Data Management Lab

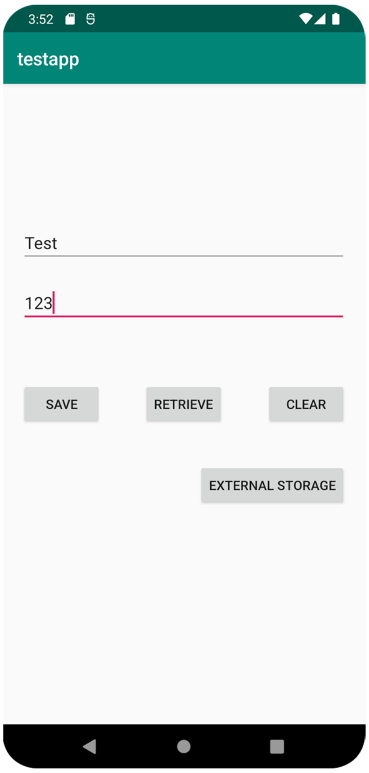
Data Management Lab

This week's lab explores the use of data management techniques such as Shared Preferences and saving to external storage.

App Behavior:

This app has two parts to it :

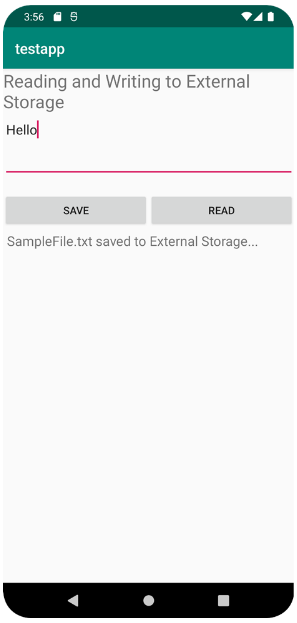
1. Shared Preferences - SharedPreferences are basically persistent maps that holds key-value pairs of simple data types. They are automatically persisted across application sessions. Here, you have two simple text fields containing "Name" and "UID". Once you fill those up and hit "Save", this should trigger the SharedPreference saving. On hitting "Clear", the fields in your EditText must clear and be available for further input. On hitting "Retrieve", you must return the **latest** name and UID stored in the shared preferences.



2. External Storage : On filling the text in the EditText, and hitting "Save", the data in the text field should save to the file as specified by **filepath** and **filename** variables. On hitting "Save", a function

to clear the EditText must also be called. A toast must be generated which tells us that the file has been saved.

```
private val filename = "SampleFile.txt"
private val filepath = "MyFileStorage"
```

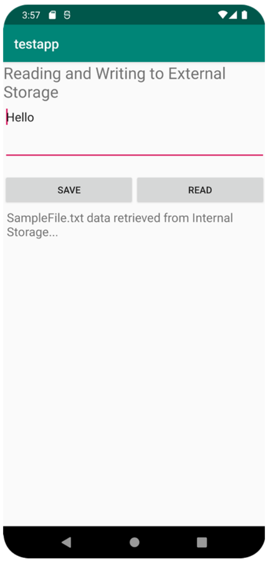


Reading and Writing to External Storage

Hello

SampleFile.txt saved to External Storage...

The “Read” button should perform the same function as the “Retrieve” button previously.



Reading and Writing to External Storage

Hello

SampleFile.txt data retrieved from Internal Storage...

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 *MainActivity*
    ii) You must use the included layout files only
3. We have implemented supporting functions like "isExternalStorageAvailable" in *External*
4. In *MainActivity,* the function **Save** must be called on clicking the "Save" button and must save the text fields to shared preferences. Similarly, **clear** and **Get** must be called on clicking the "Clear" and "Retrieve" buttons respectively.
5. In *External*, there are no functions defined for you - you are free to make a similar function structure as in the *MainActivity* to call functions to save and retrieve the data to and from the file.

Testing:

The test cases for this Lab are in the Lab5DataLab/app/src/androidTest/java/course/labs/lab5_datalab project. You can run the test cases either all at once, by right clicking the test package and then selecting Run 'Tests in 'course.lab…', or one at a time, by right clicking on an individual test case class and then continuing as before. The test classes are Robotium test cases. As you implement various steps of the Lab, run the test cases every so often to see if you are making progress toward completion of the Lab.

Warnings:

These test cases have been tested on a Pixel 5 AVD emulator with API level 31. To limit configuration problems, you should test your app against a similar AVD.

Once you've passed all the test cases, submit your project to GitLab.

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 main.

Thinking for yourself: Sometimes you might encounter problems which you might have already solved while doing previous labs. Learn to debug your errors and make sure to check the test files and figuring out why you might be failing the tests.