Assignment 1 - Email Emulator

COMP-47330 Practical Android Programming

Author *:*	*Paula Dwan* *paula.dwan@gmail.com*
Student ID :	*13208660*
Course :	*MSc Advanced Software Engineering*
Submission date :	*4 Mar 2015*
FAO :	*Tadhg O'Sullivan & Barnard Kroon* *t.OSullivan@ucd.ie & barnard.kroon@ucdconnect.ie*

Table of Contents

ASSIGNMENT 1 – REQUIREMENTS

SUMMARY OF REQUIREMENTS

Link https://csimoodle.ucd.ie/moodle/mod/assign/view.php?id=26293

Overview You are implementing an email app (doesn't need to actually email just display text in another activity) **.** Design two layouts for two different activities : (https://csimoodle.ucd.ie/moodle/mod/resource/view.php?id=23231)

1. email composition layout
2. email reading layout

Email composition should allow entering of the following fields:

- from
- to
- cc
- bcc
- subject
- email body

It should also feature two buttons: **clear** & **send**.

Email reading layout should feature exactly the same fields for display, not editing**.** '**bcc**' field should be missing though**.** The 'email reading' layout should also feature just one button, *i.e***.** "**Back**".

Feel free to use your favourite type of layout for this, i.e**.** table, linear, or relative.

Essential 1 Once you are done, create two activities : https://csimoodle.ucd.ie/moodle/mod/resource/view.php?id=23231 (one for each layout)

and make your application can do the following.

1. Your application should start in email editing activity:
   - user should be allowed to enter information in the above fields
   - pressing 'clear' button should clear the information in all the fields
   - pressing 'send' button should create an intent to call a second activity, *i.e***.** Reading Email

Essential 2

2. Email reading activity should be started by explicit intent from editing activity:
   - user should be able to see the whole email (so make sure the view is scrollable)
   - pressing 'Back' button should take user back to the email composing view

VERY importANT!

Your application should preserve all the input data! If your activity is stopped and restarted again, all entered user data should be preserved in it!

While Android helps you out a lot by preserving the state of text fields automatically, make sure you use **savedInstanceState** *in your code to preserve the state of your activities (https://csimoodle.ucd.ie/moodle/mod/resource/view.php?id=23231)***.**

*(In some cases people have had trouble using **savedInstanceState** so using **SharedPreference** instead is an acceptable solution.)*

SUBMISSION DETAILS / DEAD-LINE

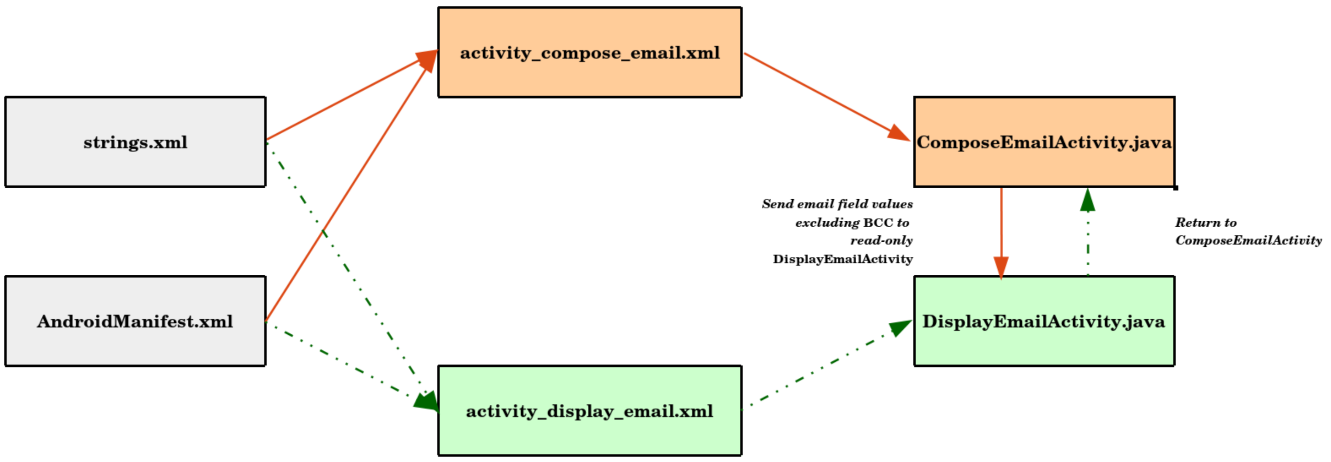
This assignment will accept submissions from Tuesday, 24 February 2015, 9:35 am

