**Android Sensor Programing**

**By: Savan Yeshwanth Rao**

**CSUid: 2784780**

Develop a contact management app version 1:

* The main activity would start with no contact, but it has a button where the user could use to add a new contact
* On clicking” Add Contact” button in the main activity, it goes to the second activity, where the user could add the contact information such as name, phone number, email, etc.
* After entering a new contact, the user could click the submit button in the second activity, which the second activity will be closed, and the app goes back to the main activity
* The main activity now should display the new contact information.

In the activity\_main.xml we create a button to which is used to add contact using the button code and also to display the contact information which is provided in the SecondActivity.java we need to pass the data from second activity to the main activity. For creating the button, we use the below code.

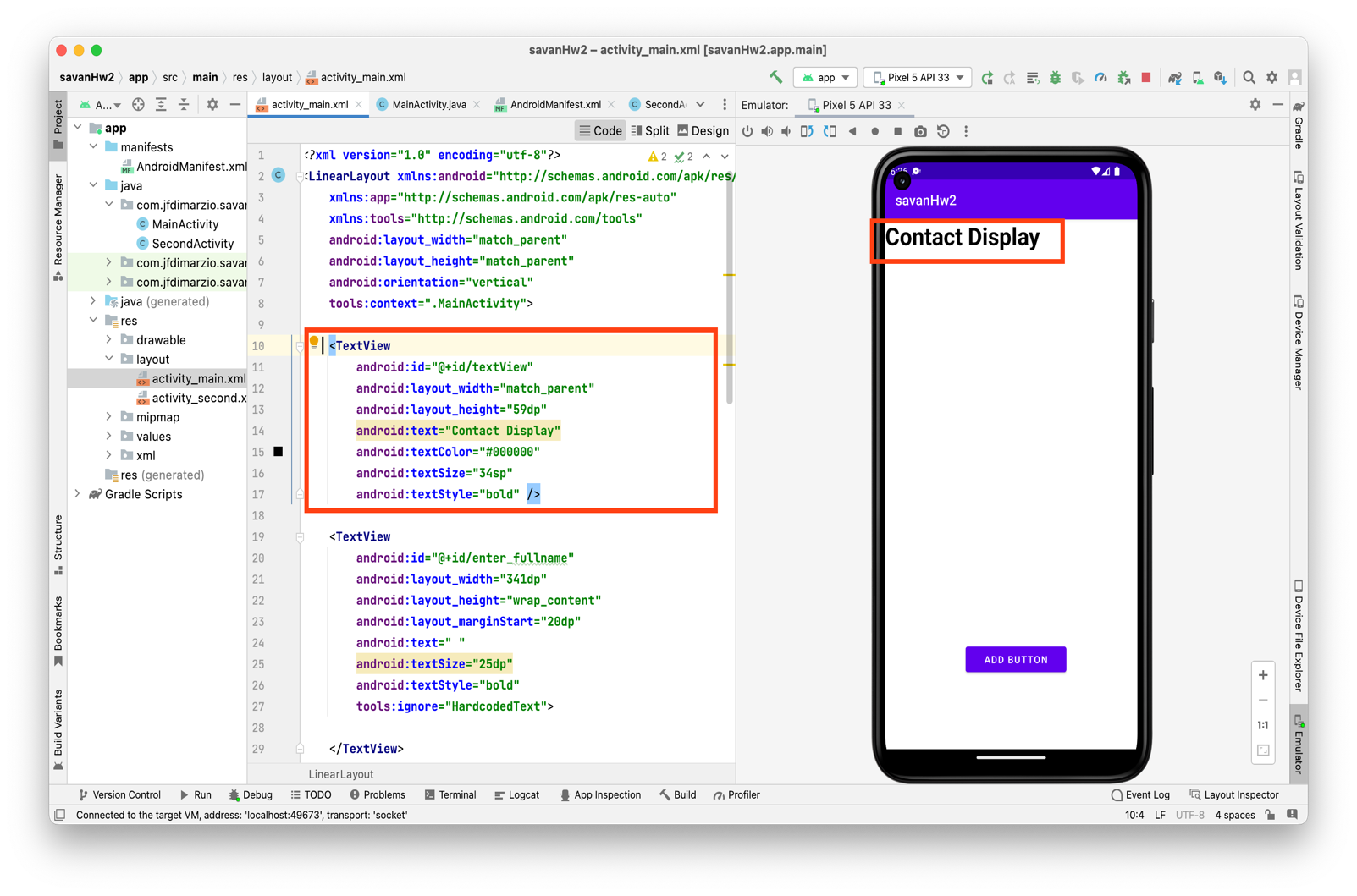
<**Button  
 android:id="@+id/buttonAdd"  
 android:layout\_width="157dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="125dp"  
 android:layout\_marginTop="450sp"  
 android:layout\_marginEnd="150dp"  
 android:layout\_marginBottom="15dp"  
 android:text="Add Button"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:ignore="HardcodedText"** />

Graphical user interface, application

Description automatically generated

I have also created a TextView “Contact Display” so that whatever is added in the secondactivity.java is displayed below it. To do so we will use the TextView code as shown below.

