

# Laboratory – The Activity Class

---

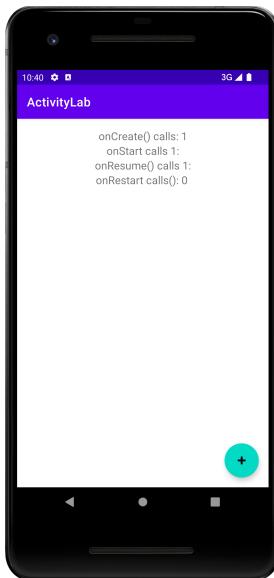
[Learn about the Activity Lifecycle](#)

## Objectives

Familiarize yourself with the Activity class, the Activity lifecycle, and the Android reconfiguration process. Create and monitor a simple application that observes an Activity as it moves through its lifecycle. Once you've completed this Lab you should understand: the Activity class, the Activity lifecycle, how to start Activities programmatically, and how to handle Activity reconfiguration.

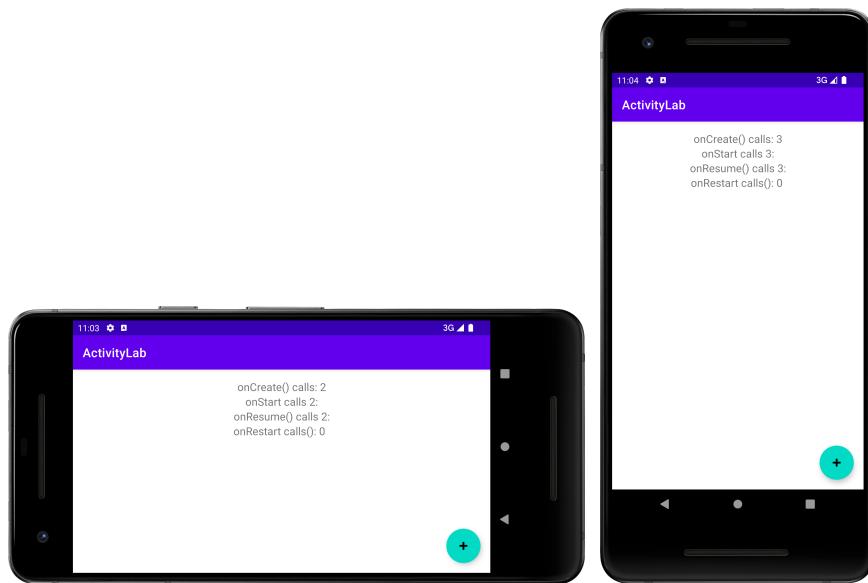
## The Activity Class

This lab is an app called ActivityLab. When it runs, the app displays a user interface like that shown below.

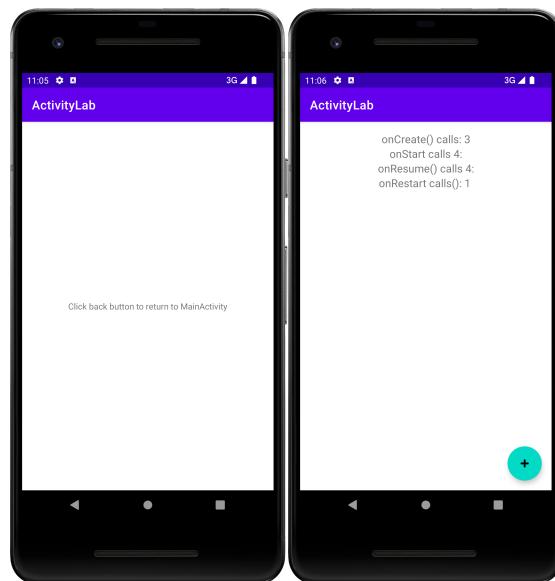


This application comprises two Activities. The first Activity is called “MainActivity,” and the second Activity is called “SecondActivity.” The display shows the number of times certain Activity lifecycle callback methods have been invoked on a MainActivity instance since the app started. These methods are onCreate(), onStart(), onResume() and onRestart().

When the user rotates the screen, MainActivity will move through its Activity lifecycle, calling various lifecycle methods mentioned above.



In addition, when MainActivity is active and the user clicks on the floating action button (fab), the button with the “+” symbol, MainActivity responds by activating SecondActivity. The user can then hit the back button to navigate back to MainActivity. During these navigation actions, various lifecycle methods will be called on MainActivity and all associated counters should be incremented.



