BCS 371 Lab – Compose LifeCycle

Overview

In this lab you will write an app that shows how the compose lifecycle works.

Create a project

Create a new Android application in Android Studio. Choose the **Empty Activity** type to create an empty activity that uses Jetpack Compose.

Setup the Main Screen

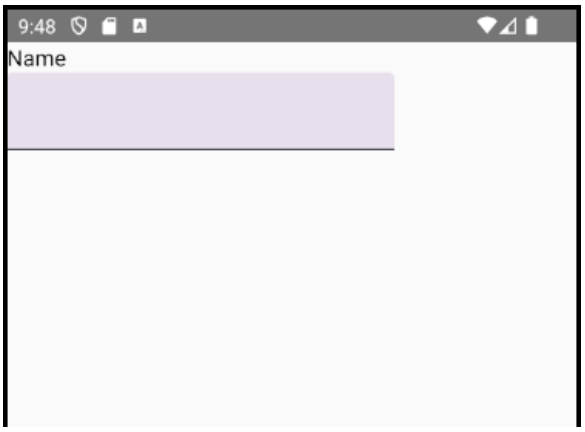
Create a Kotlin file named MainScreen.kt. Write the following composable functions:

- CreateText(d: String) – Should create a Text composable.
- CreateTextField() – Should create a TextField composable. Make sure to create a local variable to hold the TextField's data (the variable should be declared with remember and mutableStateOf).
- MainScreen(modifier: Modifier) – Should have a Column inside of it. The Column call should take the modifier as a parameter. Inside the body of the Column, it should call CreateText and CreateTextField to generate the UI (screenshot below).

Update setContent inside of MainActivity.onCreate so that it calls MainScreen. The call should go inside the Scaffold. Here is the function call:

MainScreen(modifier = Modifier.padding(innerPadding))

It should look like the following:



Add lifecycle event handlers

Add SideEffect composables to each function. Inside the SideEffect composable it should print a message to the logcat window.

- CreateText - Inside SideEffect it should print a message to the logcat window. Use the following statement to print to the logcat window:
  println("CreateText - SideEffect executed")
- CreateTextField - Inside SideEffect it should print a message to the logcat window. Use the following statement to print to the logcat window:
  println("CreateTextField - SideEffect executed")
- MainScreen - Inside SideEffect it should print a message to the logcat window. Use the following statement to print to the logcat window:
  println("MainScreen - SideEffect executed")

Note: Logcat Filter. Type in System.out as a filter in the logcat window to only see println messages

Note: Clearing Logcat. You can clear the Logcat window messages by right-clicking inside the Logcat window and choosing Clear Logcat from the context menu.

Run the App

You should see the main screen appear when you run the app. You can make the logcat window visible by clicking on the Logcat icon at the bottom left of Android Studio.

Now do the following:

- Check the logcat window and search for the "SideEffect executed" message. There should be one "SideEffect executed" message for each function. Type in System.out as a filter in the logcat window to only see println messages.
- Type characters into the name TextField. Typing in the TextField will change the value of the name variable which will cause the TextField's state to change. Since the TextField's state is changing, a recomposition will be triggered for that composable. This will cause the SideEffect to run in CreateTextField. You should see the "SideEffect executed" message for the TextField appear in the logcat for each keypress. You should not see "SideEffect executed" for the other functions.

Add address

Add a second Text and TextField to the UI. Do this by adding extra calls to CreateText and CreateTextField. It should look like the following:

- Run the app and check the logcat output.
- Initially when the app runs there should be two messages for CreateText, two messages for CreateTextField, and one message for MainScreen.
- Now add text to both the name and address TextFields. There should be additional messages for CreateTextField but not for MainScreen and CreateText.

Add a count to CreateTextField

Update CreateTextField so that it keeps a count of the number of times it is called.

- Create a local variable to hold the count. The variable should be declared with remember and mutableStateOf. Pass 0 into mutableStateOf.
- Inside SideEffect and just before the call to println you should increment the count variable.
- Append the count to the end of the println message.
- Run the app and check the logcat output.
- Initially when the app runs, the messages for CreateTextField should both show a count of 1. Each call to CreateTextField generates a separate count variable.
- Now add text to both the name and address TextFields. There should be additional messages for CreateTextField which show new counts depending on which TextField is being updated.

Add a LaunchedEffect

Add LaunchedEffect composables to each function. Inside the LaunchedEffect composable it should print a message to the logcat window.

- CreateText - Inside LaunchedEffect it should print a message to the logcat window. Use the following statement to print to the logcat window:
  println("CreateText - LaunchedEffect executed")
- CreateTextField - Inside LaunchedEffect it should print a message to the logcat window. Use the following statement to print to the logcat window:
  println("CreateTextField - LaunchedEffect executed")

- MainScreen - Inside LaunchedEffect it should print a message to the logcat window. Use the following statement to print to the logcat window:
  println("MainScreen - LaunchedEffect executed")

Run the app. The LanuchedEffect messages should appear once for each function call. There should be a total of 5 LaunchedEffect messages.

Rotate the device

Run the app and do the following:

- Check logcat message after typing in both TextFields a few times. The counts should be greater than 1.
- Do a left rotate on the device (the emulator in this case).
- Check the logcat messages. The UI is recreated from the beginning again because the left rotation causes a configuration change on the device. All functions should have LaunchedEffect called on them again. All CreateTextField counts should be reset to 1 again. Any text that was typed in the TextFields will be gone.

Use rememberSaveable

- Update each function variable to use rememberSaveable instead of remember.
- Run the app and type in the TextFields (counts will be updated). Now do a left rotation. The counts will NOT be reset this time because all variables that use rememberSaveable will be saved through the configuration change caused by the left rotation. Any text typed in the TextFields should also be retained.