Due date	Wednesday, 4 March 2015, 11:55 pm

Assignment 1 – Details & Coding

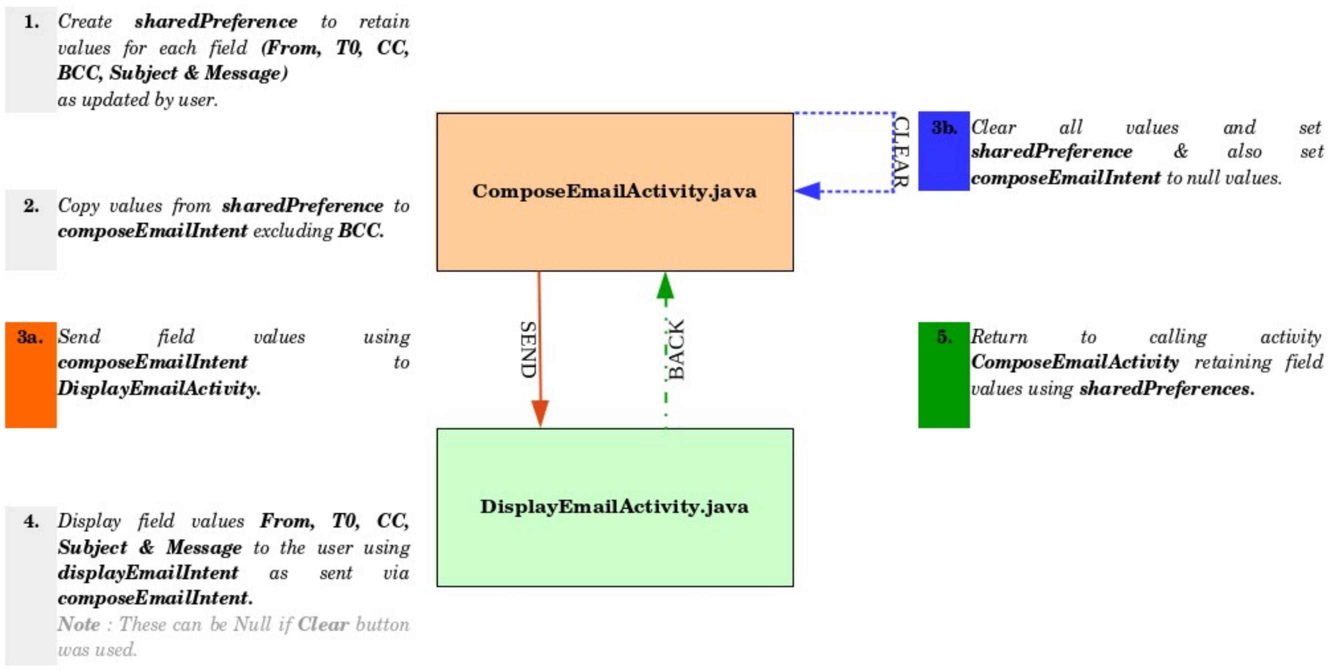
Code Implemented : Structure

Structure used is explained in the following diagram.


activity_compose_email.xml
strings.xml
ComposeEmailActivity.java
Send email field values excluding BCC to read-only DisplayEmailActivity
Return to ComposeEmailActivity
AndroidManifest.xml
DisplayEmailActivity.java
activity_display_email.xml

Code Flow

Flow of values for each email data field (*FROM*, *TO*, *CC*, *BCC*, *SUBJECT* and *MESSAGE*) is explained in the following diagram as are buttons used (*SEND*, *CLEAR* and *BACK*).



1. *Create* ***sharedPreference*** *to retain values for each field* ***(From, T0, CC, BCC, Subject & Message)*** *as updated by user.*

2. *Copy values from* ***sharedPreference*** *to* ***composeEmailIntent*** *excluding* ***BCC.***

3a. *Send field values using* ***composeEmailIntent*** *to* ***DisplayEmailActivity.***

