Lab 1 – Simple UI and Persisting Activity State

CIS399, Android Application Development

Purpose: This lab is designed to give you practice:

- Using Android Studio
- Using an Android emulator
- Creating UI widgets using the designer and declaratively using AXML
- Writing Java code to Handle UI events
- Writing Java code to save and restore the state of an Activity

Part 1: Do the textbook exercises shown below:

- 2-1, Create a Hello World app and modify its UI layout according to the instructions. You will just be changing what's there, not adding any new widgets.
- 2-2, Create the Invoice Total app UI. You will:
  - Add some TextView widgets
  - Add a EditText widget
- Arrange the widgets and set some attributeNote: Use the *Empty Activity* template and set the min API to 18 or higher.
- 3-1, Finish the Invoice Total App. You will:
  - Add an event handler for the EditText widget
  - Add code to save the state of the app when it has been paused or stopped
  - Add a launcher iconNote: that for exercise 3-1, there is a starter project in the source code provided by the publisher of the textbook. It is identical to the finished version of 2-2, so you can just use your completed exercise 2-2 instead of the starter project.

Upload a text file to Canvas in which you will report, for each exercise above, whether you:

A. Followed all the steps shown in the book and successfully compiled and ran the app.
B. Opened the completed solution in Android Studio, experimented with the code, and ran the app.
C. Read through the steps and inspected the relevant code listings without writing or running the app.
D. Didn't do any of the above.

Part 2: Create an app that counts the number of times a button is clicked and that has a button that lets the user reset the count back to zero.

1. Create a new Android app using the *Empty Activity* template. Accept all the default settings.
2. Using the existing Constraint Layout, add two buttons and an additional TextView to the XML file. (You can use either the designer or directly edit the XML source)
   - Set one Button's text to "Add One", and give it an appropriate id
   - Set the other Button's text to "Reset", and give it an appropriate id
   - Give the existing TextView an appropriate id (it will be used to display the count)
   - Set the new TextView's text property to "Count" (it will be used as a label for the other TextView)
3. Write the event handlers for the application. One of the event handlers will contain code to increment a count and display it. The other will contain code to reset the count back to zero and display it.
   - Add code to get references for the buttons and "Count" TextView
   - Implement the event handlers and set them to the appropriate widgets
4. Optional, extra credit. Store and retrieve the count so it isn't lost when the device is rotated.

Zip the folder containing your project and upload it to Canvas.

Written by Brian Bird, University of Oregon, Summer 2015, Revised Summer 2018