<**TextView  
 android:id="@+id/textView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="59dp"  
 android:text="Contact Display"  
 android:textColor="#000000"  
 android:textSize="34sp"  
 android:textStyle="bold"** />



Now to create an empty space so that the SecondActivity.java data is passed to the MainActivity.java we create an empty space TextView to view the text such as full name, phone number, email id entered by the user. Which would be displayed bellow the Contact display. Below Screenshots shows us both the emulator and the design of the XML file.

<**TextView  
 android:id="@+id/enter\_fullname"  
 android:layout\_width="341dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="20dp"  
 android:text=" "  
 android:textSize="25dp"  
 android:textStyle="bold"  
 tools:ignore="HardcodedText"**>  
  
</**TextView**>  
  
 **android:id="@+id/enter\_phonenumber"  
 android:layout\_width="338dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="20dp"  
 android:text=" "  
 android:textSize="20sp"  
 android:textStyle="normal"  
 tools:ignore="HardcodedText"**>  
  
</**TextView**>  
  
<**TextView  
 android:id="@+id/enter\_email"  
 android:layout\_width="337dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="20dp"  
 android:text=" "  
 android:textSize="20sp"  
 android:textStyle="normal"  
 tools:ignore="HardcodedText"**>  
</**TextView**>

<-To display full name

<- To display email

<- To display phone number

<- To display email

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Coming to activity\_second.xml we need to create a Save button where once we click that the data is passed back to the main activity and displayed in the mainactivity.java, to create that again we use a button command and add the required dimensions and after the design the code and the screenshot is shown below. I have used linear layouts for all the xml in activity\_second.xml individually.

<**LinearLayout  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"**>  
  
 <**Button  
 android:id="@+id/saveButton"  
 android:layout\_width="175dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="125dp"  
 android:layout\_marginTop="450sp"  
 android:layout\_marginEnd="150dp"  
 android:layout\_marginBottom="15dp"  
 android:text="Save"  
 android:textStyle="normal"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"** />

Graphical user interface, application

Description automatically generated

Next to display the name we have used a TextView, and EditText, in order to show the display of what to be inserted to in the blank space, also provided the hit what to be entered in the activity\_second.xml, so that once the user enter to the second activity he/she would be clearly guided where to enter the full name, phone number and the email that needs to be displayed in the first activity. The below code shows us the following lines to create them, and the screenshots shows the outcome of the code.

<**LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/firstLine"**>  
  
 <**TextView  
 android:layout\_width="101dp"  
 android:layout\_height="wrap\_content"  
 android:text="Full name"  
 android:textSize="20dp"  
 android:textStyle="bold"** />  
  
 <**EditText  
 android:id="@+id/enter\_text\_Fullname"  
 android:layout\_width="271dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="40dp"  
 android:layout\_marginTop="20dp"  
 android:hint="enter name"  
 android:textSize="25dp"  
 android:textStyle="normal"** />  
  
</**LinearLayout**>  
  
<**LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/secondline"  
 android:layout\_below="@id/firstLine"**>  
  
 <**TextView  
 android:layout\_width="131dp"  
 android:layout\_height="wrap\_content"  
 android:text="Cell Number"  
 android:textSize="20dp"  
 android:textStyle="bold"** />  
  
 <**EditText  
 android:id="@+id/enter\_text\_Phonenumber"  
 android:layout\_width="263dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginTop="20dp"  
 android:hint="enter phone number"  
 android:textSize="25dp"  
 android:textStyle="normal"** />  
</**LinearLayout**>  
  
<**LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/thirdline"  
 android:layout\_below="@id/secondline"**>  
  
 <**TextView  
 android:layout\_width="90dp"  
 android:layout\_height="wrap\_content"  
 android:text="Email"  
 android:textSize="20dp"  
 android:textStyle="bold"** />  
  
