PocketCalculator

Lab Objectives

- Import an existing Android Studio project
- Create your own calculator layout
- Implement button listeners
- Given a model, implement a view and controller
- Complete a simple calculator application

What to Turn in

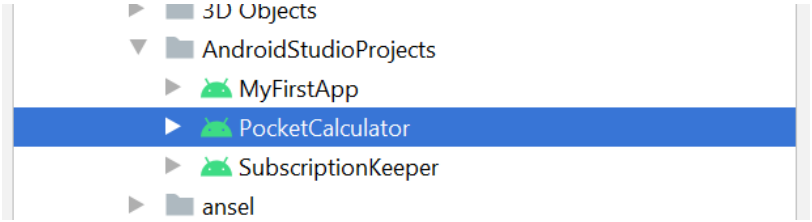
Demonstrate your final app to a TA.

Lab Description

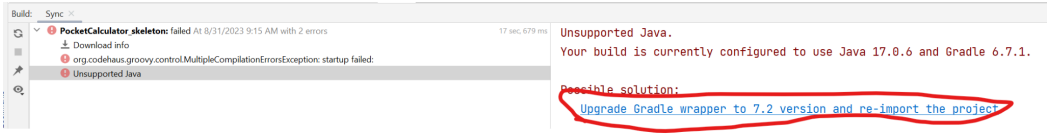
For this lab, you are given the skeleton of a simple pocket calculator app. The skeleton includes a model that implements the business logic of a calculator such as that shown in the above figure. Your task is to implement the view and the controller that will make use of this model. Refer to the lecture slides if you need a refresher on the MVC design pattern.

Steps

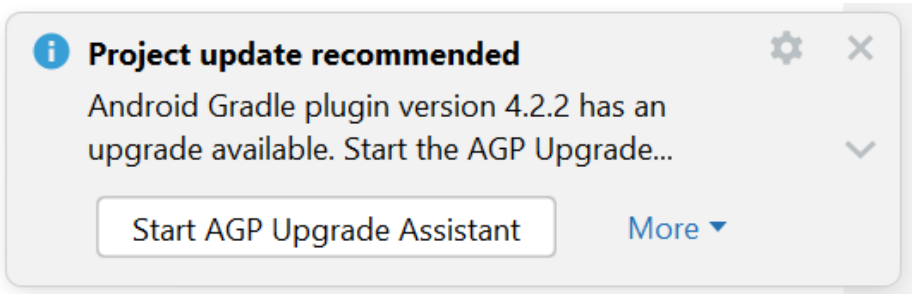
1. Download PocketCalculator_skeleton.zip from the Canvas assignment and unzip it to your AndroidStudioProjects folder (or wherever you keep your Android app code).
2. In Android Studio, open the project. While browsing, you should find that the project folder shows up with an Android icon.



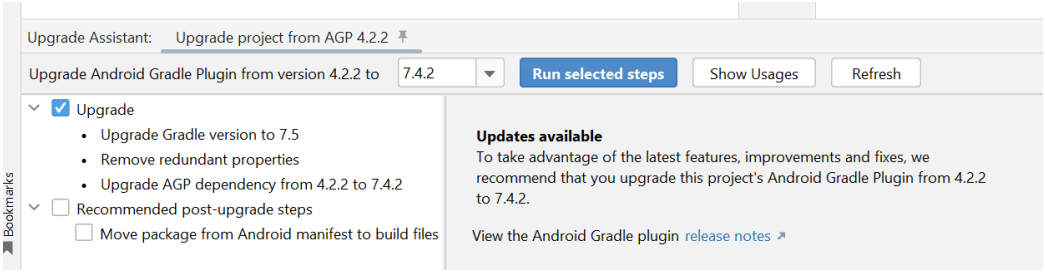
3. If you get prompted to enter the location of the SDK, use one of the following:
    a. On a lab machine, use C:\Android\sdk
    b. On a personal machine, (probably) use C:\Users\<user>\AppData\Local\Android\Sdk
4. After you open the project, Android Studio may ask to update the Gradle plugin for it. If so, perform the update. Gradle is the build tool (analogous to Make) used for Android projects.
    a. If you are not automatically prompted to update Gradle and the Gradle sync fails, check the errors in the Gradle Sync output and look for a link to click.



In our environment, after this task we were prompted to update the Gradle plugin:



Clicking "Start AGP Upgrade Assistant" then brings up this:



After this, we were prompted with ANOTHER Gradle plugin update. After completing this, the project was ready to go.

5. Implement your own calculator layout in activity_main.xml. A sample layout is provided below. Use the provided strings.xml for text.
6. Examine the CalculationStream.java code. This is the model. Take note of the model's API (public methods).
7. Implement the controller by adding event listeners for each button to MainActivity.java. These event listeners should interact with the model using the model's API. The event listener for the equals button has been done for you.
8. You've been given a method that updates the calculator's display. This method should be called for every button press.
9. Wire your view to your controller by adding each event listener as the onClick attribute for the corresponding button in activity_main.xml.

Evaluation

The finished calculator should act similar to a normal, simple pocket calculator. It should:

1. Support addition, subtraction, multiplication, and division. **(5 pts.)**
2. Be able to perform multiple computations in a row, evaluating input left-to-right and ignoring order of operations (e.g. for our purposes, 2 + 2 * 2 = 8). **(5 pts.)**

Note that the provided files are to help you get started, and you shouldn't have to change the model. But, feel free to change anything in any file, as long as you arrive at a working pocket calculator that uses the MVC design pattern.

Possible Layout

In the layout shown on the left below, each row is a horizontal LinearLayout (highlighted in pink on the right), nested inside one vertical LinearLayout (highlighted in green on the right).

This is merely a suggestion; you may use any layout you wish. If you're feeling bold, try a ConstraintLayout.

Hint: if you're adding horizontal LinearLayouts inside a vertical LinearLayout and can only see the first one, make sure you set the height to wrap the content instead of match the parent.