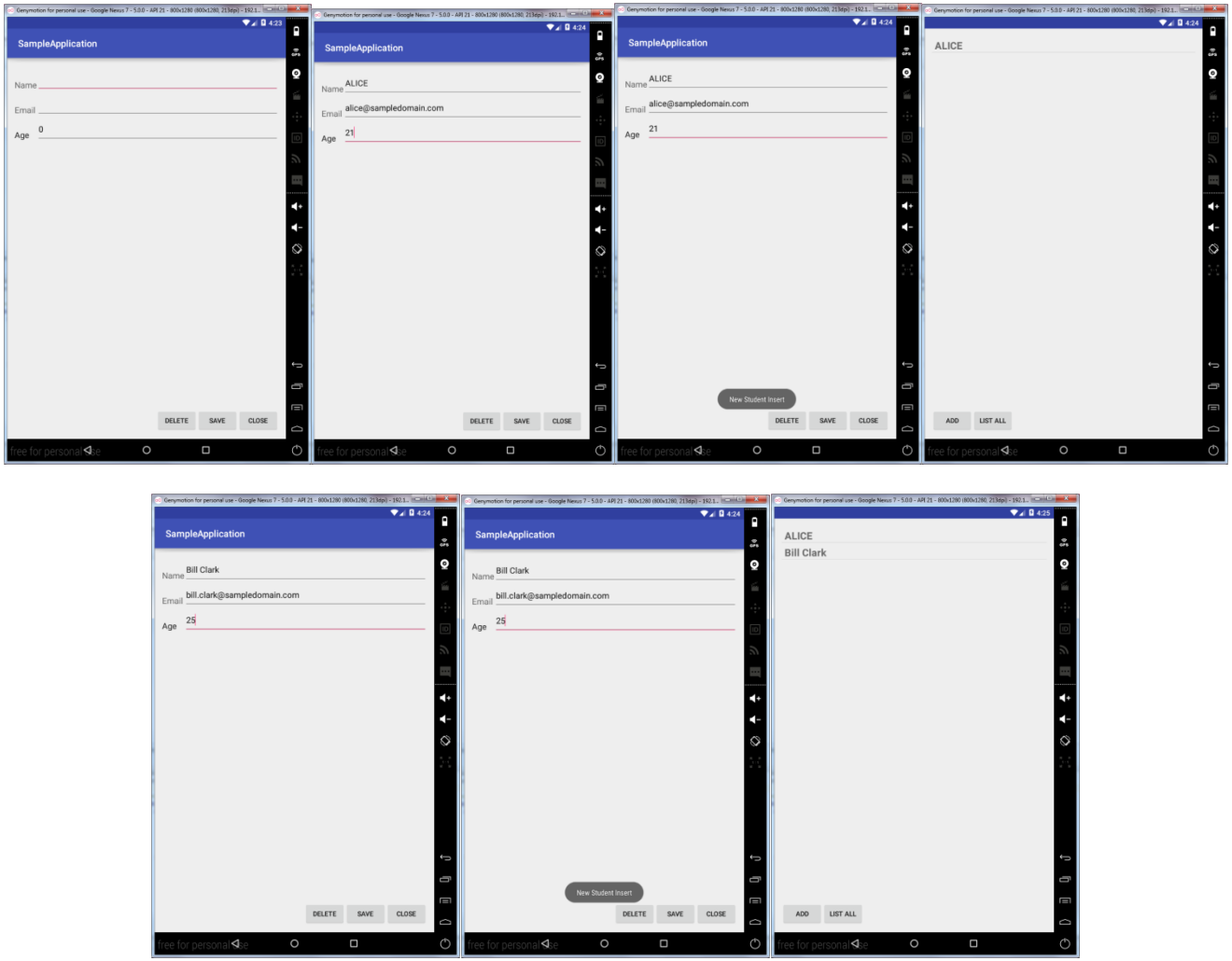


Fig 45. Test our application. Add a new record, Save it. Tap on "List All" for updating result (to query latest from the database).



