Lab 3: Android RPN Calculator

Due: Feb 21st at 11:59pm

Overview

In this lab, you will program an Android version of your Java RPN calculator from Lab 1. You are required to adapt and fix earlier code, integrate it into the event driven GUI framework, and also design the layout and user interface for the RPN calculator. **There will be no provided skeleton code for this lab: You will start from basic Android Application Project in Eclipse.** You are required to use your code, albeit improved upon, from the previous RPN lab in this lab.

Deliverables

Your submissions will include the entire Android proejct, but the files below are most important:

- `AndroidRPNCalculator.java`
- `JavaRPNCalculator.java`
- `Operator.java`
- `main_activity.xml`
- `README`

Your `README` file must include, (1) your name, (2) the name of all files in your submission, and (3) a short description of the program and your development process. Failure to include a `README` or not including sufficient information in the `README` will adversely affect your grade on this lab.

Project Retrieval and Submission Instructions

To submit this lab, once you've created the Android project in Eclipse, you will share the project over SVN to the following URI

```
https://cs71.cs.swarthmore.edu/svn/cs71/submission/<username>/03
```

Where `<username>` is replaced with your swarthmore username. Click finish, and then follow the on-screen instructions to com mitt. You may update and commit your project until the deadline. All commits after the deadline will be rolled back to the submission just prior to the deadline.

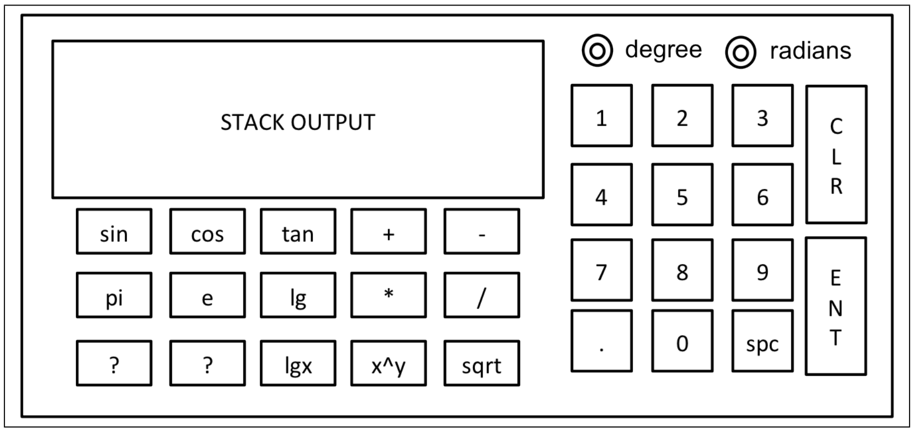

Figure 1: Sample Layout Diagram for the Android RPN Calculator

Android RPN Calculator

In this lab, you will adapt your code for a Java RPN calculator into a functional Android application. Unlike in previous labs, you will be starting from a blank slate. **You will start from base Android project in Eclipse, *i.e.*, a Hello World program**. You must adapt and change the Android project to meet the requirements of this lab.

Calculator Layout

You must design and develop the layout files for your calculator. A suggested layout is provided in Figure 1. At minimum, your layout file should include:

- Number pad with all digits, and a decimal button
- An Enter and a Clear button
- Radio buttons to switch between radians and degrees for trigonometric functions
- Buttons for all math operations and constants, including space for two additional operations not specified in the original lab document.
- A view for the output of the stack and provide feedback to the user as they type

You may add additional buttons or features as you see fit, e.g., a backspace button, but document this in your `README` file. Much of the actions and user-interface portions of this lab is also up to you. As long as your layout and user-interface includes the minimal features, you may adapt as you see fit, but such additions should not hinder the use of your calculator.

Functionality and Adapting Legacy Code

Your calculator is an RPN calculator as defined in Lab 1, and you must adapt that code to this new settings. That does not mean you must use all your old code in this assignment; on the contrary, you should improve code and/or remove redundant code as you see fit. However, **you are required to use the same API of the calculator from Lab 1**. That means, you must adapt the `Operator` class and `javaRPNCalculator` class to this setting, that is, you should be using the core logic you programmed previously, encapsulated within the `javaRPNCalculator`, but adapted to the GUI Android interface. For example, somewhere in your code, you should still be calling the method `Operator.operate` and `javaRPNCalculator.operate` and allow the `javaRPNCalculator` to maintain the stack and other calculator related information. **How you proceed in adapting legacy code to Android is part of your lab grade.**

Calculator Orientation

This is the first application in the class that will be oriented horizontally rather than vertically. This will require you to set a few values in certain places in the Android project, and also to reorient the Android Virtual Device.

The simplest way to force horizontal orientation of your Android application is in the `AndroidManifext.xml` file, which described a number of key properties of your application. In the `activity` portion of the XML specification, you can add the following orientation setting:

```
<activity
     ...
     android:screenOrientation="landscape"
     >
```

And to change the orienation of the AVD, you use the keyboard shortcut: Ctrl-F11 or Ctrl-F12. For more information about controlling the emulator, you can read the guide online:

```
http://developer.android.com/tools/help/emulator.html#controlling
```

Challenge Exercises and Extra Credit

- (**Challenge**) Style the background of the application so the calcuator apears with white button on a black background. The stack output should be similarly stylized.
- (**.5 Points**) Generate a second layout for your calculator for horizontal view that automatically is used depending on the orientation of the calculator. Place this second layout file as `res/layout/vertical_layout.xml` and include code/xml portions for automatically using the right layout.

Competition and Extra Credit

There will be a competition component to this lab, based on the competition rules. There is limited extra credit on this assignment, so the winner of the competition will still receive 1 point of extra credit; however, **the second and third runner-up will receive .25 points of extra credit** if there is enough entrants. As usual, entry into the competition has its pitfalls; bad entrants may either receive public, anonymous shaming and/or point reductions.