**Code Inspection document**

**for**

**‘*RING ME – A Mobile management application*’ Phase I requirements**

Prepared by: Team Mode Changer (Venkata Vikas Chirumamilla, Chenchu Sai Krishna Kolli, Siri Gogineni, Revanth Reddy Malreddy, Sai Teja Malle)

# Structure of Android code

To understand the code flow, one must understand how Android is built. Each android application is associated with screens which is built using XML code. This GUI is interlinked to work with server using java code at the backend. IDE builds/generates .apk file for the application using gradle build. This .apk file needs to install/deployed on the android phone and can be used.

# Execution flow:

1. All resource files are combined together by AAP[Android Asset Packing Tool]. Resource files are like audio video images other asset related files. 2.Java files converted into .class files by JVM.So, the out of the jvm will be .class files, that are heavy weight to put into android. So, that one more level of process will be taken place.
2. So, the .Class files are entered as input to DX tool. Basically, this is a tool which will convert .class files to .dex files. That mean Dalvik executable file. Those files are eligible to execute on DVM (Dalvik Virtual Machine)
3. After getting .dex files, packed them APK builder. Which is basically, Application Packaging. So, this packed files kept into devices and that will be executed by DVM.

The below figure 1.1. explains about the android code execution flow.

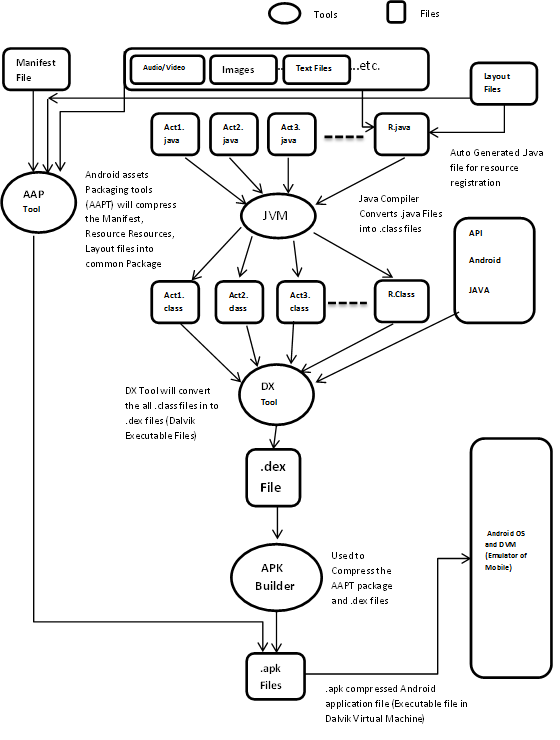


Figure 1.1. Android code execution flow

[Figure 1.1. Reference: https://stackoverflow.com/questions/5749436/android-application-control-flow]

# Source code

# AndroidManifest.xml

The **manifest** presents essential information about the application to the **Android** system, information the system must have before it can run any of the application's code. It describes the components of the application — the activities, services, broadcast receivers, and content providers that the application is composed of. It names the classes that implement each of the components and publishes their capabilities. These declarations let the Android system know what the components are and under what conditions they can be launched.

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.vivartha.modechanger"**>  
  
 *<!-- To auto-complete the email text field in the login form with the user's emails -->* <**uses-permission android:name="android.permission.GET\_ACCOUNTS"** />  
 <**uses-permission android:name="android.permission.READ\_PROFILE"** />  
 <**uses-permission android:name="android.permission.READ\_CONTACTS"** />  
 <**uses-permission android:name="android.permission.RECEIVE\_SMS"** />  
 <**uses-permission android:name="android.permission.READ\_SMS"** />  
 <**uses-permission android:name="android.permission.SEND\_SMS"** />  
  
 <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
 <**activity  
 android:name=".LoginActivity"  
 android:label="@string/app\_name"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 </**intent-filter**>  
 </**activity**>  
 <**activity android:name=".MainActivity"** />  
  
 <**receiver  
 android:name=".MyReceiver"  
 android:enabled="true"  
 android:process=":remote"**>  
 <**intent-filter android:label="MODE CHANGER"**>  
 <**action android:name="android.provider.Telephony.SMS\_RECEIVED"** />  
 </**intent-filter**>  
 </**receiver**>  
  
 <**activity  
 android:name=".about\_us"  
 android:label="@string/title\_activity\_about\_us"** />  
 <**activity  
 android:name=".Home\_Activity"  
 android:label="@string/title\_activity\_home\_"** />  
 <**activity  
 android:name=".Splash\_Screen"  
 android:theme="@style/SplashScreenTheme"  
 android:label="@string/title\_activity\_splash\_\_screen"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 </**application**>  
  
</**manifest**>

# HomeActivity

The HomeActivivty presents the information about the available features in our application and helps user to navigate to the next activities.