Section 4

Validating the changes to the DB on device

Fig 46. Download SQLite Browser for Eclipse from the link [Android SQLite Browser for Eclipse]

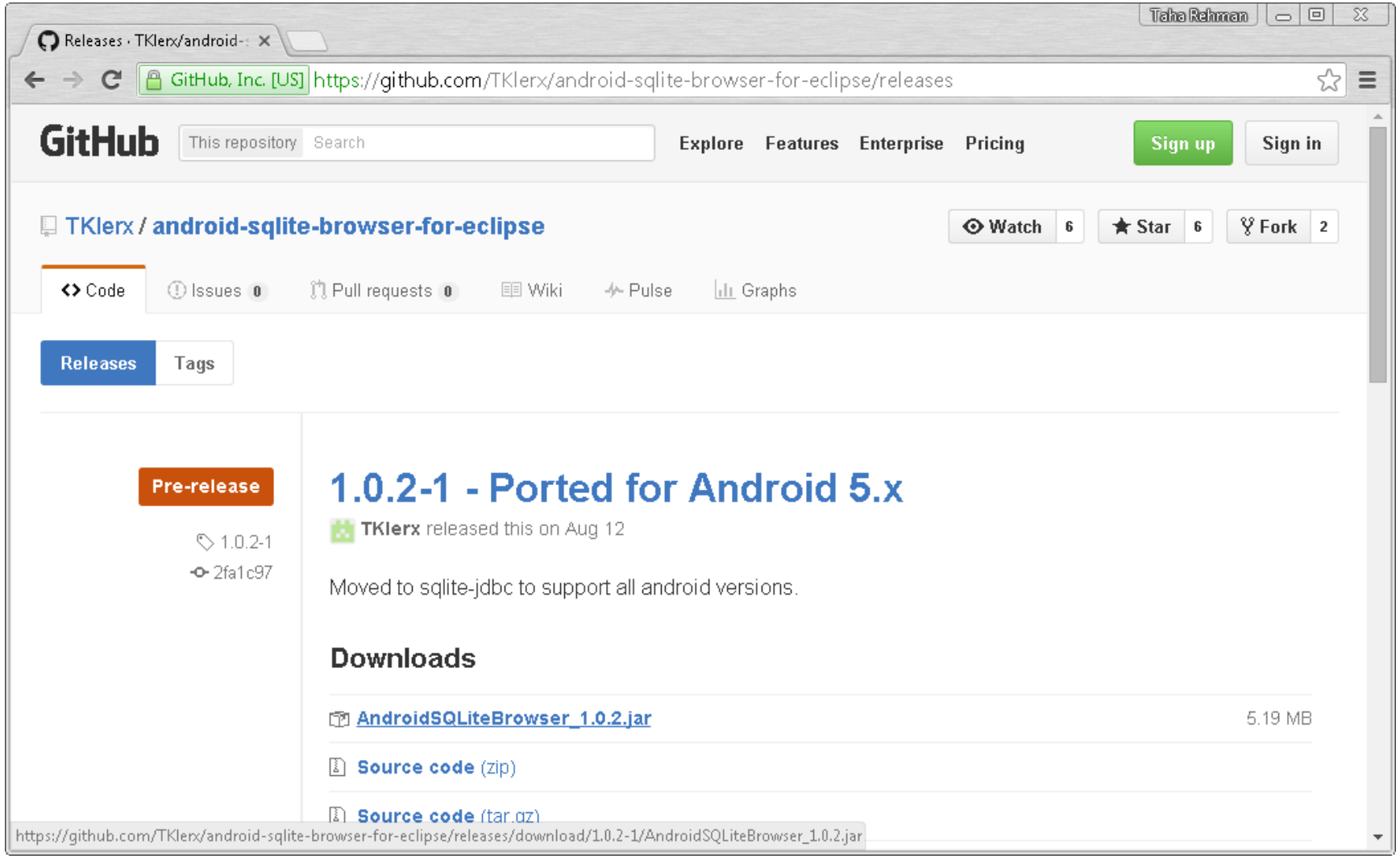


Fig 47. Find the path of Eclipse plugin directory by Opening the below Section in Settings (File → Settings) and copy by selecting it manually and pressing Ctrl + C. Now open the folder manually using File Explorer and browse to <PATH> → tools → lib → monitor-x86 → plugins