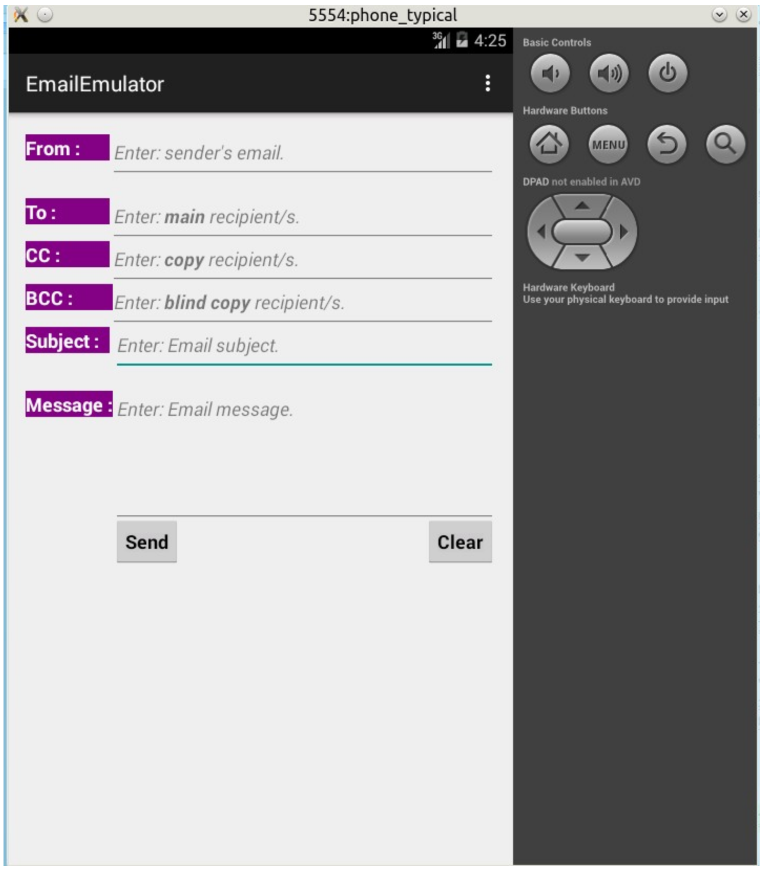
3b. *Clear all values and set* ***sharedPreference*** *& also set* ***composeEmailIntent*** *to null values.*

4. *Display field values* ***From, T0, CC, Subject & Message*** *to the user using* ***displayEmailIntent*** *as sent via* ***composeEmailIntent.***
Note : These can be Null if ***Clear*** *button was used.*

5. *Return to calling activity* ***ComposeEmailActivity*** *retaining field values using* ***sharedPreferences.***

SAMPLE IMPLEMENTATION

PHONE_TYPICAL



From : Enter: sender's email.

To : Enter: **main** recipient/s.

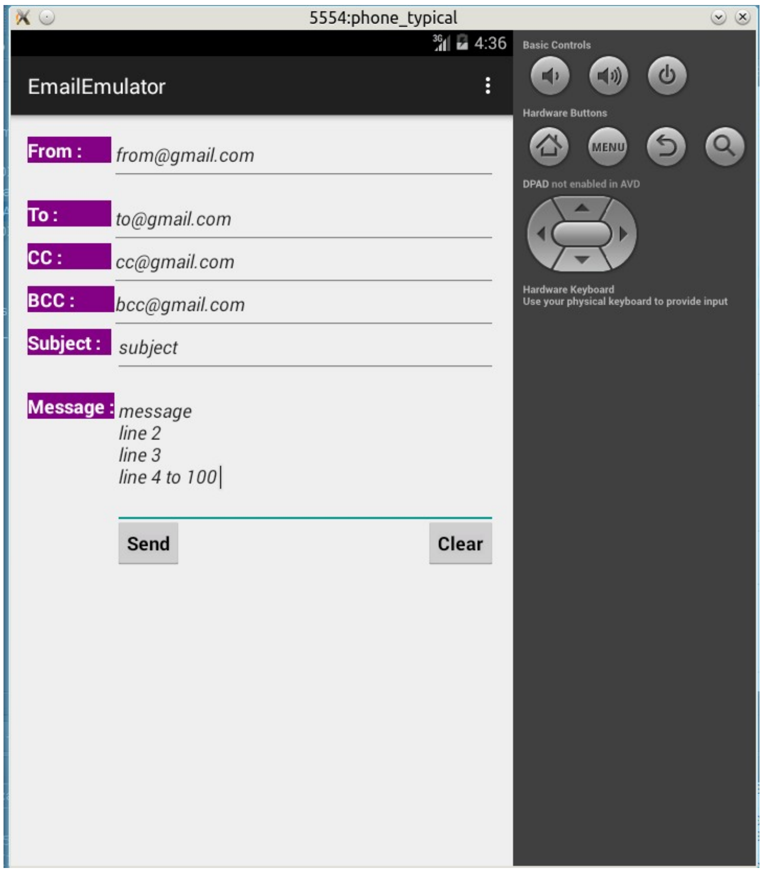
CC : Enter: **copy** recipient/s.

BCC : Enter: **blind copy** recipient/s.