HomeActivity.java

**package** com.vivartha.modechanger;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.app.Activity;  
**import** android.view.View;  
**import** android.widget.Button;  
  
**public class** Home\_Activity **extends** Activity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_home\_***);  
  
 Button btn = (Button) findViewById(R.id.***au***);  
 btn.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(Home\_Activity.**this**, about\_us.**class**);  
 startActivity(i);  
 }  
 });  
  
 Button btn1 = (Button) findViewById(R.id.***mc***);  
 btn1.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(Home\_Activity.**this**, MainActivity.**class**);  
 startActivity(i);  
 }  
 });  
  
 }

HomeActivity.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:background="@drawable/home"**>  
  
 <**LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_centerHorizontal="true"  
 android:layout\_centerVertical="true"**>  
  
 <**Button  
 android:layout\_width="200dp"  
 android:layout\_height="wrap\_content"  
 android:text="Mode changer"  
 android:id="@+id/mc"**/>  
  
 <**Button  
 android:layout\_width="200dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10.0sp"  
 android:text="How it Works"  
 android:id="@+id/au"**/>  
  
  
 </**LinearLayout**>  
  
  
</**RelativeLayout**>

# MainActivity

The MainActivity provides user to read the Default keywords and also allows user to edit the keywords.

MainActivity.java

**package** com.vivartha.modechanger;  
  
**import** android.app.Activity;  
**import** android.content.Intent;  
**import** android.content.SharedPreferences;  
**import** android.os.Bundle;  
**import** android.view.Menu;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**public class** MainActivity **extends** Activity {  
 SharedPreferences **preferences**;  
 SharedPreferences.Editor **editor**;  
 **private final** String **DEFAULT**=**""**;  
 EditText **r**,**v**,**s**;  
 Button **save**;  
 String **ring**,**vibrate**,**silent**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 *//Bind the fields* **r**=(EditText)findViewById(R.id.***editText1***);  
 **v**=(EditText)findViewById(R.id.***editText2***);  
 **s**=(EditText)findViewById(R.id.***editText3***);  
 *//vu=(EditText)findViewById(R.id.editText4);* **save** = (Button)findViewById(R.id.***button1***);  
 *//check for the shared preferences;* **preferences** = getSharedPreferences(**"modes"**, ***MODE\_PRIVATE***);  
 **ring** = **preferences**.getString(**"ring\_key"**, **DEFAULT**);  
 **vibrate** = **preferences**.getString(**"vibrate\_key"**, **DEFAULT**);  
 **silent** = **preferences**.getString(**"silent\_key"**, **DEFAULT**);  
 *//volumeup = preferences.getString("volume\_up", DEFAULT);  
  
 //This will set the keyword for RINGER MODE as ring if it is not configured by user* **if**(**ring**.equals(**DEFAULT**))  
 {  
 **editor** = **preferences**.edit();  
 **editor**.putString(**"ring\_key"**, **"ring"**);  
 **editor**.commit();  
 **ring** = **preferences**.getString(**"ring\_key"**, **DEFAULT**);  
 }  
  
 *//This will set the keyword for VIBRATE MODE as vibrate if it is not configured by user* **if**(**vibrate**.equals(**DEFAULT**))  
 {  
 **editor** = **preferences**.edit();  
 **editor**.putString(**"vibrate\_key"**, **"vibrate"**);  
 **editor**.commit();  
 **vibrate** = **preferences**.getString(**"vibrate\_key"**, **DEFAULT**);  
 }  
  
 *//This will set the keyword for SILENT MODE as silent if it is not configured by user* **if**(**silent**.equals(**DEFAULT**))  
 {  
 **editor** = **preferences**.edit();  
 **editor**.putString(**"silent\_key"**, **"silent"**);  
 **editor**.commit();  
 **silent** = **preferences**.getString(**"silent\_key"**, **DEFAULT**);  
 }  
  
 */\*if(ring.equals(DEFAULT)||vibrate.equals(DEFAULT)||silent.equals(DEFAULT)||volumeup.equals(DEFAULT)){  
 editor = preferences.edit();  
 editor.putString("ring\_key", "ring");  
 editor.putString("vibrate\_key", "vibrate");  
 editor.putString("silent\_key", "silent");  
 editor.putString("volume\_key", "volumeup");  
 editor.commit();  
 ring = preferences.getString("ring\_key", DEFAULT);  
 vibrate = preferences.getString("vibrate\_key", DEFAULT);  
 silent = preferences.getString("silent\_key", DEFAULT);  
 volumeup = preferences.getString("volumeup\_key", DEFAULT);  
 }\*/  
  
 //Setting keyword values to GUI layout* **r**.setText(**ring**);  
 **v**.setText(**vibrate**);  
 **s**.setText(**silent**);  
 *//vu.setText(volumeup);* **save**.setOnClickListener(**new** OnClickListener() {  
  