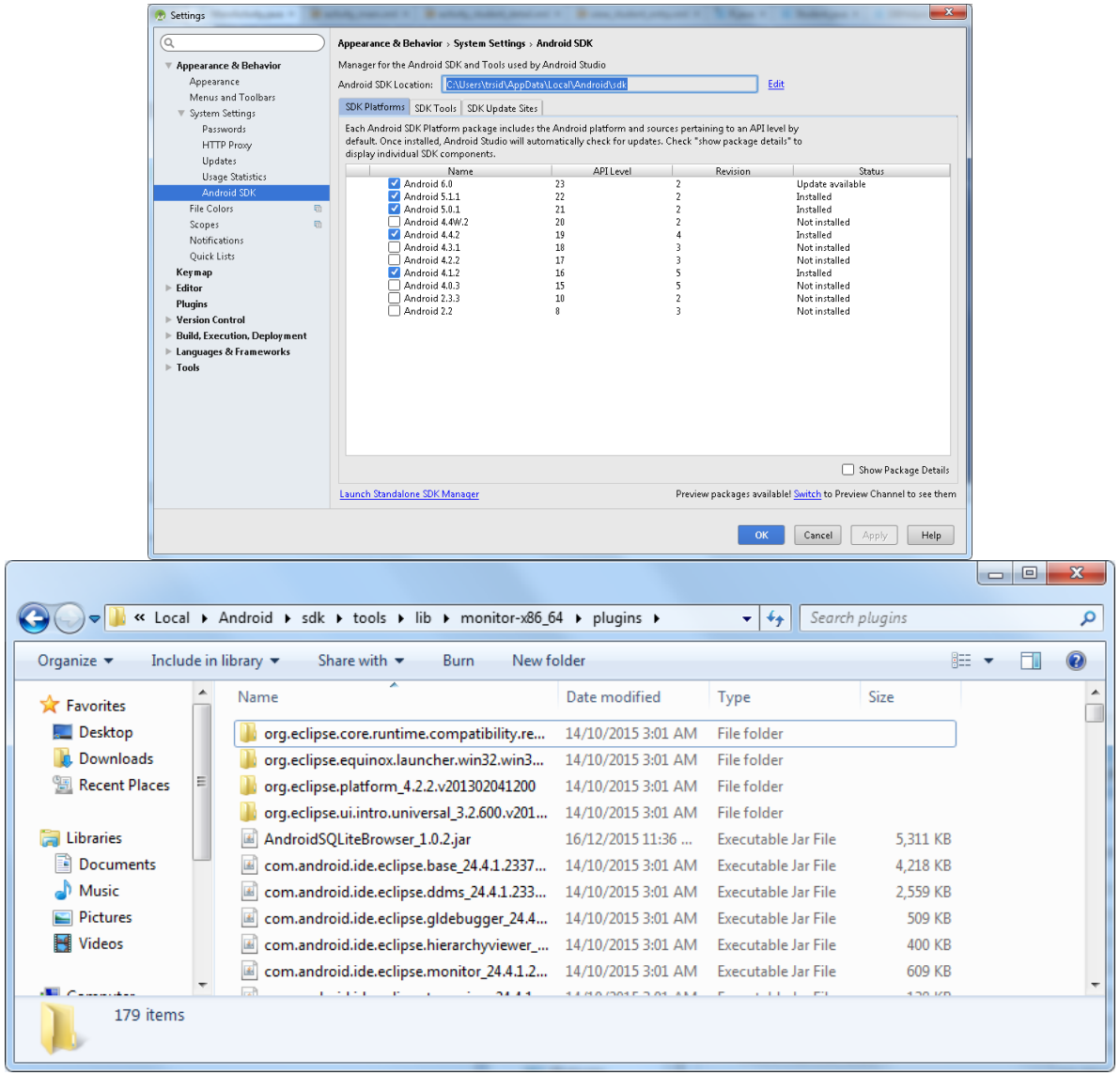


Fig 48. Copy the downloaded file and paste to above folder.