Subject : Enter: Email subject.

Message : Enter: Email message.

Figure 1 : DisplayEmailActivity - first usage ic. hints.



From : from@gmail.com

To : to@gmail.com

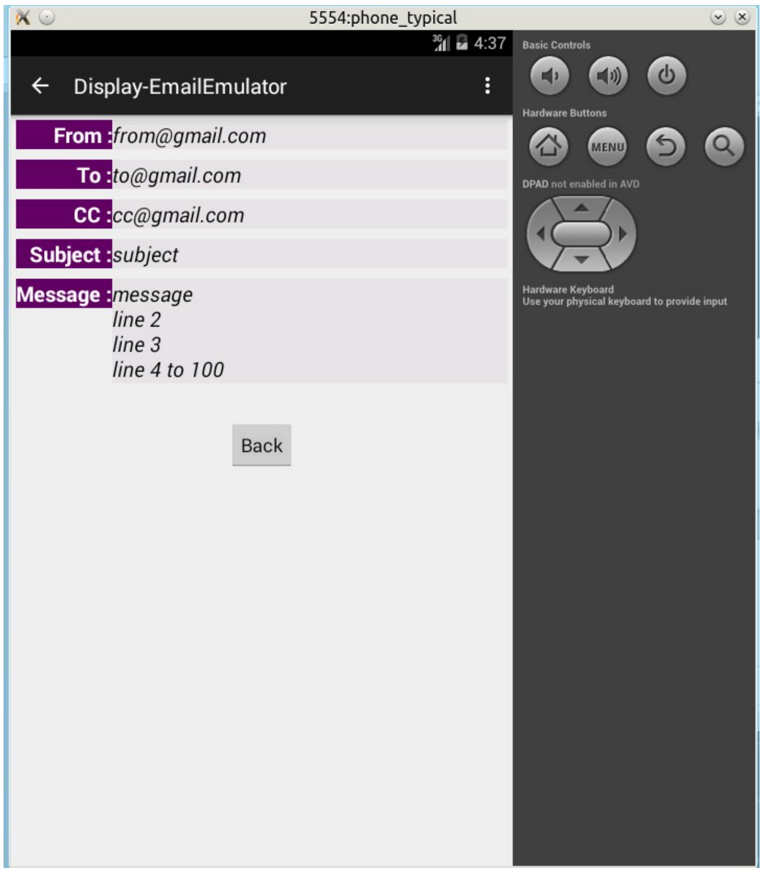
CC : cc@gmail.com

BCC : bcc@gmail.com

Subject : subject

Message : message
line 2
line 3
line 4 to 100

Figure 2 : DisplayEmailActivity - values entered into fields.



5554:phone_typical

Display-EmailEmulator

From : from@gmail.com

To : to@gmail.com

CC : cc@gmail.com

Subject : subject

Message : message
line 2
line 3
line 4 to 100

Figure 3 : DisplayEmailActivity - values as entered in ComposeEmailActivity (no BCC).

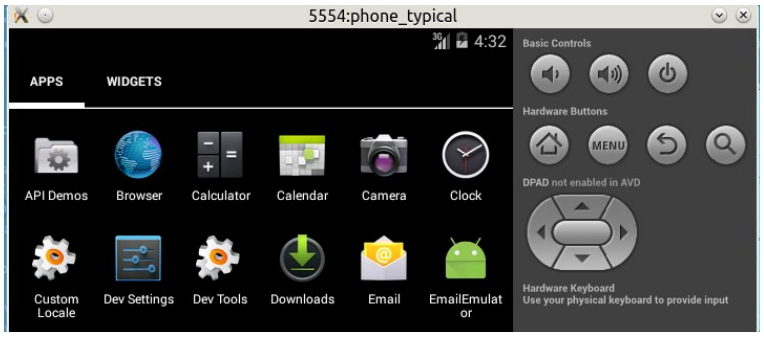


Figure 4 : EmailEmulator - move app to trash and relaunch (or close ADT)...