 @Override  
 **public void** onClick(View arg0) {  
 String temp\_ring = **r**.getText().toString().trim();  
 String temp\_vibrate = **v**.getText().toString().trim();  
 String temp\_silent = **s**.getText().toString().trim();  
 *//String temp\_volumeup = vu.getText().toString().trim();* **editor** = **preferences**.edit();  
 **editor**.putString(**"ring\_key"**, temp\_ring);  
 **editor**.putString(**"vibrate\_key"**, temp\_vibrate);  
 **editor**.putString(**"silent\_key"**, temp\_silent);  
 *//editor.putString("volumeup\_key", temp\_volumeup);* **editor**.commit();  
 Toast.*makeText*(getApplicationContext(), **"SAVED!"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
  
 *//Action to GoBack from edit screen to home screen* Button btn = (Button) findViewById(R.id.***btn\_goback***);  
 btn.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(MainActivity.**this**, Home\_Activity.**class**);  
 startActivity(i);  
 }  
 });  
 }  
  
 @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**;  
 }  
  
}

activity\_main.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:gravity="center"  
 android:orientation="vertical"  
 android:background="@drawable/settings"**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_marginTop="50dp"**>  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Keyword for Ring : "  
 android:textSize="20dp"**/>  
  
 <**EditText  
 android:id="@+id/editText1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="eg: RING"**>  
 <**requestFocus** />  
 </**EditText**>  
  
 </**LinearLayout**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_marginTop="20dp"**>  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Keyword for Vibrate : "  
 android:textSize="20dp"**/>  
  
 <**EditText  
 android:id="@+id/editText2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="eg: VIBRATE"** />  
  
 </**LinearLayout**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_marginTop="20dp"**>  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Keyword for Silent : "  
 android:textSize="20dp"**/>  
  
  
 <**EditText  
 android:id="@+id/editText3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="eg: SILENT"** />  
  
 </**LinearLayout**>  
  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"**>  
 <**Button  
 android:id="@+id/button1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="100dp"  
 android:text="SAVE"** />  
  
 <**Button  
 android:id="@+id/btn\_goback"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Go back"** />  
  
 </**LinearLayout**>  
  
  
</**LinearLayout**>

# AboutUs

The AboutUs provides the user with key information on how the application works.

about\_us.java

**package** com.vivartha.modechanger;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.app.Activity;  
**import** android.view.View;  
**import** android.widget.Button;  
  
**public class** about\_us **extends** Activity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_about\_us***);  
  
 Button btn = (Button)findViewById(R.id.***r5***);  
 btn.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(about\_us.**this**, Home\_Activity.**class**);  
 startActivity(i);  
 }  
 });  
 }  
  
}

activity\_about\_us.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout  
 android:orientation="vertical"  
 android:background="#fff0f0f0"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 xmlns:android="http://schemas.android.com/apk/res/android"**>  
  
  
 <**LinearLayout  
 android:orientation="vertical"  
 android:id="@id/w2"  
  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"**>  
  
  
 <**ImageView  
 android:layout\_width="fill\_parent"  
 android:layout\_height="90.0sp"  
 android:background="@drawable/header"** />  
  
 <**RelativeLayout  
 android:id="@id/v1"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5.0sp"**>  
  
 <**ImageView  
 android:id="@id/im1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10.0sp"  
 android:background="@drawable/phone"** />  
  
 <**TextView  
 android:id="@id/r1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginTop="25dp"  
 android:layout\_toRightOf="@id/im1"  
 android:text="Using ANY phone, Goto SMS Application."  
 android:textColor="#ff000000"** />  
 </**RelativeLayout**>  
  
 <**RelativeLayout  
 android:id="@id/v2"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5.0sp"  
 android:layout\_below="@id/v1"**>  
  
 <**ImageView  
 android:id="@id/im2"  
 android:background="@drawable/smso"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="7.0sp"** />  
  
 <**TextView  
 android:textColor="#ff000000"  
 android:id="@id/r2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="7.0sp"  
 android:layout\_marginTop="25dp"  
 android:text="In this SMS Application send KEYWORD(To the mode which you want to change) to your mobile."  
 android:layout\_toRightOf="@id/im2"** />  
  
 </**RelativeLayout**>  
  
 <**RelativeLayout  
 android:id="@id/v3"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5.0sp"  
 android:layout\_below="@id/v2"**>  
  
 <**ImageView  
 android:id="@id/im3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="7.0sp"  
 android:background="@drawable/smso"** />  
  
 <**TextView  
 android:id="@id/r3"  
 android:layout\_width="262dp"  
 android:layout\_height="55dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="16dp"  
 android:layout\_marginTop="23dp"  
 android:layout\_toRightOf="@id/im3"  
 android:text="The SMS Application in the Receivers Mobile reads the message and sends to our application"  
 android:textColor="#ff000000"** />  
 </**RelativeLayout**>  
  