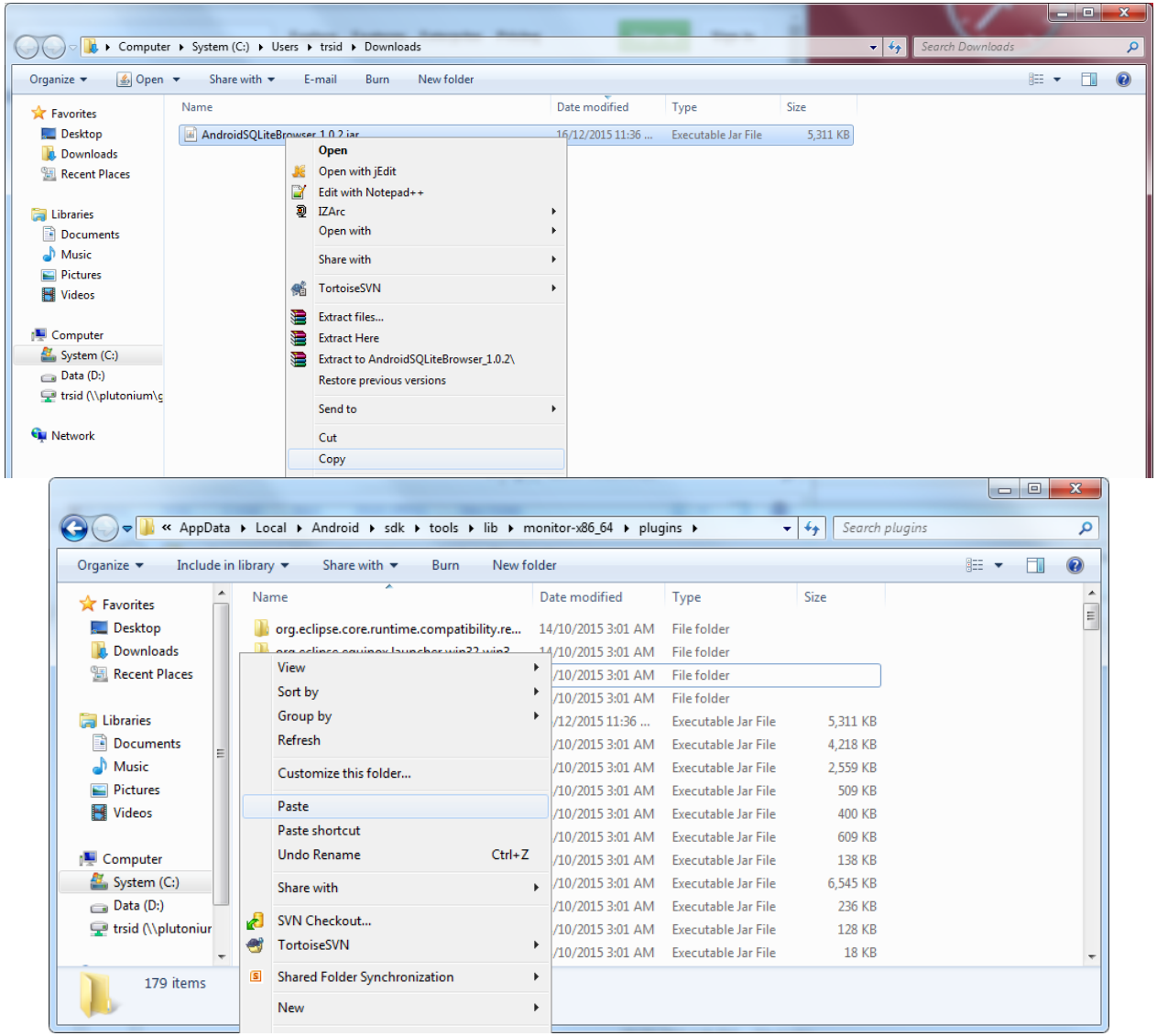


Fig 49. Keep the app running in Genymotion® and Open Android Device Monitor by clicking on the following button in the toolbar.

