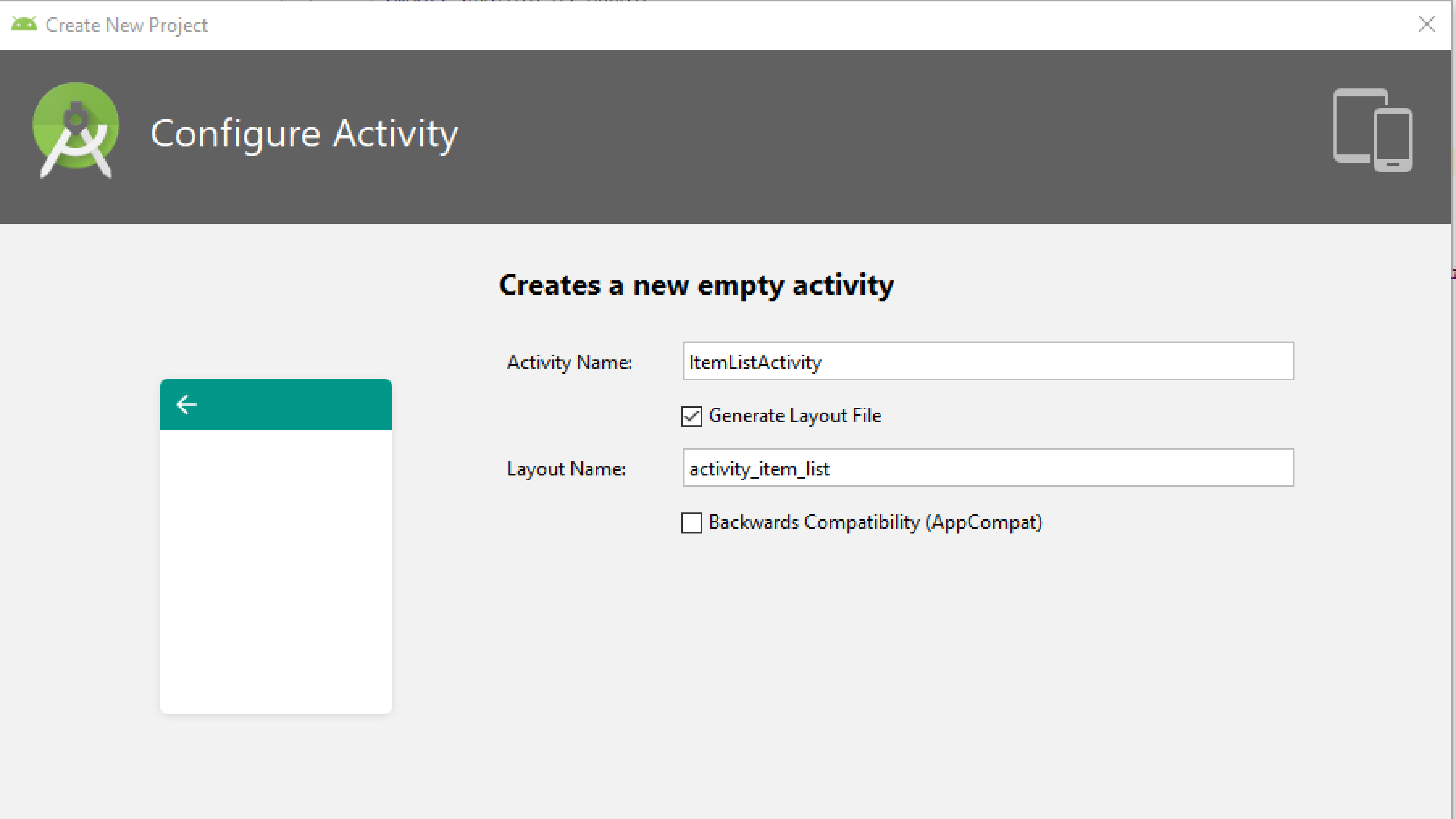
RecyclerView & MVVM tutorial

The final result of this tutorial is on github:

https://github.com/sferhah/Android-MVVM-Tutorial



Configure Activity

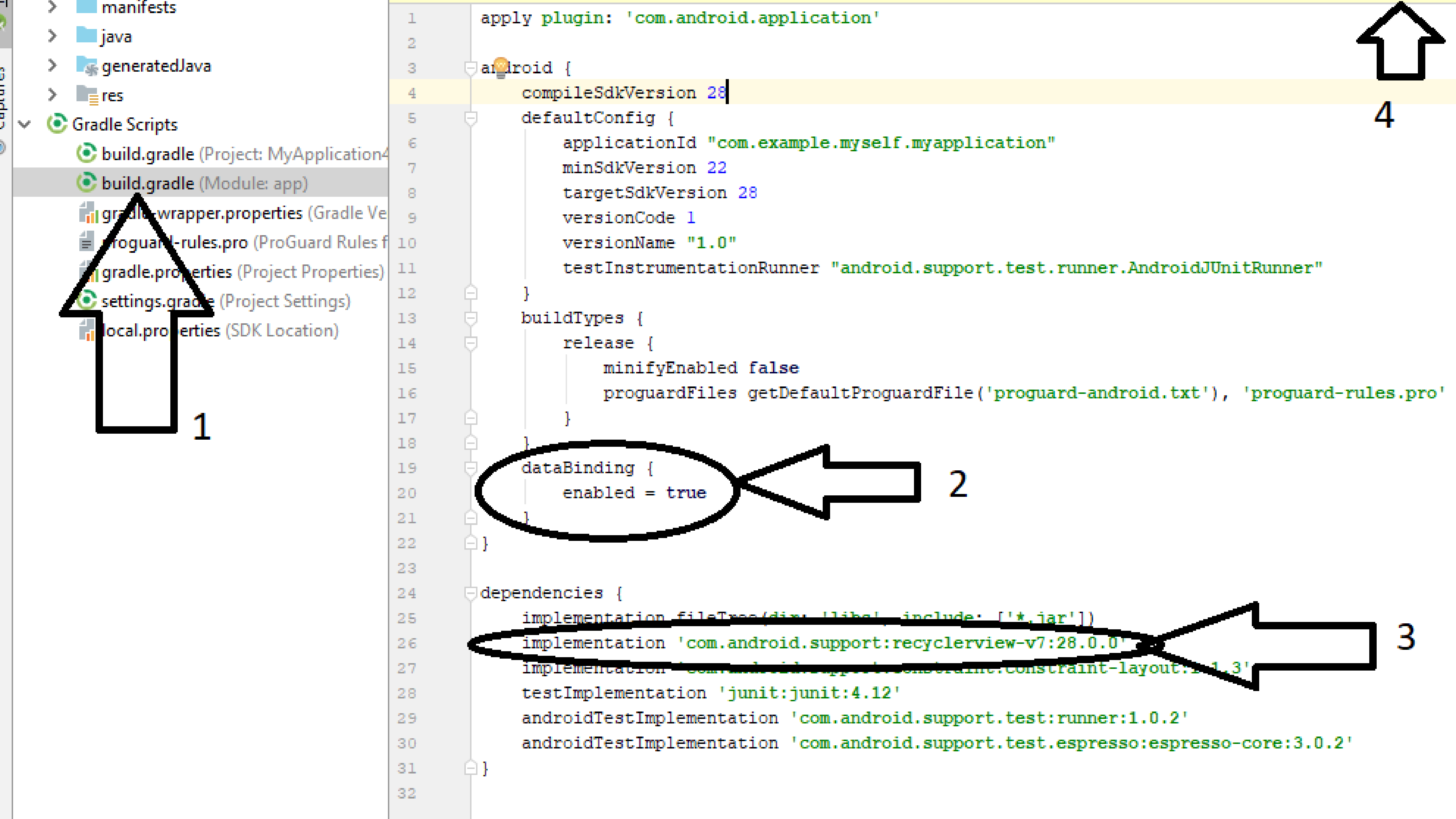
Creates a new empty activity

Activity Name: ItemListActivity

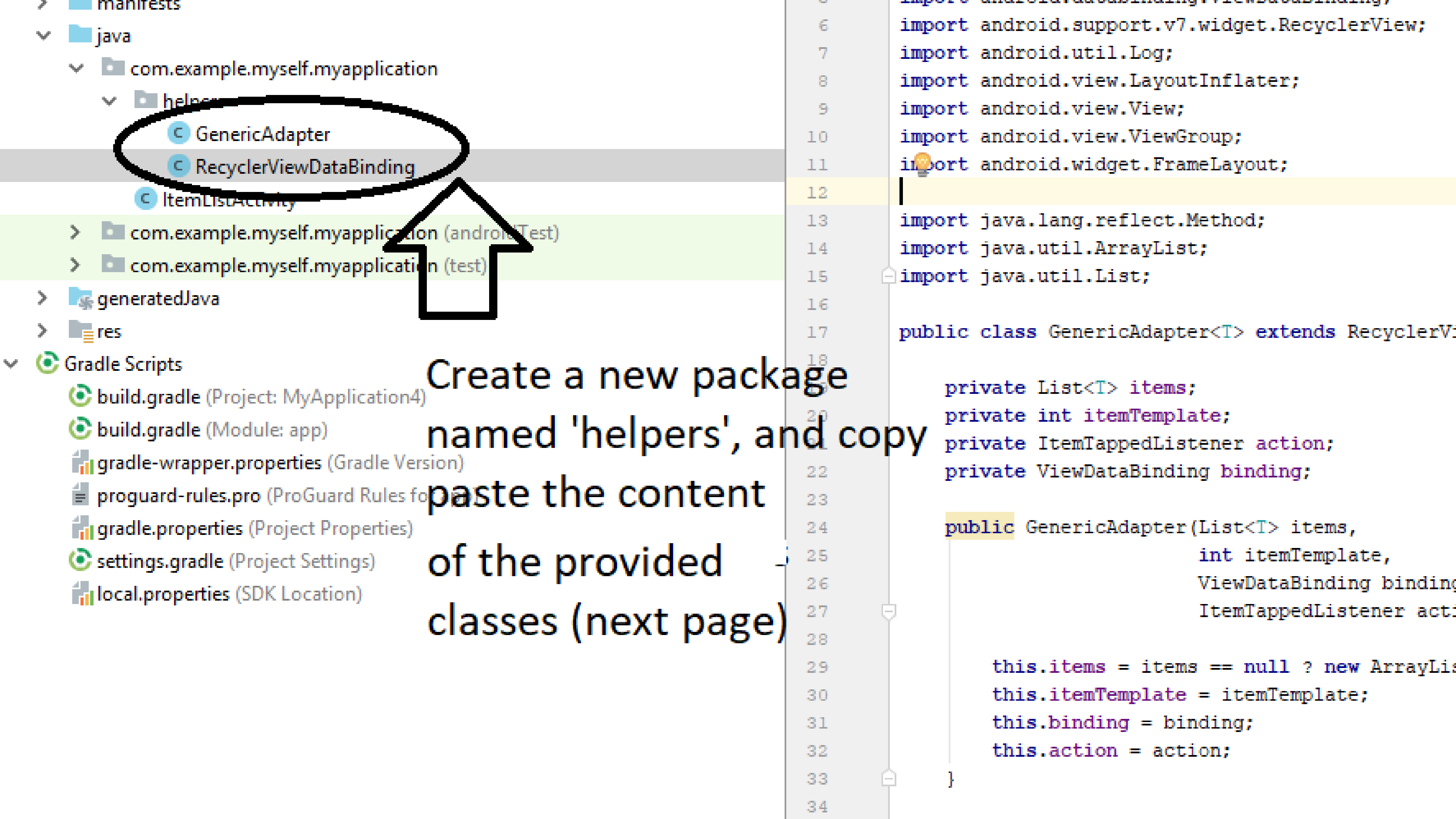
☑ Generate Layout File

Layout Name: activity_item_list

□ Backwards Compatibility (AppCompat)


manifests
java
generatedJava
res
Gradle Scripts
build.gradle (Project: MyApplication4
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Ve
proguard-rules.pro (ProGuard Rules f
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
1
2
3
4

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "com.example.myself.myapplication"
        minSdkVersion 22
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
    dataBinding {
        enabled = true
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:recyclerview-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
}
```



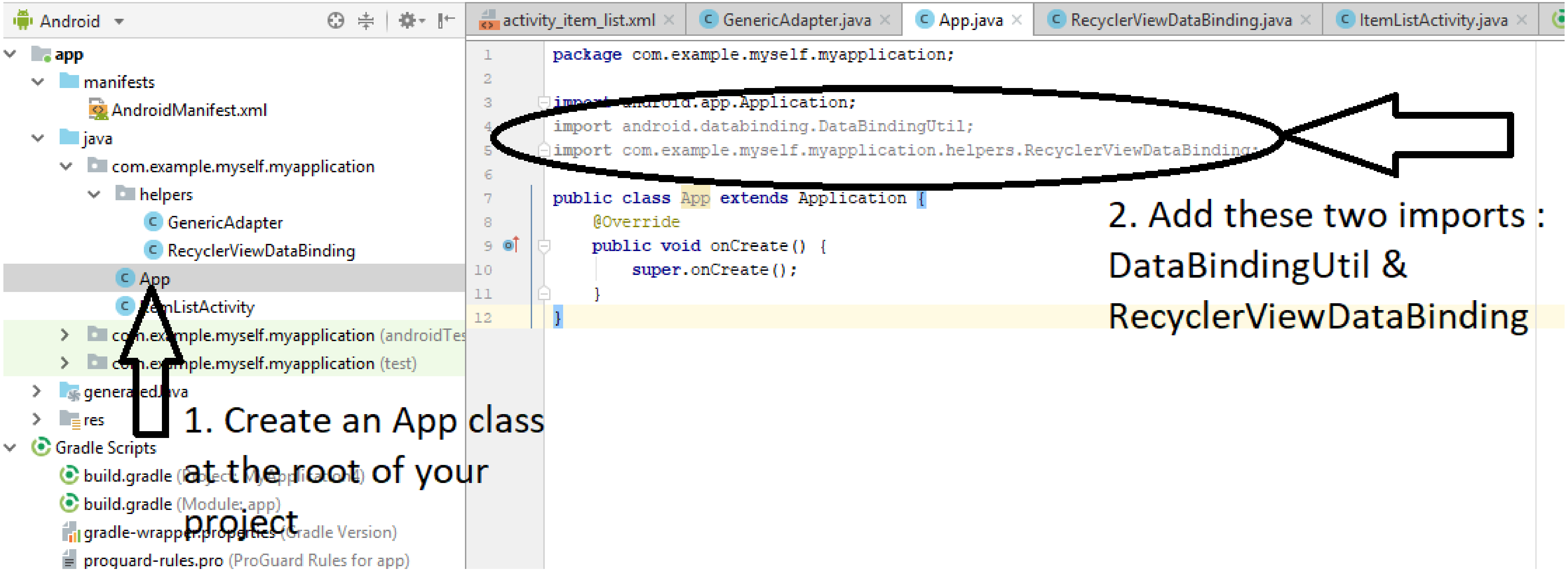
manifests
java
com.example.myself.myapplication
helpers
GenericAdapter
RecyclerViewDataBinding
ItemListActivity
com.example.myself.myapplication (androidTest)
com.example.myself.myapplication (test)
generatedJava
res
Gradle Scripts
build.gradle (Project: MyApplication4)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules fo
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)