 <**RelativeLayout  
 android:id="@id/v4"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5.0sp"  
 android:layout\_below="@id/v3"**>  
  
 <**ImageView  
 android:id="@id/im4"  
 android:background="@drawable/phone"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="7.0sp"** />  
  
 <**TextView  
 android:textColor="#ff000000"  
 android:id="@id/r4"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="25dp"  
 android:layout\_marginLeft="10.0sp"  
 android:text="Our Applications checks the keyword and changes to the mode you desired!"  
 android:layout\_toRightOf="@id/im4"** />  
  
 </**RelativeLayout**>  
  
 </**LinearLayout**>  
  
 <**Button  
 android:textSize="15.0sp"  
 android:textColor="#ffffffff"  
 android:id="@id/r5"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="40.0sp"  
 android:layout\_marginTop="20.0sp"  
 android:text="Go Back"  
 android:layout\_below="@id/w2"  
 android:layout\_centerHorizontal="true"** />  
  
</**RelativeLayout**>

# MyReceiver

The Myreceiver class runs in the background and reads the messages for the keywords and if the keyword matches with the applications value, then it performs the specified action.

MyReceiver.java

**package** com.vivartha.modechanger;  
  
**import** android.content.BroadcastReceiver;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.SharedPreferences;  
**import** android.media.AudioManager;  
**import** android.os.Bundle;  
**import** android.telephony.SmsMessage;  
**import** android.widget.Toast;  
  
**public class** MyReceiver **extends** BroadcastReceiver {  
 AudioManager **am**;  
 SharedPreferences **preferences**;  
 String **ring**,**vibrate**,**silent**,**voluming**;  
 **private final** String **DEFAULT**=**""**;  
 @Override  
 **public void** onReceive(Context context, Intent intent) {  
 **final** Bundle bundle = intent.getExtras();  
 **am** = (AudioManager) context.getSystemService(Context.***AUDIO\_SERVICE***);  
 **preferences** = context.getSharedPreferences(**"modes"**, Context.***MODE\_PRIVATE***);  
 **ring** = **preferences**.getString(**"ring\_key"**, **DEFAULT**);  
 **vibrate** = **preferences**.getString(**"vibrate\_key"**, **DEFAULT**);  
 **silent** = **preferences**.getString(**"silent\_key"**, **DEFAULT**);  
 **voluming** = **preferences**.getString(**"volume\_up"**, **DEFAULT**);  
  
  
 *// Reading SMS* **try** {  
 **if** (bundle != **null**) {  
 **final** Object[] pdusObj = (Object[]) bundle.get(**"pdus"**);  
  
 **for** (**int** i = 0; i < pdusObj.**length**; i++) {  
  
 SmsMessage currentMessage = SmsMessage  
 .*createFromPdu*((**byte**[]) pdusObj[i]);  
 String actual\_message = currentMessage.getDisplayMessageBody();  
 String message = getFirstWord(actual\_message);  
 **int** status = changeMode(message);  
 **switch** (status) {  
 **case** 1:  
 Toast.*makeText*(context, **"RING\_MODE"**, Toast.***LENGTH\_LONG***).show();  
 **break**;  
 **case** 2:  
 Toast.*makeText*(context, **"SILENT\_MODE"**, Toast.***LENGTH\_LONG***).show();  
 **break**;  
 **case** 3:  
 Toast.*makeText*(context, **"VIBRATE\_MODE"**, Toast.***LENGTH\_LONG***).show();  
 **break**;  
 **case** 4:  
 Toast.*makeText*(context, **"VOLUME\_UP"**, Toast.***LENGTH\_LONG***).show();  
 **default**:  
 **break**;  
 }  
 }  
 }  
 } **catch** (Exception e) {  
 *//* ***TODO: handle exception*** }  
 *// Change Mode* }  
 **private** String getFirstWord(String text) {  
 **if** (text.indexOf(**' '**) > -1) {  
 **return** text.substring(0, text.indexOf(**' '**));  
 } **else** {  
 **return** text;  
 }  
 }  
 **private int** changeMode(String receivedMessage) {  
 **if** (receivedMessage.equalsIgnoreCase(**ring**)) {  
 **am**.setRingerMode(AudioManager.***RINGER\_MODE\_NORMAL***);  
 **return** 1;  
 } **else if** (receivedMessage.equalsIgnoreCase(**silent**)) {  
 **am**.setRingerMode(AudioManager.***RINGER\_MODE\_SILENT***);  
 **return** 2;  
 } **else if**(receivedMessage.equalsIgnoreCase(**vibrate**)){  
 **am**.setRingerMode(AudioManager.***RINGER\_MODE\_VIBRATE***);  
 **return** 3;  
 }  
 **else if**(receivedMessage.equalsIgnoreCase(**voluming**)){  
 **am**.setStreamVolume(AudioManager.***STREAM\_MUSIC***,  
 **am**.getStreamMaxVolume(AudioManager.***STREAM\_MUSIC***),  
 0);  
 *//am.setStreamVolume(AudioManager.STREAM\_MUSIC, am.getStreamMaxVolume(AudioManager.STREAM\_MUSIC),0);* **return** 4;  
 }  
 **return** 0;  
 }  
}

# SplashScreen

The SplashScreen provides the user basic information about the project(i.e, Name and developed by, etc).