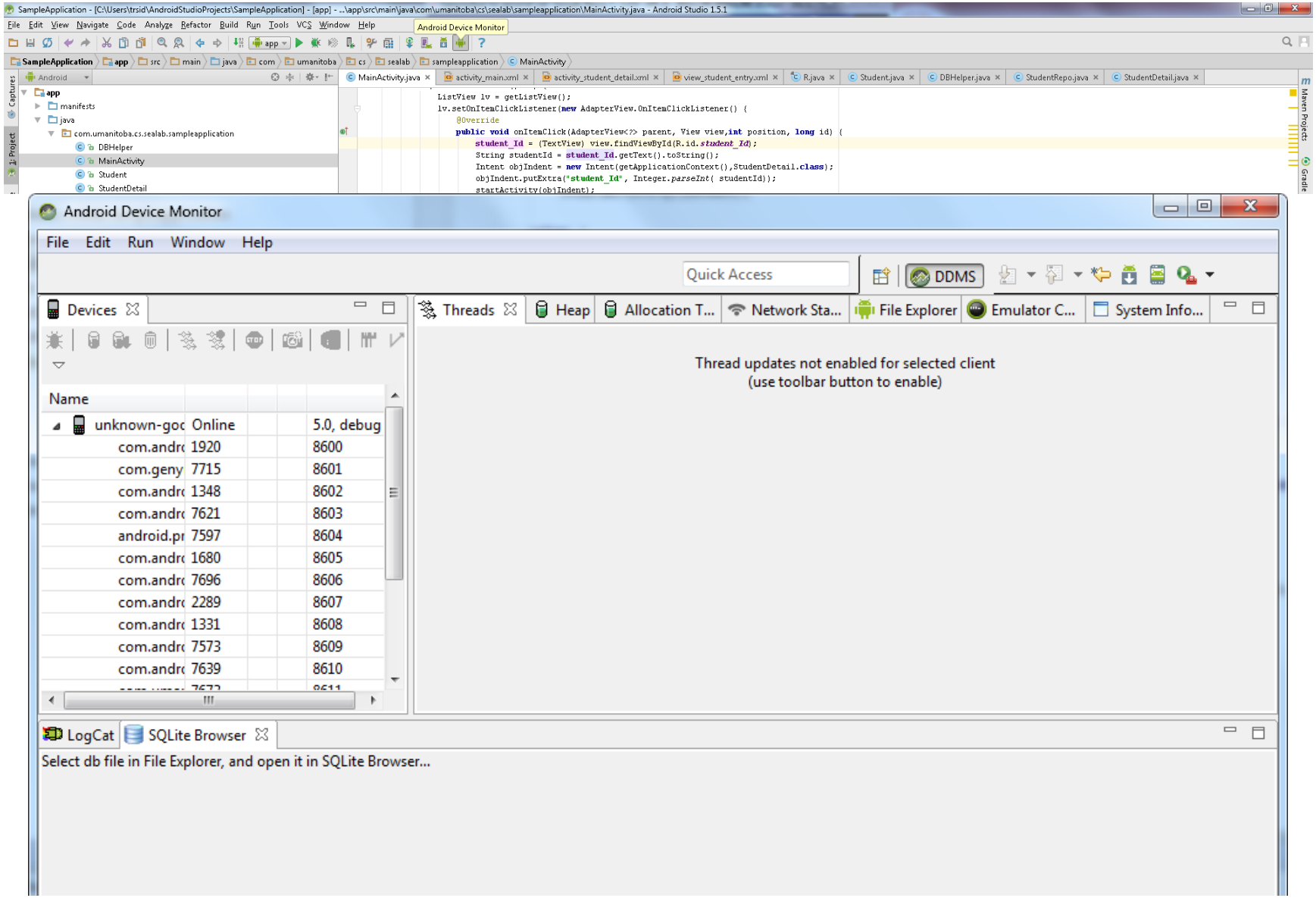


Fig 50. Open File Explorer, navigate to root → data → data → <applicationName>(com.umanitoba.cs.sealab.sampleapplication) → databases and single left click on the database file (crud.db)