```
import android.databinding.ViewDataBinding;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.FrameLayout;

import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.List;

public class GenericAdapter<T> extends RecyclerVi

    private List<T> items;
    private int itemTemplate;
    private ItemTappedListener action;
    private ViewDataBinding binding;

    public GenericAdapter(List<T> items,
                          int itemTemplate,
                          ViewDataBinding binding
                          ItemTappedListener acti

        this.items = items == null ? new ArrayLis
        this.itemTemplate = itemTemplate;
        this.binding = binding;
        this.action = action;
    }
```

Create a new package named 'helpers', and copy paste the content of the provided classes (next page)

```
public class GenericAdapter<T> extends RecyclerView.Adapter<GenericAdapter.DataViewHolder<T>> {

   private List<T> items;
   private int itemTemplate;
   private ItemTappedListener action;
   private ViewDataBinding binding;

   public GenericAdapter(List<T> items,  int itemTemplate, ViewDataBinding binding, ItemTappedListener action) {

      this.items = items == null ? new ArrayList<T>() : items; this.itemTemplate = itemTemplate; this.binding = binding; this.action = action;
  }

   @Override
   public DataViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
      final DataViewHolder holder = new DataViewHolder(LayoutInflater.from(parent.getContext()).inflate(itemTemplate, new FrameLayout(parent.getContext()), false));
      holder.itemView.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            T item = items.get(holder.getLayoutPosition());

            if(item == null) {

               Log.i("Logger", "Item is null");
               return;
            }

            Class<?> type = item.getClass();
            setItem(binding, type, item);
            action.onTapped(null);
            setItem(binding, type,null);
         }
      });
      return holder;
   }

   @Override
   public void onBindViewHolder(DataViewHolder holder, int position) {    holder.setViewModel(items.get(position));   }

   @Override
   public int getItemCount() {   return this.items.size();   }

   static void setItem(Object binding, Class<?> type, Object item) {

      if (binding == null) return;
      Method setMethod = null;
      for (Method m: binding.getClass().getMethods()) {

         Class[] paramTypes  = m.getParameterTypes();

         if(paramTypes.length == 1
               && paramTypes[0] == type){

            setMethod = m;
            break;
         }
      }

      if(setMethod != null) {  try { setMethod.invoke(binding, item); } catch (Exception e) {}}
   }


   static class DataViewHolder<T> extends RecyclerView.ViewHolder {
      ViewDataBinding binding;
      DataViewHolder(View itemView) {
         super(itemView);
         binding = DataBindingUtil.bind(itemView);
      }

      void setViewModel(T viewModel) {
         setItem(binding, viewModel.getClass(), viewModel);
      }
   }

   public interface ItemTappedListener {
      void onTapped(Object item);
   }
}
```

```
public class RecyclerViewDataBinding<T> {

   @BindingAdapter({"app:itemsSource", 
"app:itemTemplate", "app:onItemTapped"})
   public void bind(RecyclerView recyclerView, 
List<T> items, int itemTemplate, 
GenericAdapter.ItemTappedListener 
onItemTapped) {
       recyclerView.setAdapter(new 
GenericAdapter(items, itemTemplate, 
DataBindingUtil.findBinding(recyclerView), 
onItemTapped));
   }

   @BindingAdapter({"app:onItemTapped"})
   public void bind(RecyclerView recyclerView, 
GenericAdapter.ItemTappedListener 
itemTapped) {
   }
}
```

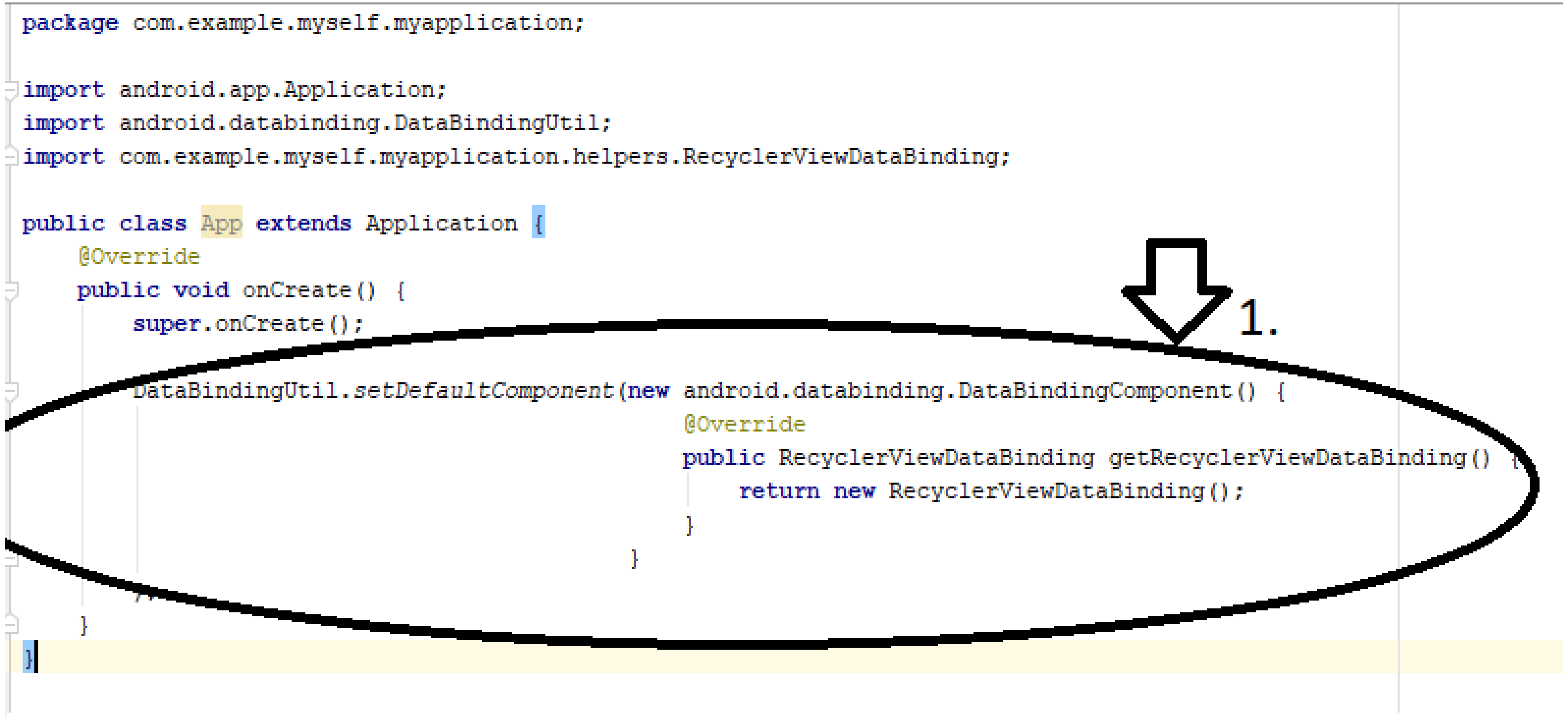


Android
activity_item_list.xml
GenericAdapter.java
App.java
RecyclerViewDataBinding.java
ItemListActivity.java
app
manifests
AndroidManifest.xml
java
com.example.myself.myapplication
helpers
GenericAdapter
RecyclerViewDataBinding
App
ItemListActivity
com.example.myself.myapplication (androidTes
com.example.myself.myapplication (test)
generatedJava
res
Gradle Scripts
build.gradle (Project: MyApplication)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for app)

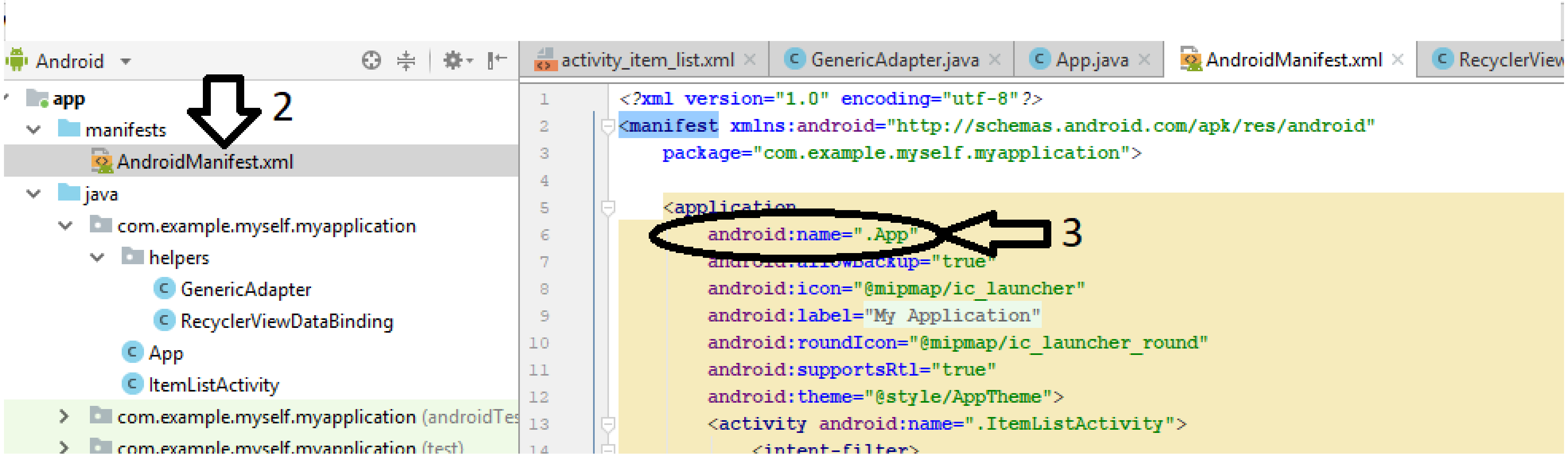
```java
package com.example.myself.myapplication;