Splash\_Screen.java

**package** com.vivartha.modechanger;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.app.Activity;  
  
**public class** Splash\_Screen **extends** Activity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_splash\_\_screen***);  
  
 Intent intent = **new** Intent(getApplicationContext(),  
 LoginActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
  
}

activity\_splash\_screen.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout 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=".Splash\_Screen"  
 android:background="@drawable/splashscreen"**>  
  
</**android.support.constraint.ConstraintLayout**>

# LoginActivity

The LoginActivity enables user to provide the credentials and validates the provided credentials. Here, since this is not included in this phase we just included screen as it is the first screen. We are not validating the provided credentials.

LoginActivity.java

**package** com.vivartha.modechanger;  
  
**import** android.animation.Animator;  
**import** android.animation.AnimatorListenerAdapter;  
**import** android.annotation.TargetApi;  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.support.annotation.NonNull;  
**import** android.support.design.widget.Snackbar;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.app.LoaderManager.LoaderCallbacks;  
  
**import** android.content.CursorLoader;  
**import** android.content.Loader;  
**import** android.database.Cursor;  
**import** android.net.Uri;  
**import** android.os.AsyncTask;  
  
**import** android.os.Build;  
**import** android.os.Bundle;  
**import** android.provider.ContactsContract;  
**import** android.text.TextUtils;  
**import** android.view.KeyEvent;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.view.inputmethod.EditorInfo;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.AutoCompleteTextView;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**import static** android.Manifest.permission.***READ\_CONTACTS***;  
  
*/\*\*  
 \* A login screen that offers login via email/password.  
 \*/***public class** LoginActivity **extends** AppCompatActivity **implements** LoaderCallbacks<Cursor> {  
  
 */\*\*  
 \* Id to identity READ\_CONTACTS permission request.  
 \*/* **private static final int *REQUEST\_READ\_CONTACTS*** = 0;  
  
 */\*\*  
 \* A dummy authentication store containing known user names and passwords.  
 \** ***TODO: remove after connecting to a real authentication system.*** *\*/* **private static final** String[] ***DUMMY\_CREDENTIALS*** = **new** String[]{  
 **"foo@example.com:hello"**, **"bar@example.com:world"** };  
 */\*\*  
 \* Keep track of the login task to ensure we can cancel it if requested.  
 \*/* **private** UserLoginTask **mAuthTask** = **null**;  
  
 *// UI references.* **private** AutoCompleteTextView **mEmailView**;  
 **private** EditText **mPasswordView**;  
 **private** View **mProgressView**;  
 **private** View **mLoginFormView**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***);  
 *// Set up the login form.* **mEmailView** = (AutoCompleteTextView) findViewById(R.id.***email***);  
 populateAutoComplete();  
  
 **mPasswordView** = (EditText) findViewById(R.id.***password***);  
 **mPasswordView**.setOnEditorActionListener(**new** TextView.OnEditorActionListener() {  
 @Override  
 **public boolean** onEditorAction(TextView textView, **int** id, KeyEvent keyEvent) {  
 **if** (id == EditorInfo.***IME\_ACTION\_DONE*** || id == EditorInfo.***IME\_NULL***) {  
 attemptLogin();  
 **return true**;  
 }  
 **return false**;  
 }  
 });  
  
 Button mEmailSignInButton = (Button) findViewById(R.id.***email\_sign\_in\_button***);  
 mEmailSignInButton.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(LoginActivity.**this**,Home\_Activity.**class**);  
 startActivity(i);  
 }  
 });  
 }  
  
 **private void** populateAutoComplete() {  
 **if** (!mayRequestContacts()) {  
 **return**;  
 }  
  
 getLoaderManager().initLoader(0, **null**, **this**);  
 }  
  
 **private boolean** mayRequestContacts() {  
 **if** (Build.VERSION.***SDK\_INT*** < Build.VERSION\_CODES.***M***) {  
 **return true**;  
 }  
 **if** (checkSelfPermission(***READ\_CONTACTS***) == PackageManager.***PERMISSION\_GRANTED***) {  
 **return true**;  
 }  
 **if** (shouldShowRequestPermissionRationale(***READ\_CONTACTS***)) {  
 Snackbar.*make*(**mEmailView**, R.string.***permission\_rationale***, Snackbar.***LENGTH\_INDEFINITE***)  
 .setAction(android.R.string.***ok***, **new** View.OnClickListener() {  
 @Override  
 @TargetApi(Build.VERSION\_CODES.***M***)  
 **public void** onClick(View v) {  
 requestPermissions(**new** String[]{***READ\_CONTACTS***}, ***REQUEST\_READ\_CONTACTS***);  
 }  
 });  
 } **else** {  
 requestPermissions(**new** String[]{***READ\_CONTACTS***}, ***REQUEST\_READ\_CONTACTS***);  
 }  
 **return false**;  
 }  
  
