

Lifecycle-Aware Components Lab

Lifecycle-Aware Components

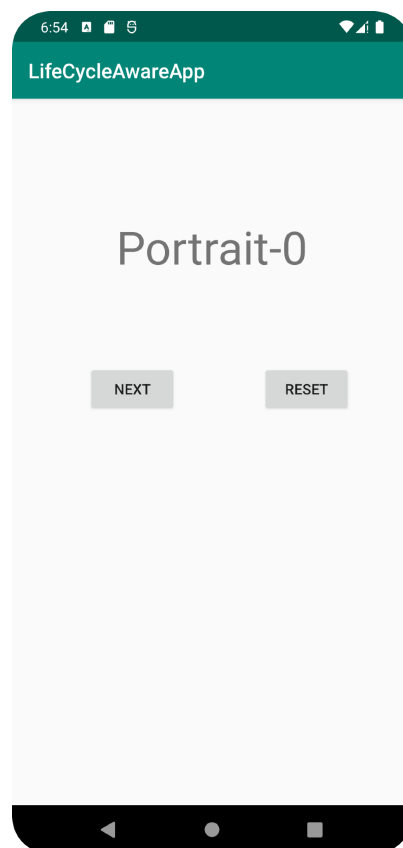
Objectives:

This week's Lab explores the use of Lifecycle aware components, view model and live data. This lab is a little different from the previous labs in the sense that we have given you a very minimal skeleton code and you are expected to implement your own logic to make the app work in the way described ahead. There will be minimal TODOs given to you for this app and these are to be treated **only as suggestions**. This will give you more freedom to design your own workflow and logic for the app. This will be helpful in preparing you for final projects.

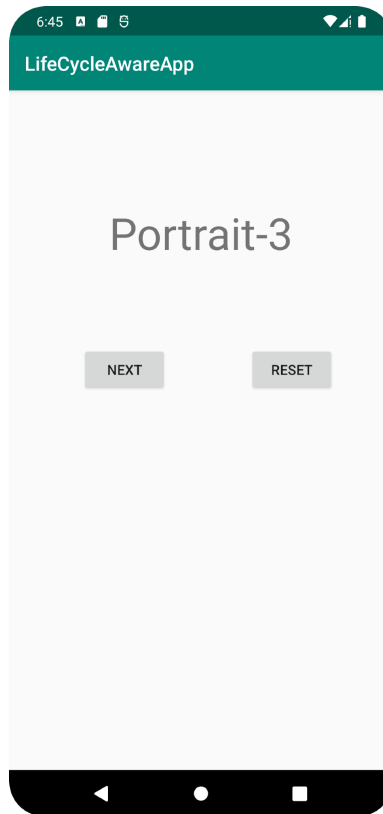
App Behavior:

This Lab involves an app which will show your current orientation of the phone and a counter which can be incremented by pressing the next button. Whenever you change the orientation of the phone, the current instance of your activity is destroyed, and a new instance is started.

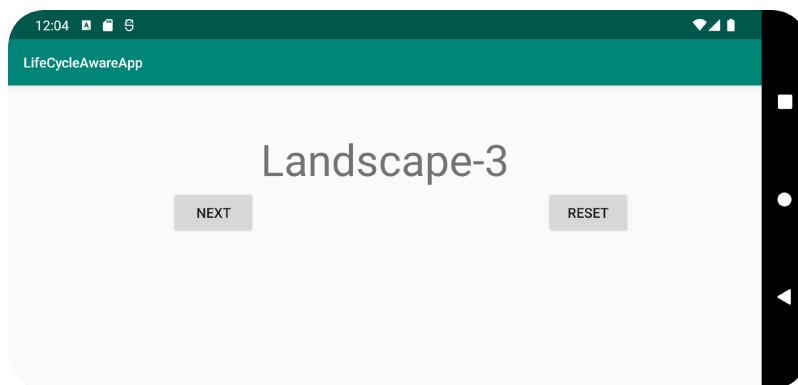
When you open the app following screen will show up with the screen orientation and counter value.



If you press the “Next” button 3 times following will be the result:



Now, if you rotate the device counterclockwise, the following screen will be show:



Notice how the counter’s value is preserved when the screen is rotated. This is possible due to the CounterViewModel which preserves data while the activity is destroyed between screen rotations.

Implementation Notes:

1. Download the application skeleton files and import them into your IDE.
2. You can modify, add, and delete any of the classes or function with some restrictions:
 - i) The entry point of app should be *LifecycleMainActivity*.
 - ii) You must use the included layout files only.
3. The displayed text should not have any space between the screen orientation, “-” and the counter value. For example: “REVERSE LANDSCAPE-0”.
4. We have implemented a supporting function called “getScreenOrientation” in *LifecycleMainActivity*.

Testing:

There are no automated tests in this lab. We will test your application manually for correctness.

Submission

To submit your work, you will need to commit your solution to your repo on GitLab by running the following command: `git push origin master`.

Additional Notes and Tips:

- 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 master`. This sets up tracking between your local branch and a branch with the same name on your repo in GitLab.
- Thinking for yourself : Sometimes you might encounter problems which you might have already solved while doing previous labs.