import android.app.Application;
import android.databinding.DataBindingUtil;
import com.example.myself.myapplication.helpers.RecyclerViewDataBinding;

public class App extends Application {
    @Override
    public void onCreate() {
        super.onCreate();
    }
}
```

1. Create an App class at the root of your project

2. Add these two imports : DataBindingUtil & RecyclerViewDataBinding

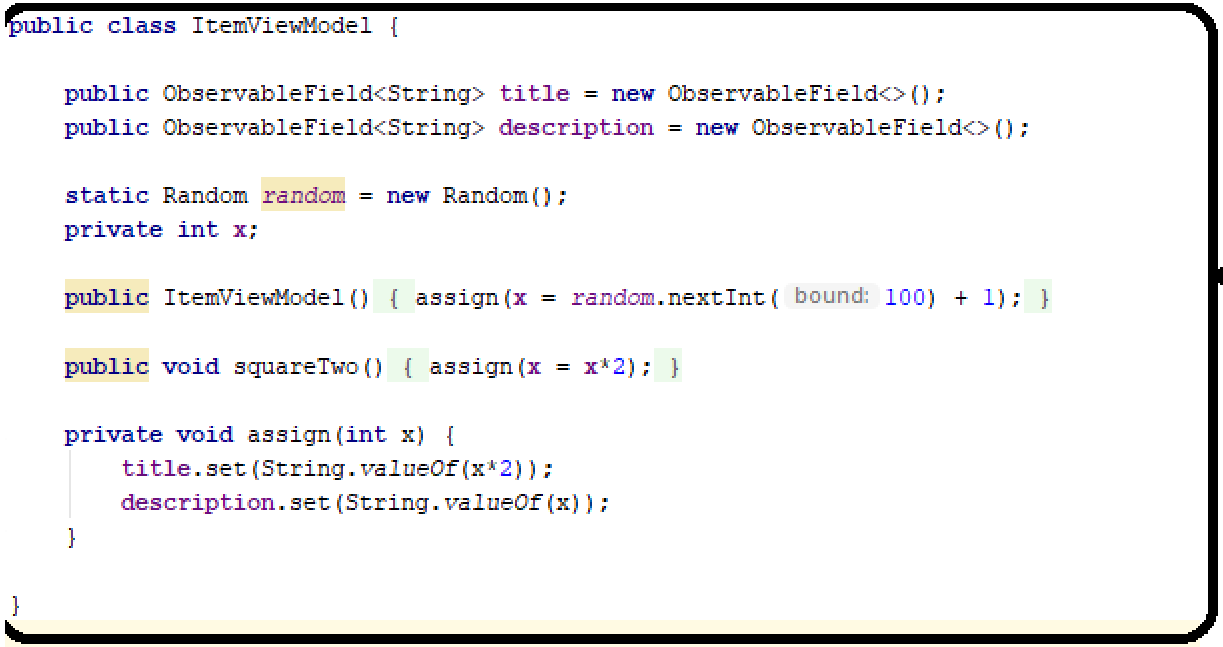


```
package com.example.myself.myapplication;

import android.app.Application;
import android.databinding.DataBindingUtil;
import com.example.myself.myapplication.helpers.RecyclerViewDataBinding;

public class App extends Application {
    @Override
    public void onCreate() {
        super.onCreate();

        DataBindingUtil.setDefaultComponent(new android.databinding.DataBindingComponent() {
                                                @Override
                                                public RecyclerViewDataBinding getRecyclerViewDataBinding() {
                                                    return new RecyclerViewDataBinding();
                                                }
                                            }
        );
    }
}
```



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

    <application
        android:name=".App"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".ItemListActivity">
            <intent-filter>
```