 */\*\*  
 \* Callback received when a permissions request has been completed.  
 \*/* @Override  
 **public void** onRequestPermissionsResult(**int** requestCode, @NonNull String[] permissions,  
 @NonNull **int**[] grantResults) {  
 **if** (requestCode == ***REQUEST\_READ\_CONTACTS***) {  
 **if** (grantResults.**length** == 1 && grantResults[0] == PackageManager.***PERMISSION\_GRANTED***) {  
 populateAutoComplete();  
 }  
 }  
 }  
  
  
 */\*\*  
 \* Attempts to sign in or register the account specified by the login form.  
 \* If there are form errors (invalid email, missing fields, etc.), the  
 \* errors are presented and no actual login attempt is made.  
 \*/* **private void** attemptLogin() {  
 **if** (**mAuthTask** != **null**) {  
 **return**;  
 }  
  
 *// Reset errors.* **mEmailView**.setError(**null**);  
 **mPasswordView**.setError(**null**);  
  
 *// Store values at the time of the login attempt.* String email = **mEmailView**.getText().toString();  
 String password = **mPasswordView**.getText().toString();  
  
 **boolean** cancel = **false**;  
 View focusView = **null**;  
  
 *// Check for a valid password, if the user entered one.* **if** (!TextUtils.*isEmpty*(password) && !isPasswordValid(password)) {  
 **mPasswordView**.setError(getString(R.string.***error\_invalid\_password***));  
 focusView = **mPasswordView**;  
 cancel = **true**;  
 }  
  
 *// Check for a valid email address.* **if** (TextUtils.*isEmpty*(email)) {  
 **mEmailView**.setError(getString(R.string.***error\_field\_required***));  
 focusView = **mEmailView**;  
 cancel = **true**;  
 } **else if** (!isEmailValid(email)) {  
 **mEmailView**.setError(getString(R.string.***error\_invalid\_email***));  
 focusView = **mEmailView**;  
 cancel = **true**;  
 }  
  
 **if** (cancel) {  
 *// There was an error; don't attempt login and focus the first  
 // form field with an error.* focusView.requestFocus();  
 } **else** {  
 *// Show a progress spinner, and kick off a background task to  
 // perform the user login attempt.* showProgress(**true**);  
 **mAuthTask** = **new** UserLoginTask(email, password);  
 **mAuthTask**.execute((Void) **null**);  
 }  
 }  
  
 **private boolean** isEmailValid(String email) {  
 *//****TODO: Replace this with your own logic* return** email.contains(**"@"**);  
 }  
  
 **private boolean** isPasswordValid(String password) {  
 *//****TODO: Replace this with your own logic* return** password.length() > 4;  
 }  
  
 */\*\*  
 \* Shows the progress UI and hides the login form.  
 \*/* @TargetApi(Build.VERSION\_CODES.***HONEYCOMB\_MR2***)  
 **private void** showProgress(**final boolean** show) {  
 *// On Honeycomb MR2 we have the ViewPropertyAnimator APIs, which allow  
 // for very easy animations. If available, use these APIs to fade-in  
 // the progress spinner.* **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***HONEYCOMB\_MR2***) {  
 **int** shortAnimTime = getResources().getInteger(android.R.integer.***config\_shortAnimTime***);  
  
 **mLoginFormView**.setVisibility(show ? View.***GONE*** : View.***VISIBLE***);  
 **mLoginFormView**.animate().setDuration(shortAnimTime).alpha(  
 show ? 0 : 1).setListener(**new** AnimatorListenerAdapter() {  
 @Override  
 **public void** onAnimationEnd(Animator animation) {  
 **mLoginFormView**.setVisibility(show ? View.***GONE*** : View.***VISIBLE***);  
 }  
 });  
  
 **mProgressView**.setVisibility(show ? View.***VISIBLE*** : View.***GONE***);  
 **mProgressView**.animate().setDuration(shortAnimTime).alpha(  
 show ? 1 : 0).setListener(**new** AnimatorListenerAdapter() {  
 @Override  
 **public void** onAnimationEnd(Animator animation) {  
 **mProgressView**.setVisibility(show ? View.***VISIBLE*** : View.***GONE***);  
 }  
 });  
 } **else** {  
 *// The ViewPropertyAnimator APIs are not available, so simply show  
 // and hide the relevant UI components.* **mProgressView**.setVisibility(show ? View.***VISIBLE*** : View.***GONE***);  
 **mLoginFormView**.setVisibility(show ? View.***GONE*** : View.***VISIBLE***);  
 }  
 }  
  