PHONE_BUDGET

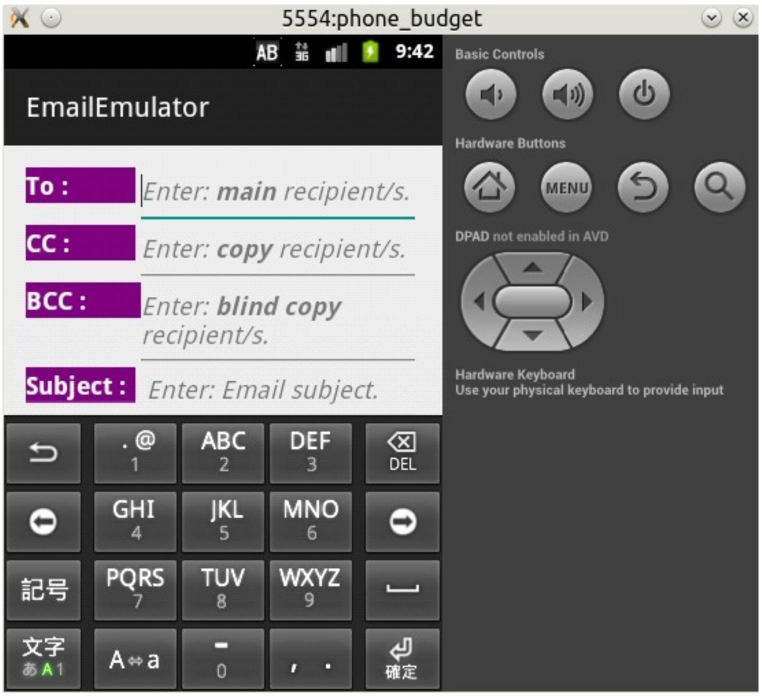


Figure 5 : ComposeEmailActivity - first usage with scrolling & keypad input : BUDGET.

PHONE_TABLET

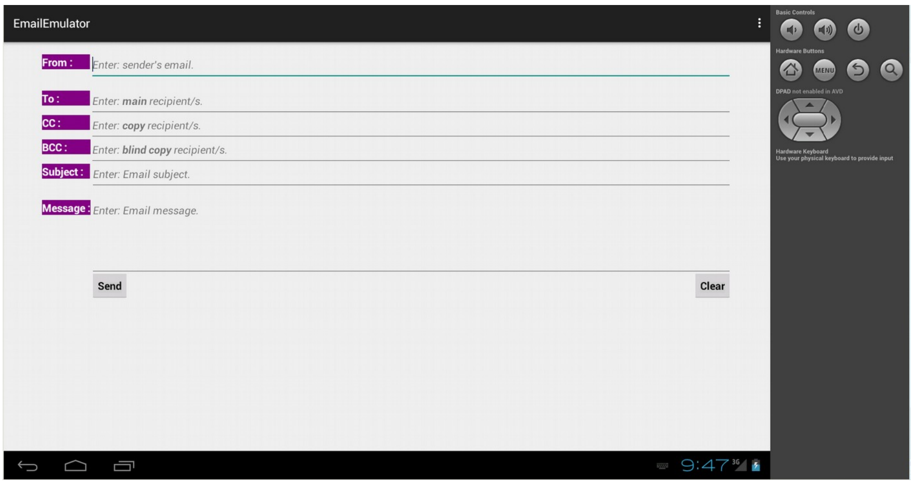


Figure 6 : ComposeEmailActivity - first usage with scrolling & keypad input : TABLET.

APPENDICES

APPENDIX I – CODE : STRINGS.XML

```
<?xml version ="1.0" encoding="utf-8"?>

<resources>
    <string name="app_name">EmailEmulator</string>
    <string name="app_logo">EE</string>
    <string name="title_activity_display_email">Display-EmailEmulator</string>

    <string name="action_settings">Settings</string>

    <string name="hint_from">Enter: sender\'s email.</string>
    <string name="hint_to">Enter: <b>main</b> recipient/s.</string>
    <string name="hint_cc">Enter: <b>copy</b> recipient/s.</string>
    <string name="hint_bcc">Enter: <b>blind copy</b> recipient/s.</string>
    <string name="hint_subject">Enter: Email subject.</string>
    <string name="hint_message">Enter: Email message.</string>

    <string name="EMAIL_FROM">From :</string>
    <string name="EMAIL_TO">To :</string>
    <string name="EMAIL_BCC">BCC :</string>
    <string name="EMAIL_CC">CC :</string>
    <string name="EMAIL_SUBJECT">Subject :</string>
    <string name="EMAIL_MESSAGE">Message :</string>

    <string name="btn_send_email">Send</string>
    <string name="btn_clear_email">Clear</string>
    <string name="btn_back_email">Back</string>

    <string name="error_email_null_value">Info : Nothing entered!</string>
    <string name="error_nok_email">Error : Invalid email entered!</string>
    <string name="error_nok_EMAIL_CC">Warning : no <b><i>copy</i></b> recipient/s entered!</string>
    <string name="error_nok_EMAIL_BCC">Warning : no <b><i>blind copy</i></b> recipient/s entered!</string>
    <string name="error_nok_subject">Warning : no subject entered!</string>
    <string name="error_nok_message">Warning : no message entered!</string>

</resources>
```