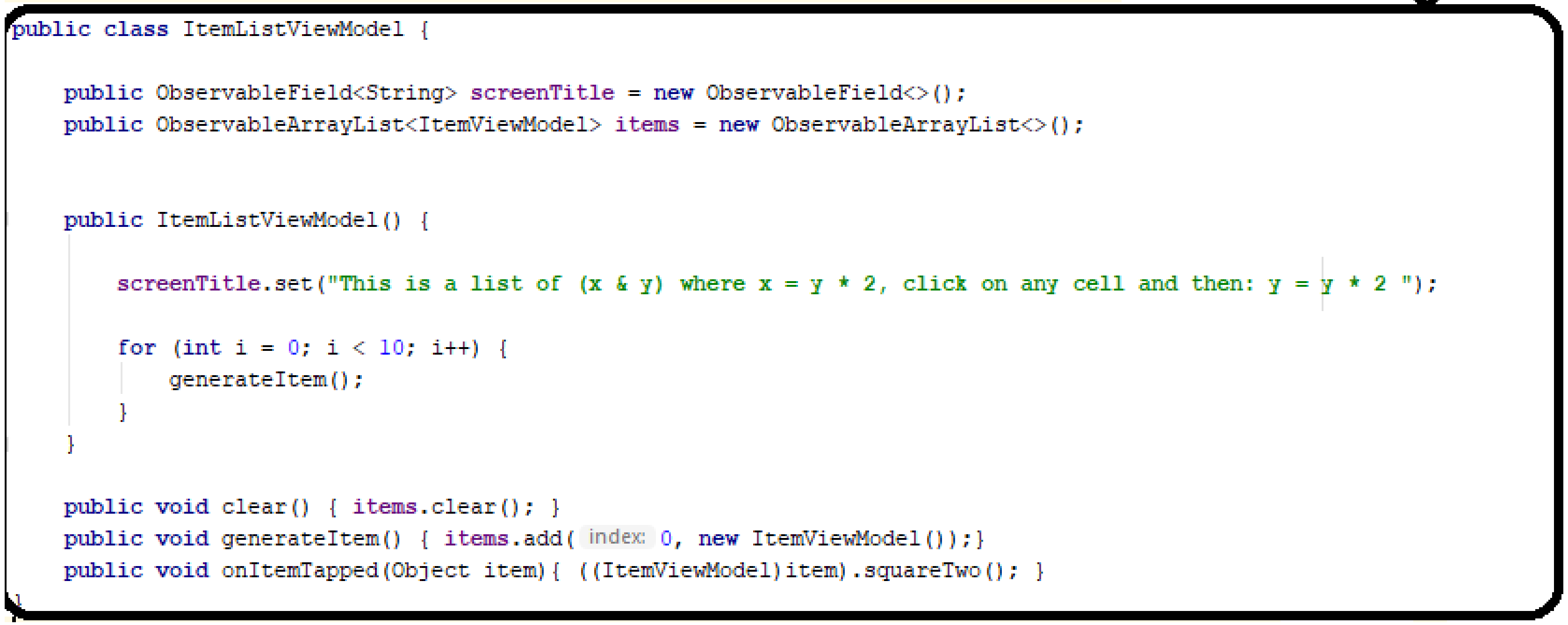
blog-android-clean-coding-approach-master 〉 app 〉 src 〉 main 〉



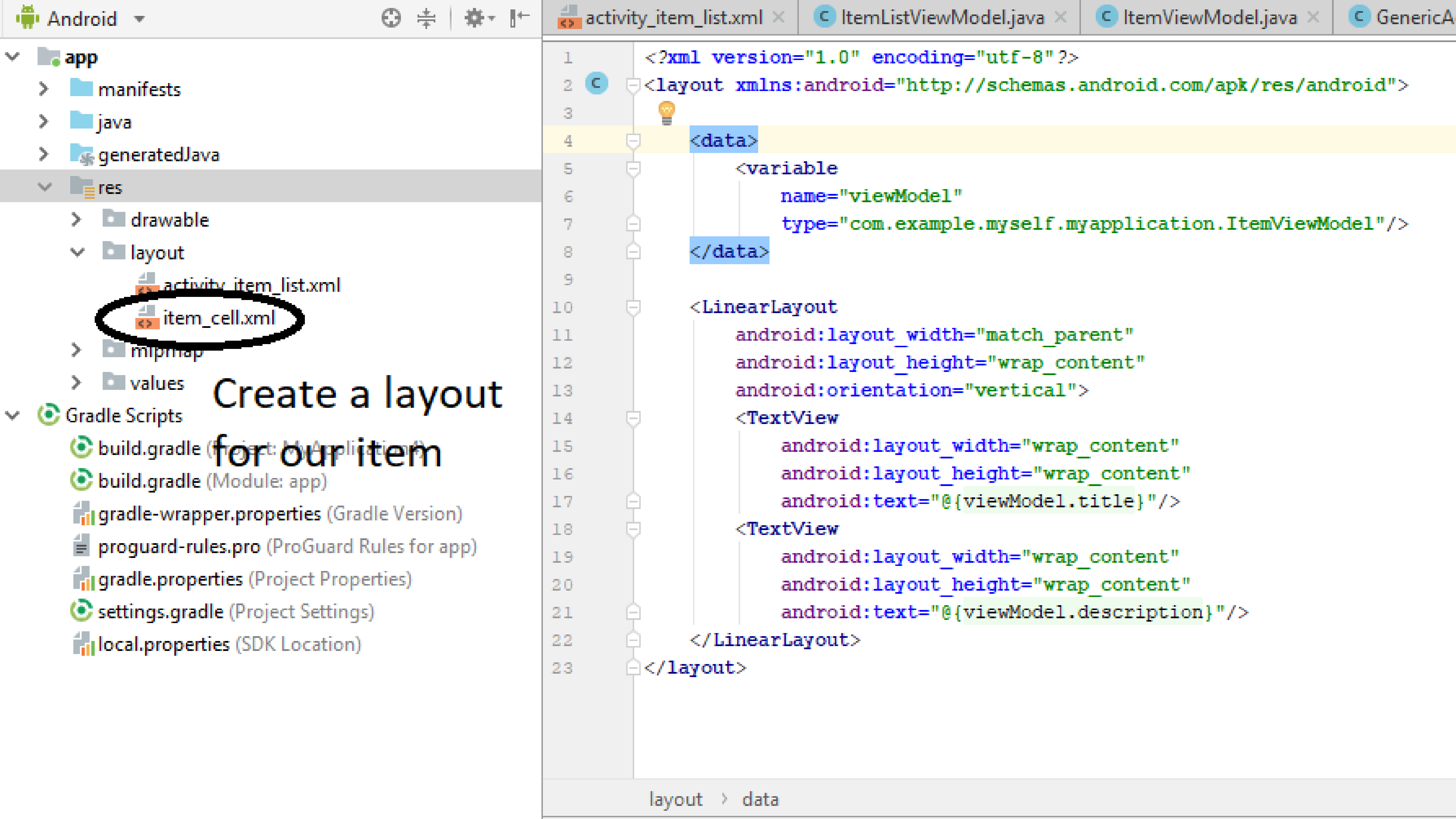
```
public class ItemViewModel {

    public ObservableField<String> title = new ObservableField<>();
    public ObservableField<String> description = new ObservableField<>();

    static Random random = new Random();
    private int x;

    public ItemViewModel() { assign(x = random.nextInt( bound: 100) + 1); }

    public void squareTwo() { assign(x = x*2); }

    private void assign(int x) {
        title.set(String.valueOf(x*2));
        description.set(String.valueOf(x));
    }

}
```