 @Override  
 **public** Loader<Cursor> onCreateLoader(**int** i, Bundle bundle) {  
 **return new** CursorLoader(**this**,  
 *// Retrieve data rows for the device user's 'profile' contact.* Uri.*withAppendedPath*(ContactsContract.Profile.***CONTENT\_URI***,  
 ContactsContract.Contacts.Data.***CONTENT\_DIRECTORY***), ProfileQuery.***PROJECTION***,  
  
 *// Select only email addresses.* ContactsContract.Contacts.Data.***MIMETYPE*** +  
 **" = ?"**, **new** String[]{ContactsContract.CommonDataKinds.Email  
 .***CONTENT\_ITEM\_TYPE***},  
  
 *// Show primary email addresses first. Note that there won't be  
 // a primary email address if the user hasn't specified one.* ContactsContract.Contacts.Data.***IS\_PRIMARY*** + **" DESC"**);  
 }  
  
 @Override  
 **public void** onLoadFinished(Loader<Cursor> cursorLoader, Cursor cursor) {  
 List<String> emails = **new** ArrayList<>();  
 cursor.moveToFirst();  
 **while** (!cursor.isAfterLast()) {  
 emails.add(cursor.getString(ProfileQuery.***ADDRESS***));  
 cursor.moveToNext();  
 }  
  
 addEmailsToAutoComplete(emails);  
 }  
  
 @Override  
 **public void** onLoaderReset(Loader<Cursor> cursorLoader) {  
  
 }  
  
 **private void** addEmailsToAutoComplete(List<String> emailAddressCollection) {  
 *//Create adapter to tell the AutoCompleteTextView what to show in its dropdown list.* ArrayAdapter<String> adapter =  
 **new** ArrayAdapter<>(LoginActivity.**this**,  
 android.R.layout.***simple\_dropdown\_item\_1line***, emailAddressCollection);  
  
 **mEmailView**.setAdapter(adapter);  
 }  
  
  
 **private interface** ProfileQuery {  
 String[] ***PROJECTION*** = {  
 ContactsContract.CommonDataKinds.Email.***ADDRESS***,  
 ContactsContract.CommonDataKinds.Email.***IS\_PRIMARY***,  
 };  
  
 **int *ADDRESS*** = 0;  
 **int *IS\_PRIMARY*** = 1;  
 }  
  
 */\*\*  
 \* Represents an asynchronous login/registration task used to authenticate  
 \* the user.  
 \*/* **public class** UserLoginTask **extends** AsyncTask<Void, Void, Boolean> {  
  
 **private final** String **mEmail**;  
 **private final** String **mPassword**;  
  
 UserLoginTask(String email, String password) {  
 **mEmail** = email;  
 **mPassword** = password;  
 }  
  
 @Override  
 **protected** Boolean doInBackground(Void... params) {  
 *//* ***TODO: attempt authentication against a network service.* try** {  
 *// Simulate network access.* Thread.*sleep*(2000);  
 } **catch** (InterruptedException e) {  
 **return false**;  
 }  
  
 **for** (String credential : ***DUMMY\_CREDENTIALS***) {  
 String[] pieces = credential.split(**":"**);  
 **if** (pieces[0].equals(**mEmail**)) {  
 *// Account exists, return true if the password matches.* **return** pieces[1].equals(**mPassword**);  
 }  
 }  
  
 *//* ***TODO: register the new account here.* return true**;  
 }  
  
 @Override  
 **protected void** onPostExecute(**final** Boolean success) {  
 **mAuthTask** = **null**;  
 showProgress(**false**);  
  
 **if** (success) {  
 finish();  
 } **else** {  
 **mPasswordView**.setError(getString(R.string.***error\_incorrect\_password***));  
 **mPasswordView**.requestFocus();  
 }  
 }  
  
 @Override  
 **protected void** onCancelled() {  
 **mAuthTask** = **null**;  
 showProgress(**false**);  
 }  
 }  
}

activity\_login.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout 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:gravity="center\_horizontal"  
 android:orientation="vertical"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 tools:context=".LoginActivity"  
 android:layout\_centerHorizontal="true"  
 android:layout\_centerVertical="true"  
 android:background="@drawable/login"**>  
  
 *<!-- Login progress -->* <**ProgressBar  
 android:id="@+id/login\_progress"  
 style="?android:attr/progressBarStyleLarge"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="8dp"  
 android:visibility="gone"** />  
  
  
  