APPENDIX II – CODE : ANDROIDMANIFEST.XML

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.dwan.paula.a01_emailemulator" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".ComposeEmailActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>name="org.dwan.paula.a01_emailemulator.DisplayEmailActivity"
            android:label="@string/title_activity_display_email"
            android:parentActivityName="org.dwan.paula.a01_emailemulator.ComposeEmailActivity" >
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="org.dwan.paula.a01_emailemulator.ComposeEmailActivity" />
        </activity>
    </application>

</manifest>
```

APPENDIX III – CODE : ACTIVITY_COMPOSE_EMAIL.XML

```
<?xml version ="1.0" encoding="utf-8"?>

<ScrollView
    android:id="@+id/scrollViewComposeEmail"
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    tools:context=".DisplayContact" >

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="450dp"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin" >

        <TextView
            android:id="@+id/textViewemailFrom"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignTop="@+id/editTextemailFrom"
            android:layout_alignParentLeft="true"
            android:text="@string/EMAIL_FROM"
            android:textColor="#ffffff"
            android:textStyle="bold"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:background="#800080"
            android:layout_alignRight="@+id/textViewemailSubject"
            android:layout_alignEnd="@+id/textViewemailSubject"> </TextView>
        <EditText
            android:id="@+id/editTextemailFrom"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:layout_alignParentLeft="true"
            android:layout_marginTop="5dp"
            android:layout_marginLeft="80dp"
            android:ems="10"
            android:textStyle="italic"
            android:inputType="text|textEmailAddress"
            android:hint="@string/hint_from"
            android:layout_alignRight="@+id/btnClearEmail"
            android:layout_alignEnd="@+id/btnClearEmail"> <requestFocus /> </EditText>

<!--
    same for : to, cc, bcc, subject & message, except for layout usage as aligns with other fields
    -->

        <Button
            android:id="@+id/btnSendEmail"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:onClick="sendWrittenEmail"
            android:textStyle="bold"
            android:text="@string/btn_send_email"
            android:layout_alignParentBottom="true"
            android:layout_toRightOf="@+id/textViewemailMessage"
            android:layout_toEndOf="@+id/textViewemailMessage"> </Button>
        <Button
            android:id="@+id/btnClearEmail"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:onClick="clearWrittenEmail"
            android:textStyle="bold"
            android:text="@string/btn_clear_email"
            android:layout_alignParentBottom="true"
            android:layout_alignRight="@+id/editTextemailMessage"
            android:layout_alignEnd="@+id/editTextemailMessage"> </Button>

    </RelativeLayout>
</ScrollView>
```

APPENDIX IV – CODE : ACTIVITY_DISPLAY_EMAIL.XML

```
<?xml version ="1.0" encoding="utf-8"?>

<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/linearLayoutEmail"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <TableLayout
        android:id="@+id/tableLayoutEmail"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:stretchColumns="2"
        android:shrinkColumns="1" >

        <TableRow
            android:id="@+id/tableRowFrom"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_margin="5dp">
            <TextView
```

```
                android:id="@+id/textViewFromTitle"
                android:layout_row="1"
                android:layout_column="1"
                android:textColor="#ffffff"
                android:background="#660066"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:gravity="right"
                android:textAlignment="gravity"
                android:textSize="20sp"
                android:textStyle="bold"
                android:text="@string/EMAIL_FROM" ></TextView>
            <TextView
                android:id="@+id/textViewFromValue"
                android:layout_row="1"
                android:layout_column="2"
                android:textColor="#000000"
                android:background="#e1e1e1"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:textSize="20sp"
                android:textStyle="italic"
                android:text="" > </TextView>
        </TableRow>

<!--
    Same for : to, cc, subject & message
    -->

        <TableRow android:id="@+id/tableRowButtons"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_span="2"
            android:gravity="center"
            android:padding="30dp">
                <Button
                    android:id="@+id/btnBackEmail"
                    android:text="@string/btn_back_email"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center_horizontal"
                    android:padding="5dp"
                    android:onClick="backToEMailWriter"
                    android:layout_column="1"> </Button>
        </TableRow>

    </TableLayout>

</ScrollView>
```

Appendix V – Code : ComposeEmailActivity.java

```
package org.dwan.paula.a01_emailemulator;

import android.content.Intent;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.TextView;
import android.util.Log;

public class DisplayEmailActivity extends ActionBarActivity {

    private static final String CLASS_NAME = "ComposeEmailActivity";