Create our ViewModels

ItemViewModel
& ItemListViewModel



```
public class ItemListViewModel {

    public ObservableField<String> screenTitle = new ObservableField<>();
    public ObservableArrayList<ItemViewModel> items = new ObservableArrayList<>();


    public ItemListViewModel() {

        screenTitle.set("This is a list of (x & y) where x = y * 2, click on any cell and then: y = y * 2 ");

        for (int i = 0; i < 10; i++) {
            generateItem();
        }
    }

    public void clear() { items.clear(); }
    public void generateItem() { items.add( index: 0, new ItemViewModel());}
    public void onItemTapped(Object item){ ((ItemViewModel)item).squareTwo(); }
}
```

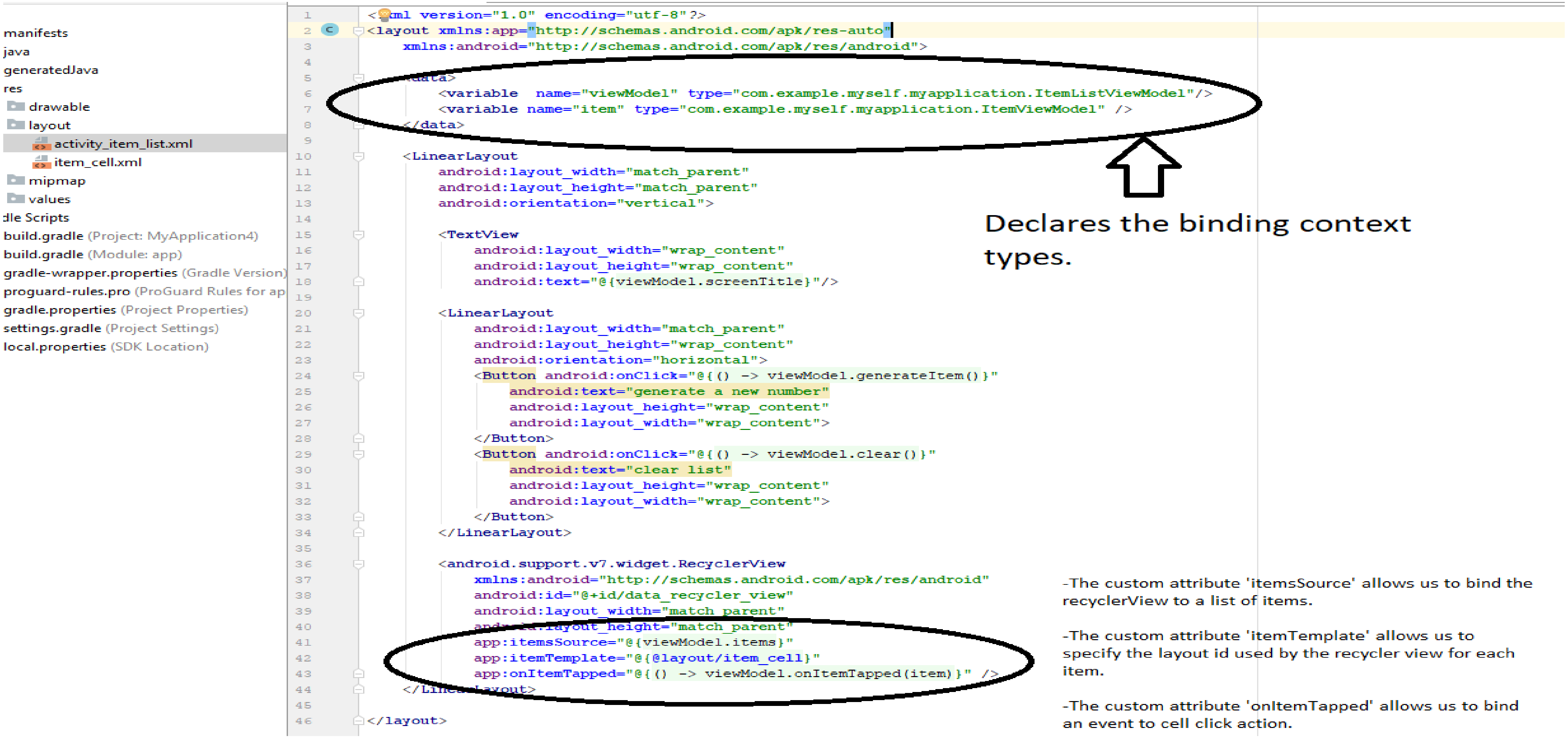


Android
app
manifests
java
generatedJava
res
drawable
layout
activity_item_list.xml
item_cell.xml
mipmap
values
Gradle Scripts
build.gradle (Project: MyApplication4)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for app)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
activity_item_list.xml
ItemListViewModel.java
ItemViewModel.java
GenericA
layout › data

Create a layout for our item

```xml
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">

    <data>
        <variable
            name="viewModel"
            type="com.example.myself.myapplication.ItemViewModel"/>
    </data>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@{viewModel.title}"/>
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@{viewModel.description}"/>
    </LinearLayout>
</layout>
```

Thanks to the helper classes, we're now able to have a bindable RecyclerView



```
manifests
java
generatedJava
res
  drawable
  layout
    activity_item_list.xml
    item_cell.xml
  mipmap
  values
dle Scripts
build.gradle (Project: MyApplication4)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for ap
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
```