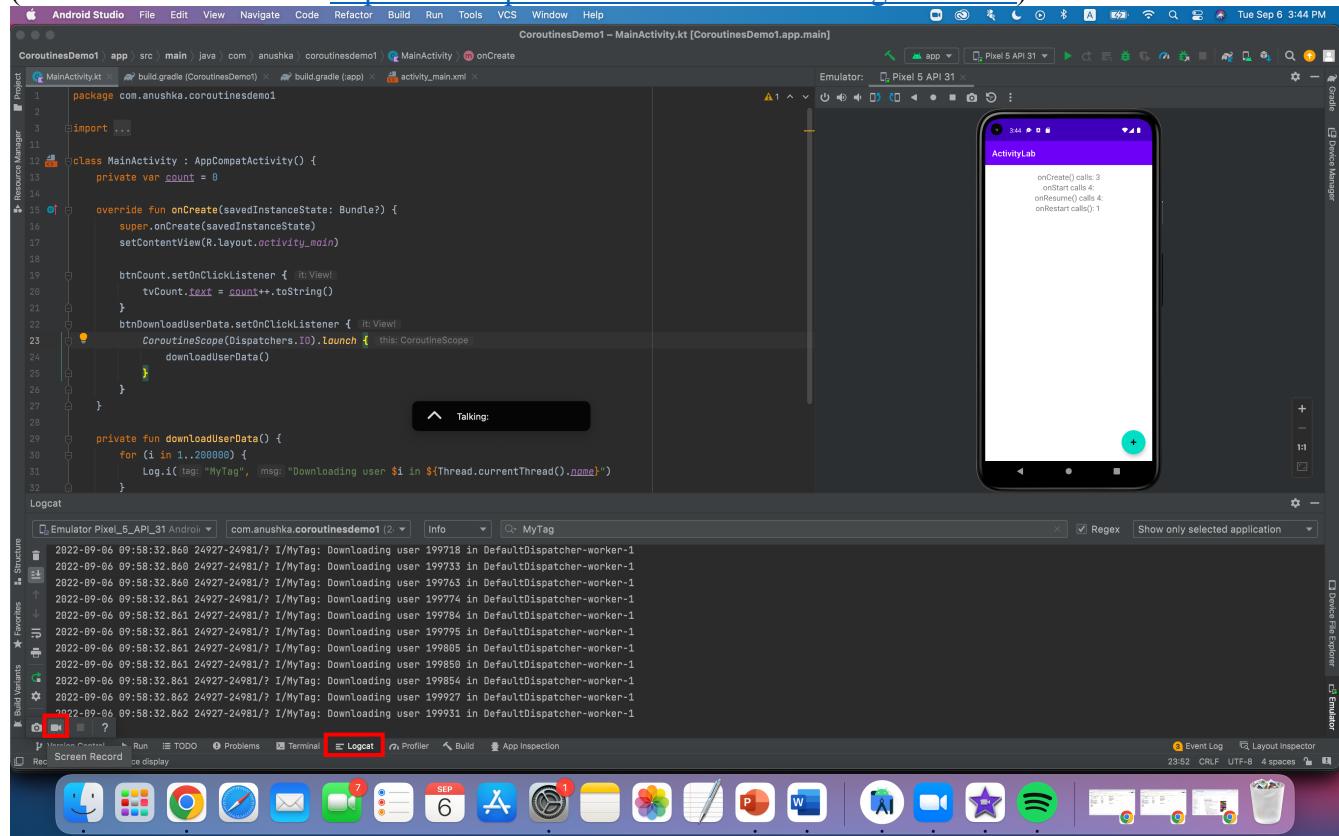
See the screencast, LabActivity.mp4, that's included in the Lab directory.

# Testing

After completing your solution, you will record a screencast while performing a manual test. Afterward, you will submit your code and the screencast via git.

## \* How to record a screencast using Logcat

(More info available at - <https://developer.android.com/studio/debug/am-video>)



There is one test case with several evaluation points. Each evaluation occurs at This test case operates as follows:

1. Start the ActivityLab app in portrait mode
2. Check the lifecycle method invocation counts (evaluation point 1)
3. Rotate the device to landscape mode
4. Check the lifecycle method invocation counts (evaluation point 2)
5. Rotate back to portrait mode
6. Check the lifecycle method invocation counts (evaluation point 3)
7. Click on the fab button to start SecondActivity
8. Click on device back button
9. Check the lifecycle method invocation counts (evaluation point 4)

# Submission

When you are ready just commit your solution to your repo on GitLab by running the following

commands:

```
% git add path/to/changed/files  
% git commit -m "completed Lab2_Activity"  
% git push origin main
```

Note: if you have not already pushed this branch to your repo on GitLab you will need to make a slight modification for this first time and run this instead:

```
git push -u origin main
```

This sets up tracking between your local branch and a branch with the same name on your repo in GitLab.

## Some Implementation Notes

We are providing template code and layout resources for this application. Only modify the areas marked with the word TODO.

We have done our testing on an emulator using a Pixel 5 AVD with API level 31. To limit configuration problems, you should test your app against a similar AVD.

You can reorient your device in the emulator by pressing Ctrl+F12 (Command+F12 on Mac) or by hitting one of the rotate icons in the emulator window. Make sure that Auto Rotate is turned on in your Display settings before you try reorienting your device.

\* How to turn on Auto Rotate in emulator settings (Pixel 5 with API lv 31)