    /**
     * Create layout once activity is implemented.
     * @param savedInstanceState
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        Log.d(CLASS_NAME, " :\tonCreate");

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_display_email);
        Intent displayEmailIntent = getIntent();

        displayEmailIntent(displayEmailIntent);
    }

    /**
     *  Display EMail using values as received from ComposeEmailActivity for the email fields : <br/>
     *  <ul><li>from</li>
     *      <li>to</li>
     *      <li>cc</li>
     *      <li>subject</li>
     *      <li>message</li></ul>
     */
    protected void displayEmailIntent(Intent intent) {
        Log.d(CLASS_NAME, " :\tdisplayEmailIntent");

        String from = "";
        String to = "";
        String cc = "";
        String subject = "";
        String message = "";

        if (intent != null) {
            from = intent.getStringExtra(ComposeEmailActivity.EMAIL_FROM);
            to = intent.getStringExtra(ComposeEmailActivity.EMAIL_TO);
            cc = intent.getStringExtra(ComposeEmailActivity.EMAIL_CC);
            subject = intent.getStringExtra(ComposeEmailActivity.EMAIL_SUBJECT);
            message = intent.getStringExtra(ComposeEmailActivity.EMAIL_MESSAGE);
        }

        TextView textViewFrom = (TextView) findViewById(R.id.textViewFromValue);
        textViewFrom.setText(from);
        TextView textViewTo = (TextView) findViewById(R.id.textViewToValue);
        textViewTo.setText(to);
        TextView textViewCc = (TextView) findViewById(R.id.textViewCcValue);
        textViewCc.setText(cc);
        TextView textViewSubject =  (TextView) findViewById(R.id.textViewSubjectValue);
        textViewSubject.setText(subject);
        TextView textViewMessage =  (TextView) findViewById(R.id.textViewMessageValue);
        textViewMessage.setText(message);

    }

    /**
     * Returns to calling activity : <b>ComposeEmailActivity</b>.
     * @param view
     */
    protected void backToEMailWriter (View view) {
        Log.d(CLASS_NAME, " :\tbackToEMailWriter");
        finish();
    }

// REMOVED – unused methods.
}
```

APPENDIX VI – CODE : DISPLAYEMAILACTIVITY.JAVA

```
                                                          package org.dwan.paula.a01_emailemulator;

import android.content.Intent;
import android.os.Bundle;
import android.content.Context;
import android.content.SharedPreferences;
import android.content.SharedPreferences.Editor;
import android.support.v7.app.ActionBarActivity;
import android.view.Menu;
import android.view.View;
import android.widget.TextView;
import android.util.Log;

public class ComposeEmailActivity extends ActionBarActivity {

    TextView emailFrom;
    TextView emailTo;
    TextView emailCC;
    TextView emailBCC;
    TextView emailSubject;
    TextView emailMessage;

    private static final String CLASS_NAME = "ComposeEmailActivity";

    public static final String MY_PREFERENCES = "MyPrefs";
    public static final String EMAIL_FROM = "fromKey";
    public static final String EMAIL_TO= "toKey";
    public static final String EMAIL_CC = "ccKey";
    public static final String EMAIL_BCC = "bccKey";
    public static final String EMAIL_SUBJECT = "subjectKey";
    public static final String EMAIL_MESSAGE = "messageKey";

    sharedPreferences sharedPreferences;

    /**
     * Create layout once activity is implemented.
     * @param savedInstanceState
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        Log.d(CLASS_NAME, " :\tonCreate");

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_compose_email);

        populateEditFields();
        populateSharedPreferenceUsingEditFields();
    }

    /**
     * Updates each TextEdit field for the EMail Emulator.
     */
    private void populateEditFields() {
        Log.d(CLASS_NAME, " :\tpopulateEditFields");

        emailFrom = (TextView) findViewById(R.id.editTextemailFrom);
        emailTo = (TextView) findViewById(R.id.editTextemailTo);
        emailCC = (TextView) findViewById(R.id.editTextemailCC);
        emailBCC = (TextView) findViewById(R.id.editTextemailBCC);
        emailSubject = (TextView) findViewById(R.id.editTextemailSubject);
        emailMessage = (TextView) findViewById(R.id.editTextemailMessage);
    }