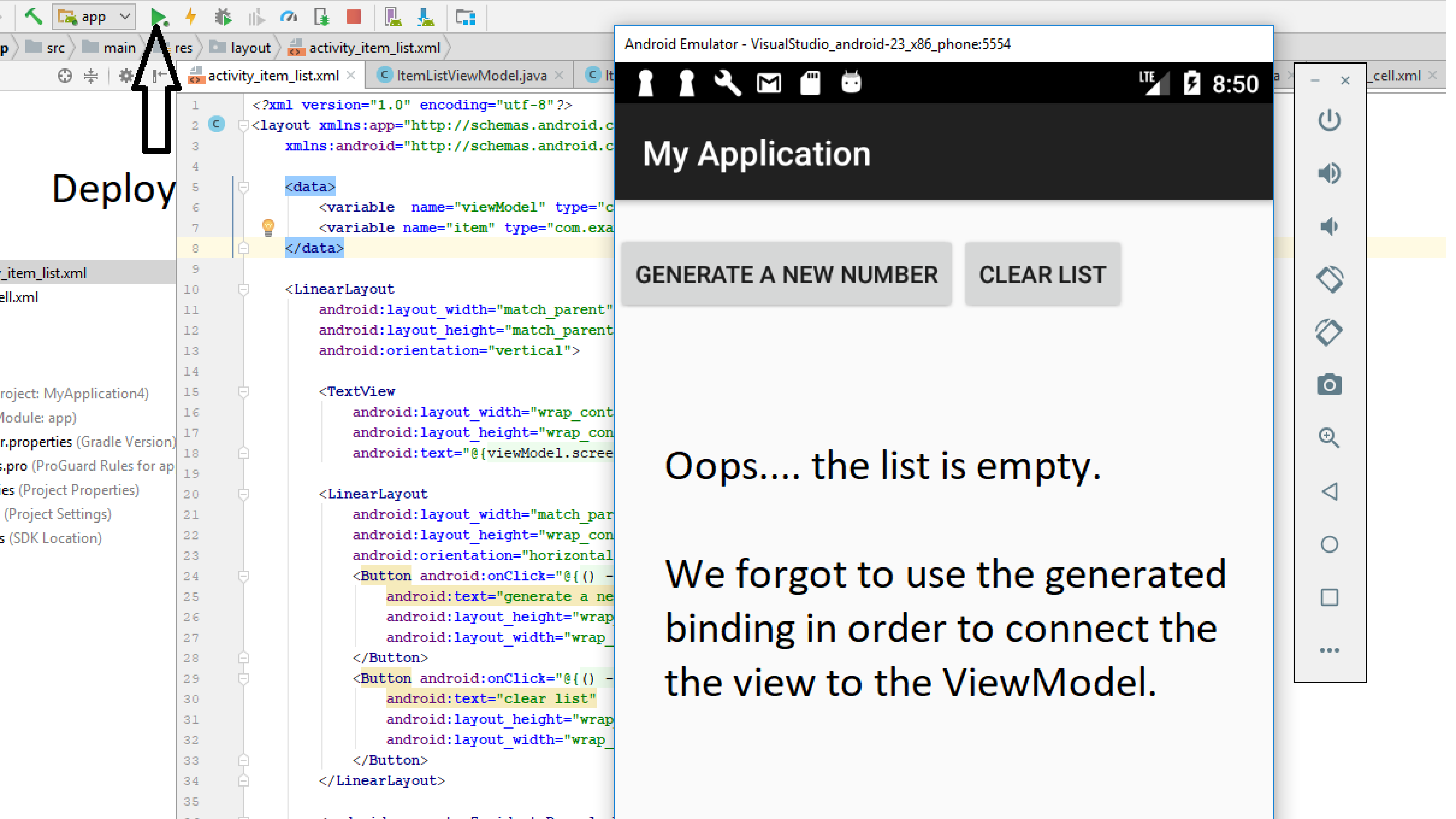
```xml
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <data>
        <variable  name="viewModel" type="com.example.myself.myapplication.ItemListViewModel"/>
        <variable name="item" type="com.example.myself.myapplication.ItemViewModel" />
    </data>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@{viewModel.screenTitle}"/>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <Button android:onClick="@{() -> viewModel.generateItem()}"
                android:text="generate a new number"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content">
            </Button>
            <Button android:onClick="@{() -> viewModel.clear()}"
                android:text="clear list"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content">
            </Button>
        </LinearLayout>

        <android.support.v7.widget.RecyclerView
            xmlns:android="http://schemas.android.com/apk/res/android"
            android:id="@+id/data_recycler_view"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:itemsSource="@{viewModel.items}"
            app:itemTemplate="@{@layout/item_cell}"
            app:onItemTapped="@{() -> viewModel.onItemTapped(item)}" />
    </LinearLayout>

</layout>
```

Declares the binding context types.

-The custom attribute 'itemsSource' allows us to bind the recyclerView to a list of items.

-The custom attribute 'itemTemplate' allows us to specify the layout id used by the recycler view for each item.

-The custom attribute 'onItemTapped' allows us to bind an event to cell click action.

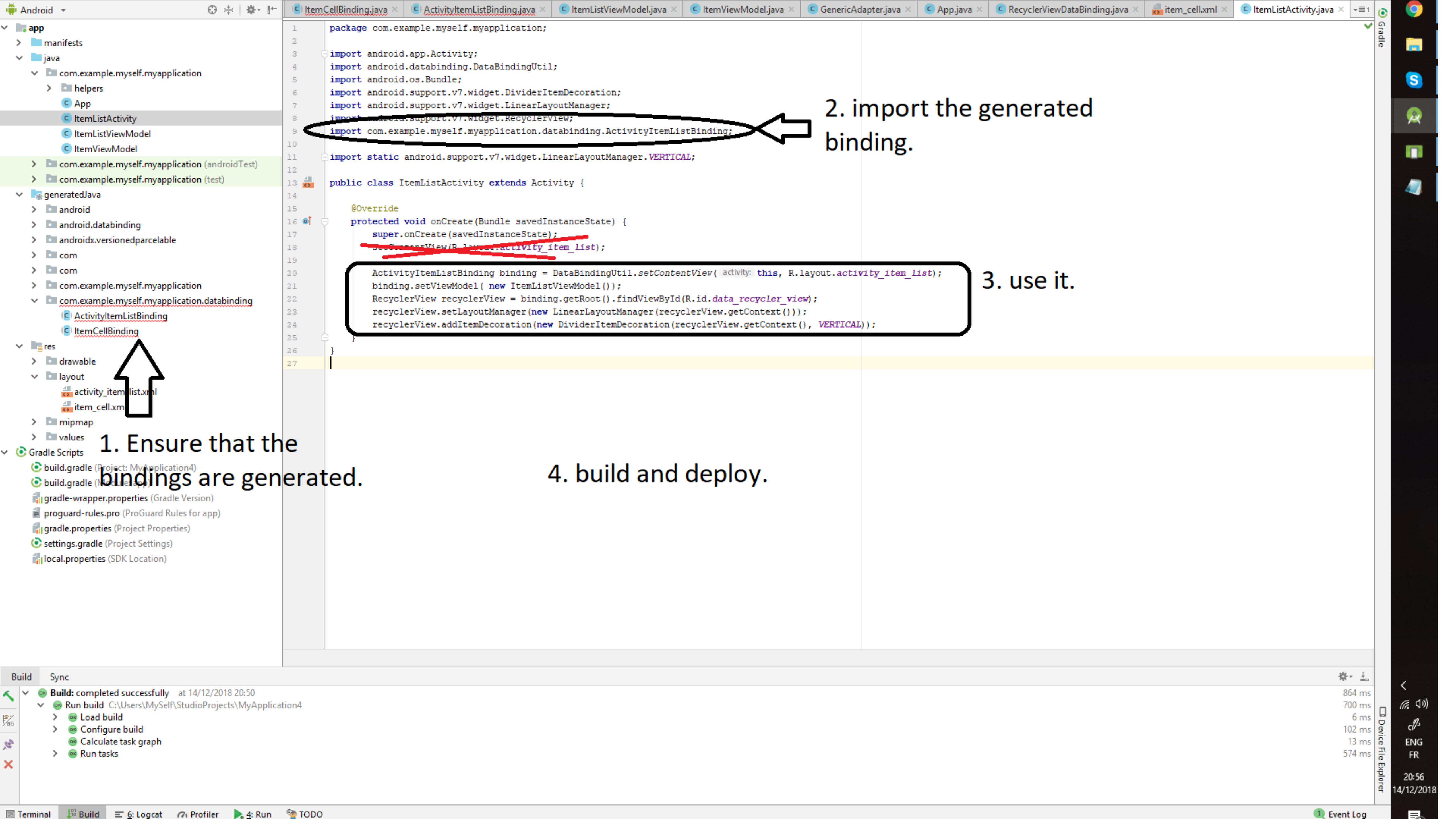


Deploy

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:app="http://schemas.android.c
    xmlns:android="http://schemas.android.c

    <data>
        <variable  name="viewModel" type="c
        <variable name="item" type="com.exa
    </data>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent
        android:orientation="vertical">

        <TextView
            android:layout_width="wrap_cont
            android:layout_height="wrap_con
            android:text="@{viewModel.scree