 <**LinearLayout  
 android:id="@+id/email\_login\_form"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_marginTop="200dp"**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**AutoCompleteTextView  
 android:id="@+id/email"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Email (optional)"  
 android:inputType="textEmailAddress"  
 android:maxLines="1"  
 android:singleLine="true"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**EditText  
 android:id="@+id/password"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_password"  
 android:imeActionId="6"  
 android:imeActionLabel="@string/action\_sign\_in\_short"  
 android:imeOptions="actionUnspecified"  
 android:inputType="textPassword"  
 android:maxLines="1"  
 android:singleLine="true"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**Button  
 android:id="@+id/email\_sign\_in\_button"  
 style="?android:textAppearanceSmall"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="16dp"  
 android:text="@string/action\_sign\_in"  
 android:textStyle="bold"** />  
  
 </**LinearLayout**>  
  
</**LinearLayout**>

**Values**

The values Directory contains already defined values such as id, strings, colors, dimens, style. We can directly inherit these values into the required classes.

colors.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**resources**>  
 <**color name="colorPrimary"**>#008577</**color**>  
 <**color name="colorPrimaryDark"**>#00574B</**color**>  
 <**color name="colorAccent"**>#D81B60</**color**>  
</**resources**>

dimens.xml

<**resources**>  
 *<!-- Default screen margins, per the Android Design guidelines. -->* <**dimen name="activity\_horizontal\_margin"**>16dp</**dimen**>  
 <**dimen name="activity\_vertical\_margin"**>16dp</**dimen**>  
 <**dimen name="fab\_margin"**>16dp</**dimen**>  
</**resources**>

ids.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**resources**>  
 <**item name="w1" type="id"**></**item**>  
 <**item name="w2" type="id"**></**item**>  
 <**item name="v1" type="id"**></**item**>  
 <**item name="im1" type="id"**></**item**>  
 <**item name="r1" type="id"**></**item**>  
 <**item name="v2" type="id"**></**item**>  
 <**item name="im2" type="id"**></**item**>  
 <**item name="r2" type="id"**></**item**>  
 <**item name="v3" type="id"**></**item**>  
 <**item name="im3" type="id"**></**item**>  
 <**item name="r3" type="id"**></**item**>  
 <**item name="v4" type="id"**></**item**>  
 <**item name="im4" type="id"**></**item**>  
 <**item name="r4" type="id"**></**item**>  
 <**item name="r5" type="id"**></**item**>  
</**resources**>

strings.xml

<**resources**>  
 <**string name="app\_name"**>ModeChanger</**string**>  
 *<!-- Strings related to login -->* <**string name="prompt\_email"**>Email</**string**>  
 <**string name="prompt\_password"**>Password (optional)</**string**>  
 <**string name="action\_sign\_in"**>Sign in or register</**string**>  
 <**string name="action\_sign\_in\_short"**>Sign in</**string**>  
 <**string name="error\_invalid\_email"**>This email address is invalid</**string**>  
 <**string name="error\_invalid\_password"**>This password is too short</**string**>  
 <**string name="error\_incorrect\_password"**>This password is incorrect</**string**>  
 <**string name="action\_settings"**>Settings</**string**>  
 <**string name="error\_field\_required"**>This field is required</**string**>  
 <**string name="permission\_rationale"**>"Contacts permissions are needed for providing email  
 completions."  
 </**string**>  
 <**string name="title\_activity\_home"**>homeActivity</**string**>  
 <**string name="title\_activity\_about\_us"**>about\_us</**string**>  
 <**string name="title\_activity\_home\_"**>Home\_Activity</**string**>  
 <**string name="title\_activity\_splash\_\_screen"**>Splash\_Screen</**string**>  
</**resources**>

styles.xml

<**resources**>  
  
 *<!-- Base application theme. -->* <**style name="AppTheme" parent="Theme.AppCompat.NoActionBar"**>  
 *<!-- Customize your theme here. -->* <**item name="colorPrimary"**>@color/colorPrimary</**item**>  
 <**item name="colorPrimaryDark"**>@color/colorPrimaryDark</**item**>  
 <**item name="colorAccent"**>@color/colorAccent</**item**>  
 </**style**>  
  
 <**style name="Theme.AppCompat.NoActionBar"**>  
 <**item name="windowActionBar"**>false</**item**>  
 <**item name="windowNoTitle"**>true</**item**>  
 </**style**>  
  
 <**style name="AppTheme.AppBarOverlay" parent="Theme.AppCompat.NoActionBar"** />  
  
 <**style name="AppTheme.PopupOverlay" parent="Theme.AppCompat.NoActionBar"** />  
 <**style name = "NoActionBar" parent = "@android:style/Theme.Holo.Light"**>  
 <**item name = "android:windowActionBar"**>false</**item**>  
 <**item name = "android:windowNoTitle"**>true</**item**>  
 </**style**>  
 <**style name="SplashScreenTheme" parent="Theme.AppCompat.Light.DarkActionBar"**>  
 <**item name ="android:windowBackground"**> @drawable/splashscreen</**item**>  
 </**style**>  
  
</**resources**>