    /**
     * Updates each field of the EMail Emulator using sharedPreferences and thus stored values,
     * if applicable.
     */
    private void populateSharedPreferenceUsingEditFields() {
        Log.d(CLASS_NAME, " :\tpopulateSharedPreferenceUsingEditFields");

        sharedPreferences = getSharedPreferences(MY_PREFERENCES, Context.MODE_private);

        if (sharedPreferences.contains(EMAIL_FROM)) {
            emailFrom.setText(sharedPreferences.getString(EMAIL_FROM, ""));
        }
        if (sharedPreferences.contains(EMAIL_TO)) {
            emailTo.setText(sharedPreferences.getString(EMAIL_TO, ""));
        }
        if (sharedPreferences.contains(EMAIL_CC)) {
            emailCC.setText(sharedPreferences.getString(EMAIL_CC, ""));
        }
        if (sharedPreferences.contains(EMAIL_BCC)) {
            emailBCC.setText(sharedPreferences.getString(EMAIL_BCC, ""));
        }
        if (sharedPreferences.contains(EMAIL_SUBJECT)) {
            emailSubject.setText(sharedPreferences.getString(EMAIL_SUBJECT, ""));
        }
        if (sharedPreferences.contains(EMAIL_MESSAGE)) {
            emailMessage.setText(sharedPreferences.getString(EMAIL_MESSAGE, ""));
        }
    }

    /**
     * Saves values of each edit field for the Email Emulator so all are visible in the
     * second activity <b>DisplayEmailActivity</b> when called via Intent. <br/>
     * Also save values of each edit field for the Email Emulator using <b><i>SharedPreferences</i></b>
     * & <b><i>Editor</i></b> so all values are retained when the application is closed/re-opened. <br/>
     * Method is instigated when <b>[Send]</b> button is clicked.
     * @param view
     */
    public void sendWrittenEmail (View view) {
        Log.d(CLASS_NAME, " :\tsendWrittenEmail");

        String from = emailFrom.getText().toString();
        String to = emailTo.getText().toString();
        String cc = emailCC.getText().toString();
        String bcc = emailBCC.getText().toString();
        String subject = emailSubject.getText().toString();
        String message = emailMessage.getText().toString();

        Editor editor = sharedPreferences.edit();
        editor.putString(EMAIL_FROM, from);
        editor.putString(EMAIL_TO, to);
        editor.putString(EMAIL_CC, cc);
        editor.putString(EMAIL_BCC, bcc);
        editor.putString(EMAIL_SUBJECT, subject);
        editor.putString(EMAIL_MESSAGE, message);
        editor.apply();

        Intent composeEmailIntent = new Intent(this, org.dwan.paula.a01_emailemulator.DisplayEmailActivity.class);
        composeEmailIntent.putExtra(EMAIL_FROM, from);
        composeEmailIntent.putExtra(EMAIL_TO, to);
        composeEmailIntent.putExtra(EMAIL_CC, cc);
        composeEmailIntent.putExtra(EMAIL_SUBJECT, subject);
        composeEmailIntent.putExtra(EMAIL_MESSAGE, message);
        startActivity(composeEmailIntent);
    }

    /**
     * Clears and thus thus resets the values of each edit field for the Email Emulator. <br/>
     * Method instigated when <b>[Clear]</b> button is clicked.
     * @param view
     */
    public void clearWrittenEmail (View view) {
        Log.d(CLASS_NAME, " :\tclearWrittenEmail");

        final String BLANK_EMAIL_TEXT = "";

        emailFrom.setText(BLANK_EMAIL_TEXT);
        emailTo.setText(BLANK_EMAIL_TEXT);
        emailCC.setText(BLANK_EMAIL_TEXT);
        emailBCC.setText(BLANK_EMAIL_TEXT);
        emailSubject.setText(BLANK_EMAIL_TEXT);
        emailMessage.setText(BLANK_EMAIL_TEXT);

        Editor editor = sharedPreferences.edit();
        editor.putString(EMAIL_FROM, BLANK_EMAIL_TEXT);
        editor.putString(EMAIL_TO, BLANK_EMAIL_TEXT);
        editor.putString(EMAIL_CC, BLANK_EMAIL_TEXT);
        editor.putString(EMAIL_BCC, BLANK_EMAIL_TEXT);
        editor.putString(EMAIL_SUBJECT, BLANK_EMAIL_TEXT);
        editor.putString(EMAIL_MESSAGE, BLANK_EMAIL_TEXT);
        editor.apply();
    }

// REMOVED – unused methods.

}
```

References

1. www.publicdomainpictures.net : picture of moon → logo (not used in first draft)
2. http://developer.android.com/training/basics/activity-lifecycle/index.html : overview of activity life-cycle http://developer.android.com/training/basics/firstapp/starting-activity.html : adding second activity
3. http://www.tutorialspoint.com/android/android_shared_preferences.htm : SharedPreferences tutorial