        <LinearLayout
            android:layout_width="match_par
            android:layout_height="wrap_con
            android:orientation="horizontal
            <Button android:onClick="@{() -
                android:text="generate a ne
                android:layout_height="wrap
                android:layout_width="wrap_
            </Button>
            <Button android:onClick="@{() -
                android:text="clear list"
                android:layout_height="wrap
                android:layout_width="wrap_
            </Button>
        </LinearLayout>
```

Oops.... the list is empty.

We forgot to use the generated binding in order to connect the the view to the ViewModel.


ItemCellBinding.java
ActivityItemListBinding.java
ItemListViewModel.java
ItemViewModel.java
GenericAdapter.java
App.java
RecyclerViewDataBinding.java
item_cell.xml
ItemListActivity.java
Android
app
manifests
java
com.example.myself.myapplication
helpers
App
ItemListActivity
ItemListViewModel
ItemViewModel
com.example.myself.myapplication (androidTest)
com.example.myself.myapplication (test)
generatedJava
android
android.databinding
androidx.versionedparcelable
com
com
com.example.myself.myapplication
com.example.myself.myapplication.databinding
ActivityItemListBinding
ItemCellBinding
res
drawable
layout
activity_item_list.xml
item_cell.xml
mipmap
values
Gradle Scripts
build.gradle (Project: MyApplication4)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for app)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
Build
Sync
Terminal
Build
6: Logcat
Profiler
4: Run
TODO
Event Log
Gradle
Device File Explorer
ENG
FR
20:56
14/12/2018

```java
package com.example.myself.myapplication;

import android.app.Activity;
import android.databinding.DataBindingUtil;
import android.os.Bundle;
import android.support.v7.widget.DividerItemDecoration;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import com.example.myself.myapplication.databinding.ActivityItemListBinding;

import static android.support.v7.widget.LinearLayoutManager.VERTICAL;

public class ItemListActivity extends Activity {

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

        ActivityItemListBinding binding = DataBindingUtil.setContentView( activity: this, R.layout.activity_item_list);
        binding.setViewModel( new ItemListViewModel());
        RecyclerView recyclerView = binding.getRoot().findViewById(R.id.data_recycler_view);
        recyclerView.setLayoutManager(new LinearLayoutManager(recyclerView.getContext()));
        recyclerView.addItemDecoration(new DividerItemDecoration(recyclerView.getContext(), VERTICAL));
    }
}
```

1. Ensure that the bindings are generated.

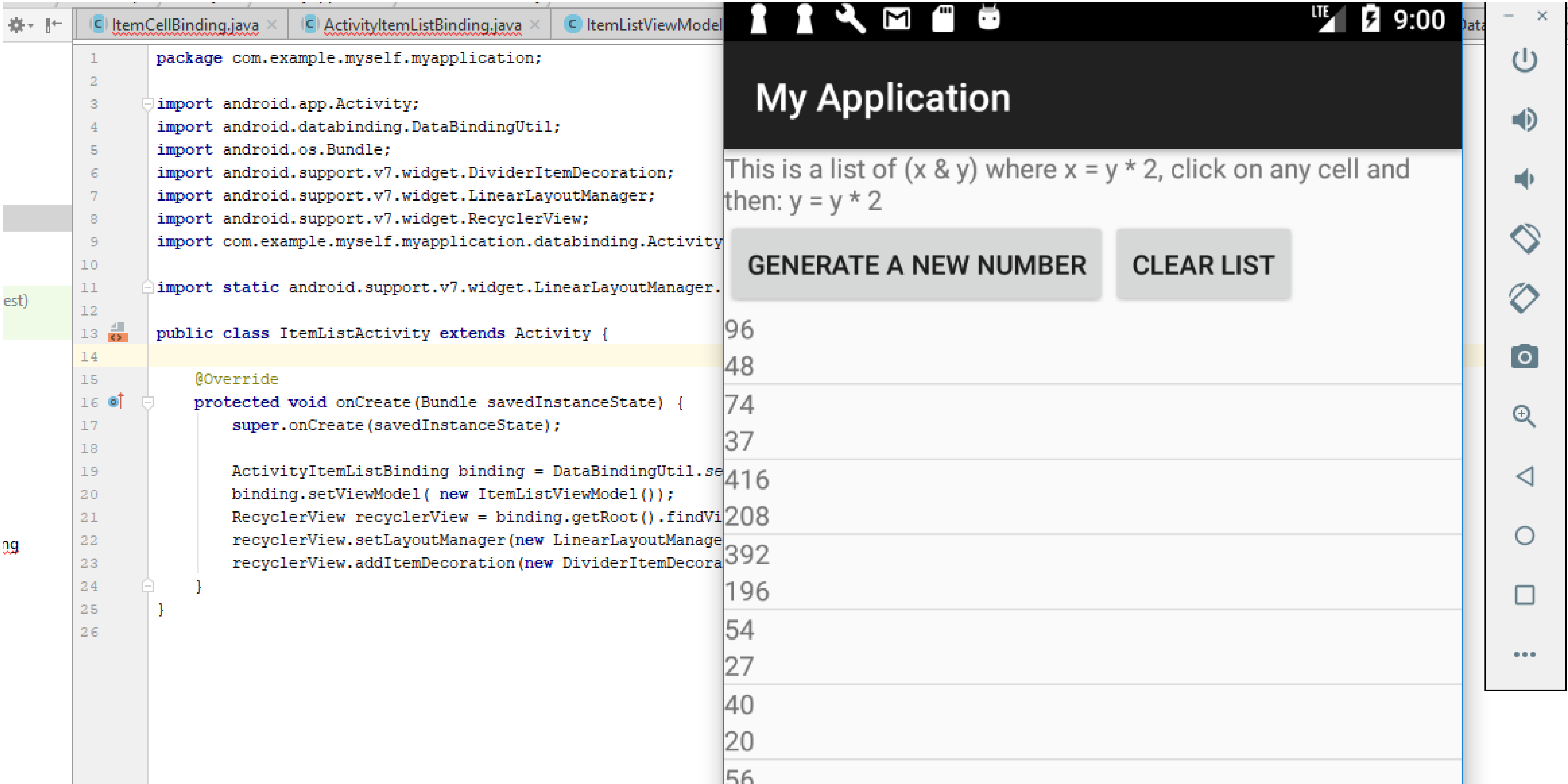
2. import the generated binding.

3. use it.

4. build and deploy.

```
Build: completed successfully   at 14/12/2018 20:50      864 ms
  Run build C:\Users\MySelf\StudioProjects\MyApplication4  700 ms
    Load build                                               6 ms
    Configure build                                        102 ms
    Calculate task graph                                    13 ms
    Run tasks                                              574 ms
```

The sexyiest app ever made by a homo-sapiens.



```
package com.example.myself.myapplication;

import android.app.Activity;
import android.databinding.DataBindingUtil;
import android.os.Bundle;
import android.support.v7.widget.DividerItemDecoration;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import com.example.myself.myapplication.databinding.Activity

import static android.support.v7.widget.LinearLayoutManager.

public class ItemListActivity extends Activity {

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

        ActivityItemListBinding binding = DataBindingUtil.se
        binding.setViewModel( new ItemListViewModel());
        RecyclerView recyclerView = binding.getRoot().findVi
        recyclerView.setLayoutManager(new LinearLayoutManage
        recyclerView.addItemDecoration(new DividerItemDecora
    }
}
```

This is a list of (x & y) where x = y * 2, click on any cell and then: y = y * 2

```
96
48
74
37
416
208
392
196
54
27
40
20
56
```

End of the tutorial

The final result of this tutorial is on github:

https://github.com/sferhah/Android-MVVM-Tutorial