# Lab 2 part 3

In this part, students will develop a mobile application to display data as a list and using menu for further setting. The layout included will be shown below:

* LinearLayout
* TextView
* RecyclerView
* Option menu
* Context menu

Here are the screen captures of the application:

Graphical user interface, text

Description automatically generated with medium confidenceGraphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, text, application, chat or text message

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Students will also apply some object oriented programming concept in this lab such as interface.

Step 1:

In Android Studio, create a new project named Lab 2 Part 3 with following project setting:

* Choose your project : **Empty Activity**
* Application Name : **Lab 2 Part 3**
* Project location : use the default setting
* Language : **Java**
* Minimum API level: **API 21: Android 5.0 (Lollipop)**
* Click **Finish**

Step 2:

Define string values in **res/values/strings.xml**

<resources>  
 <string name="app\_name">Lab 2 Part 3</string>  
 <string name="planet">Planet</string>  
 <string name="planets">Planet - </string>  
 <string-array name="planets\_array">  
 <item>Mercury</item>  
 <item>Venus</item>  
 <item>Earth</item>  
 <item>Mars</item>  
 <item>Jupiter</item>  
 <item>Saturn</item>  
 <item>Uranus</item>  
 <item>Neptune</item>  
 </string-array>  
</resources>

Step 3:

Open the **res/layout/activity\_main.xml** file and insert the following:

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/selection"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/planet" />  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/rV"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
  
</LinearLayout>

Step 4:

Create list\_items.xml to define the element of recycler view. Right click layout-> New -> Layout Resource File, type “list\_items” and click OK.

Background pattern

Description automatically generated

Step 5:

Open the **res/layout/list\_items.xml** file and insert the following:

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="85dp"  
 android:gravity="center\_vertical"  
 android:orientation="horizontal">  
  
 <TextView  
 android:id="@+id/txtItem"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="20dp"  
 android:layout\_weight="1" />  
  
</LinearLayout>

Step 6:

Right click com.example.lab2part3 -> New -> Java class, choose **Interface** and type “OnRecyclerViewClickListener”, then press Enter.

OnRecyclerViewClickListener.java is used to provide a framework to handle the event of recycler view clicking, it is no necessary to define the detail in the interface.

Step 7:

Open OnRecyclerViewClickListener**.java** and insert the following code for the class:

public interface OnRecyclerViewClickListener {  
 void onItemClickListener(View view);  
}

Step 8:

Right click com.example.lab2part3 -> New -> Java class, choose **Java** and type “MyViewHolder”, then press Enter.

Step 9:

Open MyViewHolder.java and insert the following code:

public class MyViewHolder extends RecyclerView.ViewHolder{

*//Item inside the RecyclerView* protected TextView txtItem;  
 public MyViewHolder(@NonNull View itemView) {  
 super(itemView);  
 txtItem=itemView.findViewById(R.id.*txtItem*);  
  
  
 }  
}

Why we need to use itemView.findViewById instead of findViewById? Since MyViewHolder is under itemView, it cannot be initialized in Activity layout directly.

Step 10:

Right click com.example.lab2part3 -> New -> Java class, choose **Java** and type “MyAdapter”, then press Enter.

Step 11 (you can copy the whole code at last page):

Open MyAdapter.java, add extends RecyclerView.Adapter<MyViewHolder>, then you will see the following code:

public class MyAdapter extends RecyclerView.Adapter<MyViewHolder> {  
 @NonNull  
 @Override  
 public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
  
 return null;  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {

}  
  
 @Override  
 public int getItemCount() {  
 return 0;  
 }  
  
}

Step 12:

Add Context object and String array for the information return from the interface. It also used to set a constructor.

Context context;  
String[] mData;

Step 13:

Create a constructor

public MyAdapter(Context context, String[] mData){ *//constructor* this.context=context;  
 this.mData=mData;  
}

Step 14:

Create an attribute and method of onRecyclerViewClickListener.