 <**EditText  
 android:id="@+id/enter\_text\_Email"  
 android:layout\_width="276dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="40dp"  
 android:layout\_marginTop="20dp"  
 android:hint="enter email-id"  
 android:textSize="25dp"  
 android:textStyle="normal"** />  
</**LinearLayout**>

<- To enter full name text

<- To enter phone number

<- To enter email id

Graphical user interface, application

Description automatically generated

As the SecondActivity.java plays the main role in this assignment, as the data to be displayed is entered in the 2nd activity we do it using the Intent option available, as though in the class. The below code shows us how each line that would be entered in the SecondActivity.java. Each of the id we have used in the activity\_second.xml is called in SecondActivity.java using “findViewById(R.id.xxxxxx)”, later onClick(View view) we create the string we needed and is pushed to the temp value, which act as temp storage to store all full name, phone number and the email. The later we push the values called back in the MainActivity.java. Below we can see how the Intent option is used and the outcome of the SecondActivity.java after debug.

**public class** SecondActivity **extends** AppCompatActivity {  
 EditText **enter\_Fullname**, **enter\_Phonenumber**, **enter\_Email**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_second***);  
 Button saveButton = findViewById(R.id.***saveButton***);  
  
 **enter\_Fullname** = (EditText) findViewById(R.id.***enter\_text\_Fullname***);  
 **enter\_Phonenumber** = (EditText) findViewById(R.id.***enter\_text\_Phonenumber***);  
 **enter\_Email** = (EditText) findViewById(R.id.***enter\_text\_Email***);  
  
saveButton.setOnClickListener(**new** View.OnClickListener() {

**public void** onClick(View view) {  
 String strFullname = **enter\_Fullname**.getText().toString();  
 String strPhonenumber = **enter\_Phonenumber**.getText().toString();  
 String strEmail = **enter\_Email**.getText().toString();  
 Intent activityIntent = **new** Intent(getApplicationContext(), MainActivity.**class**);  
 activityIntent.putExtra(**"temp\_fullname"**, strFullname);  
 activityIntent.putExtra(**"temp\_phonenumber"**, strPhonenumber);  
 activityIntent.putExtra(**"temp\_email"**, strEmail);  
 startActivity(activityIntent);

Graphical user interface, text, application

Description automatically generated

Now, in final step we setup and configure our MainActivity.java which is used to take the data that has been passed from SecondActivity.java and is to be displayed under the contact display. We first use TextView and declare global variables such as display\_Fullname, display\_Phonenumber, display\_Email, to display the entered full name, phone number, and the email in the SecondActivity.java by using the “Intent intent = getIntent();” to call the function from second activity. Then all the temp values entered on the second activity is displayed on the first activity. The button is used to add contact in the second activity. The code and the outcome is showed below.

**public class** MainActivity **extends** AppCompatActivity {  
 TextView **display\_Fullname**, **display\_Phonenumber**, **display\_Email**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 Button buttonAdd = findViewById(R.id.***buttonAdd***);  
  
 **display\_Fullname** = (TextView) findViewById(R.id.***enter\_fullname***);  
 **display\_Phonenumber** = (TextView) findViewById(R.id.***enter\_phonenumber***);  
 **display\_Email** = (TextView) findViewById(R.id.***enter\_email***);  
  
 Intent intent = getIntent();  
 String str\_fullname = intent.getStringExtra(**"temp\_fullname"**);  
 String str\_phonenumber = intent.getStringExtra(**"temp\_phonenumber"**);  
 String str\_email = intent.getStringExtra(**"temp\_email"**);  
 **display\_Fullname**.setText(str\_fullname);  
 **display\_Phonenumber**.setText(str\_phonenumber);  
 **display\_Email**.setText(str\_email);  
  
 buttonAdd.setOnClickListener(**new** View.OnClickListener() {  
 **public void** onClick(View view) {  
 Intent activity2Intent = **new** Intent(getApplicationContext(), SecondActivity.**class**);  
 startActivity(activity2Intent);  
 }  
 });  
  
 }

Graphical user interface, application

Description automatically generated

The outcome of the of the contact application is shown below when run on the emulator and all the function are being tested. The demo video has also been uploaded in the screenshot zip file.

1: Empty contact list. 2: To enter details in 2nd activity. 3: After entering the details.

|  |  |  |
| --- | --- | --- |
| Shape  Description automatically generated | Graphical user interface, application, Teams  Description automatically generated | Graphical user interface, text, application, chat or text message, email  Description automatically generated |

|  |
| --- |
| Text  Description automatically generated with low confidence |

4: After all the details are being entered in the 3 we can see that the entered details are been displayed back in the MainActivity below the contact display.