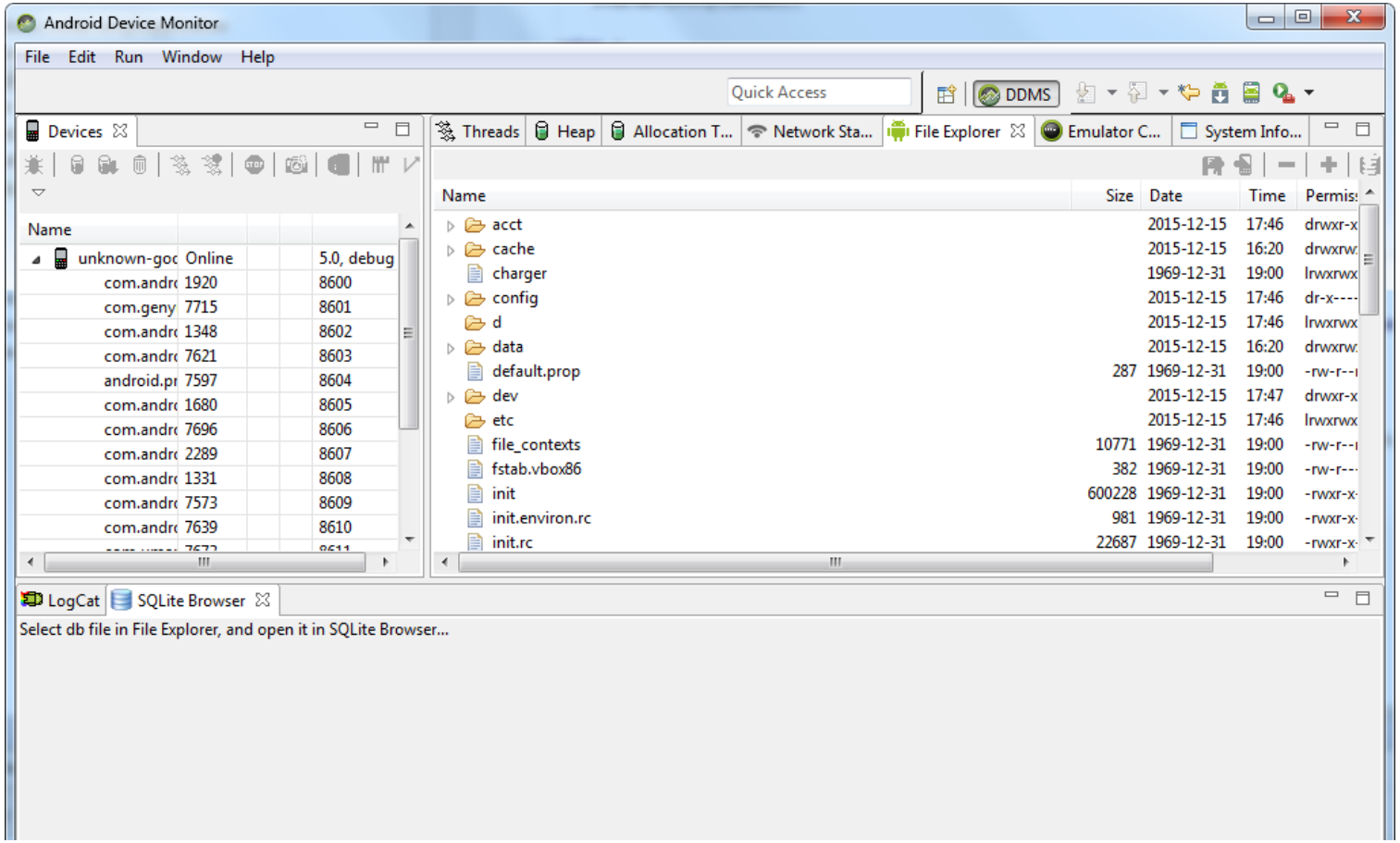


Fig 52. When you click on the database file, SQLite Browser plugin button will enable on the top right section of the Device Monitor. Click on the button to show the content of the database in the SQLite Browser tab in the bottom section of the window.