Context context;  
String[] mData;  
private OnRecyclerViewClickListener listener; *//Object of OnRecyclerViewClickListener*

public void setItemClickListener(OnRecyclerViewClickListener itemClickListener) { *//build setItemClickListener（）inside the Adapter* listener = itemClickListener;  
}

Step 15:  
  
Replace the following codes of onCreateViewHolder() and onBindViewHolder()

@Override  
 public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 *// link list\_item* View view = LayoutInflater.*from*(parent.getContext())  
 .inflate(R.layout.*list\_items*, parent, false);  
  
 MyViewHolder myViewHolder=new MyViewHolder(view);  
  
 *//interface e feedback* if(listener != null){  
 view.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 listener.onItemClickListener(v);  
 }  
 });}  
  
 return myViewHolder;  
}  
  
 @Override  
 public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {  
 holder.txtItem.setText(mData[position]); *// set the content that need to display in txtItem* }

Step 16:

Open MainActivity.java and insert the following code:

public class MainActivity extends AppCompatActivity {  
  
  
 private TextView selection;  
 private RecyclerView recycler\_view;  
 private MyAdapter adapter;  
 String[] dataArray;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 dataArray = getResources().getStringArray(R.array.*planets\_array*);  
  
 recycler\_view = findViewById(R.id.*rV*);  
 selection = findViewById(R.id.*selection*);  
  
 *// set RecyclerView as a list* recycler\_view.setLayoutManager(new LinearLayoutManager(this));  
 *// pass data to adapter* adapter = new MyAdapter(this, dataArray);  
 *// set adapter to recycler\_view* recycler\_view.setAdapter(adapter);  
  
  
 adapter.setItemClickListener(new OnRecyclerViewClickListener() { *//need to implement method* @Override  
 public void onItemClickListener(View view) {  
  
 int position = recycler\_view.getChildAdapterPosition(view); *//get the position* selection.setText("Planet - " + dataArray[position]);  
  
 }  
});  
  
 }  
  
 }

Step 17:

Create menu directory and menu.xml

Step 18:

Insert the following code to menu.xml

*<?*xml version="1.0" encoding="utf-8"*?>*<menu xmlns:android="http://schemas.android.com/apk/res/android">  
 <item  
 android:id="@+id/action\_wiki"  
 android:title="Wiki">  
 </item>  
 <item  
 android:id="@+id/action\_exit"  
 android:title="Exit">  
 </item>  
</menu>

Step 19:

Add the following code outside the onCreate() method to create option menu.

@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*, menu);  
 return true;  
}  
  
@Override  
public boolean onOptionsItemSelected(@NonNull MenuItem item) {  
 int id = item.getItemId();  
 switch (id) {  
 case R.id.*action\_wiki*:  
 Intent intent = new Intent(Intent.*ACTION\_VIEW*,  
 Uri.*parse*("https://en.wikipedia.org/wiki/Planet"));  
 startActivity(intent);  
 return true;  
  
 case R.id.*action\_exit*:  
 finish();  
 return true;  
 default: return super.onOptionsItemSelected(item);  
 }  
  
}

Step 20:

Add the following code outside the onCreate() method to create context menu.

@Override  
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {  
 super.onCreateContextMenu(menu, v, menuInfo);  
 MenuInflater inflater = getMenuInflater();  
 inflater.inflate(R.menu.*menu*, menu);  
}  
  
@Override  
public boolean onContextItemSelected(@NonNull MenuItem item) {  
 switch (item.getItemId()) {  
 case R.id.*action\_wiki*:  
 Intent intent = new Intent(Intent.*ACTION\_VIEW*,  
 Uri.*parse*("https://en.wikipedia.org/wiki/Planet"));  
 startActivity(intent);  
 return true;  
  
 case R.id.*action\_exit*:  
 finish();  
 return true;  
 default:  
 return super.onContextItemSelected(item);  
 }  
}

Step 21:

You will see context menu cannot appear at the moment. Add the following code inside the onCreate() method.

registerForContextMenu(selection);

Step 22:

Open AndroidManifest.xml and add the following code:

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

MyAdapter.java

public class MyAdapter extends RecyclerView.Adapter<MyViewHolder> {  
 @NonNull  
 Context context;  
 String[] mData;  
 private OnRecyclerViewClickListener listener; public void setItemClickListener(OnRecyclerViewClickListener itemClickListener) { listener = itemClickListener;  
 }  
  
 public MyAdapter(Context context, String[] mData){ this.context=context;  
 this.mData=mData;  
 }  
 @Override  
 public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext())  
 .inflate(R.layout.*list\_items*, parent, false);  
  
 MyViewHolder myViewHolder=new MyViewHolder(view);  
  
 if(listener != null){  
 view.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 listener.onItemClickListener(v);  
 }  
 });  
}  
  
 return myViewHolder;  
}  
  
 @Override  
 public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {  
 holder.txtItem.setText(mData[position]);

}  
  
 @Override  
 public int getItemCount() {  
 return mData.length;  
 }  
  
  
}