1 Overview

This week, you are going to implement an Android application.

In these instructions, we refer to the four members of your team as Teammates A, B, C, and D. Although it is possible to complete these four tasks in sequence independently, it is best to sit together while you complete the lab, so that you are all familiar with the four steps and so that you can help each other.

2 Teammate A: Create an Android project

In your team repo create a new directory called lab8. Working on cdf, change to your new lab8 directory. Get a copy of the starter code for this lab as follows:

```
cp /u/dianeh/pub/Lab8.zip .
unzip Lab8.zip
```

Don’t leave out the dot at the end of the first line!

In Android Studio, create a New Project with:

- Application name Lab8,
- Company domain b07.cs, and
- Project location group.XXX/lab8 directory, which is the directory you have checked out/updated from your team repository.

Next:

1. Select “Phone and Tablet” and set “Minimum SDK” to API 15 — these should be the default settings.
2. Select “Blank Activity”. 
3. Leave the default values for Activity Name (MainActivity), Layout Name (activity_main), Title (MainActivity), and Menu Resource Name (menu_main).

4. Click Finish. Wait...

5. Commit the Android project (everything except for the contents of the bin directory, the gen directory, and hidden files).

3 Teammate B: Changing the Layout and the displayed text, and adding a new Activity.

1. In directory lab8 in your local copy of your team’s repository, run svn update.

2. Use the Text view (i.e., the XML view) to edit your activities. While it may be tempting to drag-and-drop in the Design view, you need to be able to understand the XML, and this is a good opportunity to practice.

3. In res -> layout -> content_main.xml, change the layout from RelativeLayout to LinearLayout. Add android:orientation="vertical" to the specifications of the Layout.

4. Change the displayed text from “Hello, World!” to “Welcome to our first Android App!”. But instead of hard-coding the text in the text field, we will do the right thing and create a new string in res -> values -> strings.xml with the ID greeting and value Welcome to our first Android App!:

   <string name="greeting">Welcome to our first Android App!</string>

   The specification of the TextView becomes:

   <TextView android:text="@string/greeting"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
   />

   Commit the Android project (everything except for the contents of the bin directory, the gen directory, and hidden files).

5. You may remove the automatically generated FloatingActionButton or leave it there.

4 Teammate C: Adding an EditText and a Button

1. In directory lab8 in your local copy of your team’s repository, run svn update.

2. Create two more strings in the strings.xml file:

   • enter_text with value Enter text here, and
   • go with value Go!

3. Add a new EditText (plain text field), with the following specifications:

   • id is @+id/entered_data
   • layout_width is fill_parent
   • layout_height is wrap_content
• hint is @string/enter_text

The id gives an ID to the field, so that you can “find” it in your Java code later.

4. And a new Button, with the following specifications:

• id is @+id/go_button
• layout_width is wrap_content
• layout_height is wrap_content
• text is @string/go
• onClick is enterData

The method name specified in the onClick field will be run when the user clicks the button. We will implement this Java method later.

5. Notice that your project has another activity named DisplayActivity. It was created using:

New -> Activity -> Empty Activity

(Note: the code won’t compile at this point, since method enterData doesn’t exist yet.)

Commit the Android project (everything except for the contents of the bin directory, the gen directory, and hidden files).

5 Teammate D: Add the method for the button action

• In directory lab8 in your local copy of your team’s repository, run svn update.

• In MainActivity.java, add this method:

public void enterData(View view) {
    // Specifies the next Activity to move to: DisplayActivity.
    Intent intent = new Intent(this, DisplayActivity.class);

    // Gets the data from the EditText field.
    EditText dataField = (EditText) findViewById(R.id.entered_data);
    String data = dataField.getText().toString();

    // Passes the String data to DisplayActivity.
    intent.putExtra("dataKey", data);
    startActivity(intent); // Starts DisplayActivity.
}

• In activity_display.xml, add a TextView with the following specifications:

– id is @+id/display_data
– layout_width is fill_parent
– layout_height is wrap_content

• In DisplayActivity.java, add this code to the end of method onCreate:
// get the Intent that launched me
Intent intent = getIntent();

// get the String that was put into the Intent with key dataKey
String data = (String) intent.getSerializableExtra("dataKey");

// find the TextView object for TextView with id display_id
TextView newlySubmitted = (TextView) findViewById(R.id.display_data);

// specify text to be displayed in the TextView
newlySubmitted.setText(data);

• Run the application (using the emulator). It will take some time — be patient! Test the app to make sure it runs without errors and works as expected.

Commit the Android project — everything except for:
.idea/
.gradle
/*local.properties
/*/out
/*/build
/*/production
	.*.iml
	.*.iws
	.*.ipr
	*
	.*.swp

6 Teammates A, B, C: Run the app

Teammate D has run the app, now it’s time for the rest of the team to do so. This step will confirm that you can successfully share Android code among team members.

Run `svn update` and launch the application. You will need to create a name for your virtual device. It is important that you do not leave the default name. Instead name it as `teamname_A`, `teamname_B`, `teamname_C`. 