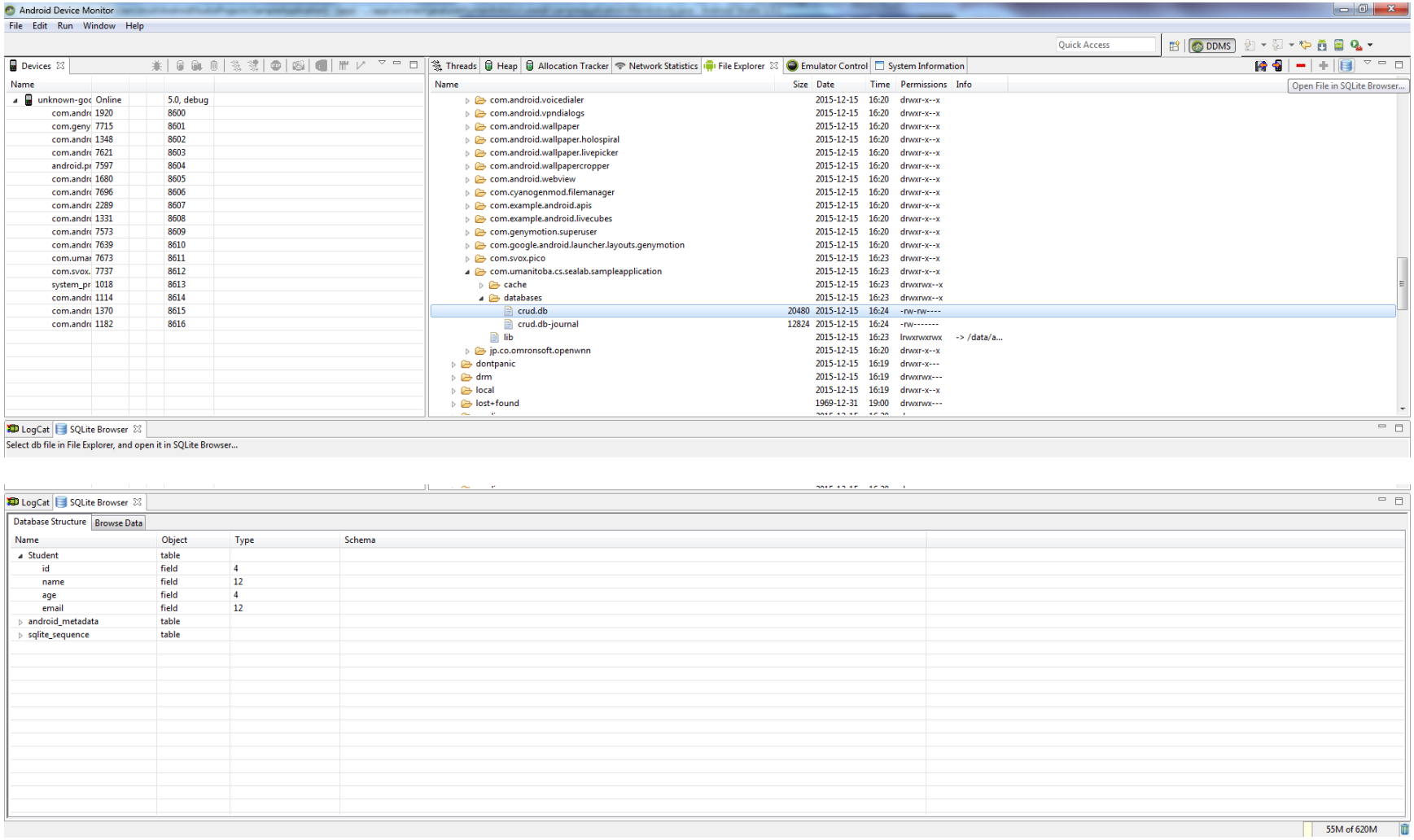


Fig 53. Click on the Browse Data tab and see the data you inserted in the application.