CST2335 – Graphical Interface Programming

Lab 2

Introduction:

The goal of this lab is become familiar with how XML and Java are combined for creating user interfaces. The layouts and widgets are created in XML, and Java then gets references to the objects created in XML to control the behavior of the application

References:

<https://developer.android.com/guide/topics/ui/controls/button>

<https://www.tutorialspoint.com/android/android_user_interface_controls.htm>

<https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/ui/controls.html>

Steps:

1. Create a branch of your software from Lab 1. Select the VCS menu in Android Studio and click “Git” -> “branches”. From the dialog box, select “New Branch” and call it “Lab 2”. Using a new branch means that all of your software commits will only change this branch, and not the original branch. This means you can go back to the “master” branch at any time.
2. Go to http://flaglane.com/ and look for an image of a Canadian flag. You should download the small PNG file, 200 pixels wide. Once downloaded, rename the file **flag.png**. Copy the file into the drawable folder in AndroidStudio. Next right-click on the “res” folder and select “New” -> “Android Resource Directory”. Set the resource type to be drawable, and set the directory name to be “drawable-XX”, but replace the XX with the two letter language code that you used for your strings file, for example “drawable-es” for Spanish. Next, download a flag of that country from <http://flaglane.com/>, a PNG file 200 pixels wide. Rename that file **flag.png** and copy it into the new drawable-XX folder you created. You will use these flags for the ImageButton later on.
3. The rest of this lab is to create three different layouts that all look like the image on the next page.
   1. One should use only LinearLayout. Call it activity\_main\_linear.xml.
   2. One should use a GridLayout. Call this activity\_main\_grid.xml.
   3. One should use a RelativeLayout. Call this activity\_main\_relative.xml

A screenshot of a cell phone

Description automatically generated

Figure The layout for application. Recreate this LinearLayout, GridLayout, and RelativeLayout

Row 1: The top item is a <TextView> that has the string “Look at my GUI:”. Use the parameter android:textSize=”30dp” to make the text bigger than normal.

Row 2: Below the text view is a row that has a <Button> with text=”Click here”, and a <CheckBox> with the text=”Check this out!”. They should each take approximately 50% of the width of the row. Remember that @drawable will be either the drawables folder, or drawables-XX folder that you created in step 2. If you change the phone’s language, the flag should switch to the one you downloaded for that country.

Row 3: On the left side is an <ImageButton> whose src=”@drawable/flag”. On the right side is a <Switch> object with the text=”Switch on / off”.

Row 4: There is only an <EditText> with the hint=”Type here”. Set the inputType=”textPersonName”. This means when you type, every first letter will be capitalized. The EditText should take up the entire row.

1. Add a click listener to the “Click Here” button, which shows a Toast message “Here is more information”, and for a Toast.LENGTH\_LONG amount of time. The toast message should be translated into other languages so you should use getResources.getString(R.string.toast\_message) to get the <string name=”toast\_message” from the right values-XX language folder.
2. Add a checked change listener to the Switch to show a Snackbar which says “The switch is now”, followed by “off” if the Boolean parameter is false, or “on” if the Boolean is true. Add an “Undo” action button to the Snackbar which sets the button back to its original state. You should be able to do this using the two parameters of onCheckChanged(CompoundButton cb, boolean b){ …

//In the Snackbar’s setAction( ):

Snackbar.setAction( “Undo”, click -> cb.setChecked(!b);

}

1. For the LinearLayout, the root tag should be a vertical LinearLayout, and each row should be a horizontal LinearLayout. For the items in each row, set layout\_weight=”1” and gravity=”fill” if the items should take 50% of the row. If there is only 1 item in a row, then set layout\_width=”match\_parent”.
2. For the GridLayout, use a grid with 4 rows and 2 columns. Set layout\_weight=”1” and layout\_gravity=”fill” to make the objects use the entire grid box. Also, use layout\_columnSpan=”2” to make an object use 2 grid boxes.
3. For the RelativeLayout, use a combination of layout\_alignParentTop, layout\_alignParentStart, layout\_alignParentEnd to align widgets to the sides of the screen. Use layout\_below, layout\_toEndOf, layout\_toStartOf, to position objects properly.
4. Translate all of the strings used in the layouts to your second language. Anywhere you set text=”something” or hint=”something” should use the “@string/” format.
5. When you are finished, change the setContentView() to load your activity\_main\_linear.xml layout. **Commit this with the message “step 1”.**
6. Create a branch called “lab2\_step2”. Change your MainActivity.java file to load activity\_main\_grid.xml layout file. **Commit this with the message “step 2”.**
7. Create a branch called “lab2\_step3”. Change your MainActivity.java to load activity\_main\_relative.xml layout. **Commit this with the message “step 3”.** Show your lab professor each of the three layouts by checking out the three different branches. Also, change the language to show that the flag image changes for your second language, and your strings change to the second language.
8. Right-click on the “App” folder and select “Git” -> “Repository” -> “Push”. Now your changes are pushed to Github.

Marks:

The LinearLayout looks correct. +3

The GridLayout looks correct. +3

The RelativeLayout looks correct. +3

ImageView changes to show the different flag +1

strings.xml file has been updated with text for second language (step 7) +1

Clicking on the button shows a Toast +1

Clicking on the Switch shows a Snackbar, with an Undo action +1

The layout files are separated on different Git branches +1