BCS 371 Lab – NavHost Pass Data Between Screens

Overview

In this lab you will create an app that uses a NavHost and has functionality to pass data between screens.

Create a project

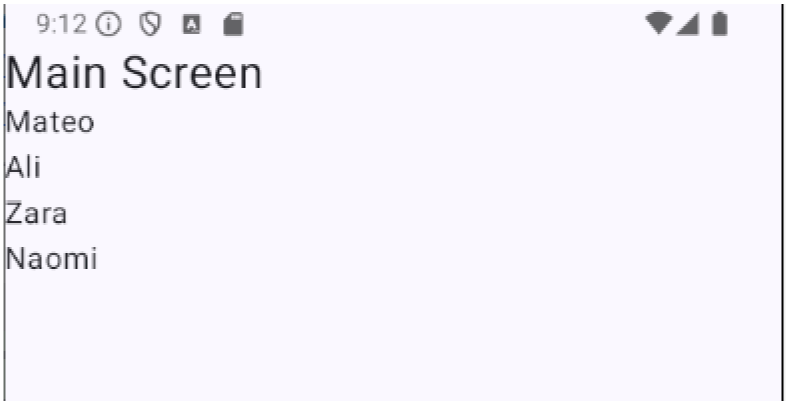
Create a new Android application in Android Studio. Choose the **Empty Activity** type to create an empty activity that uses Jetpack Compose.

Set up the main screen basic functionality

Create a Kotlin file named MainScreen.kt. Add a composable function named MainScreen. Here are the specifications for this function:

- Add a LazyColumn and populate it with names. You can use the following for the list:
  var nameList = mutableListOf("Mateo", "Ali", "Zara", "Naomi")
- If an item is clicked in the LazyColumn it should display a toast containing the name that was clicked.

It should look like the following:



9:12

Main Screen

Mateo

Ali

Zara

Naomi

Setup MainActivity

Update MainActivity.onCreate so that it will call the MainScreen function.

Run the App

You should see the main screen appear when you run the app. Click a name and confirm that a toast displaying that name appears.

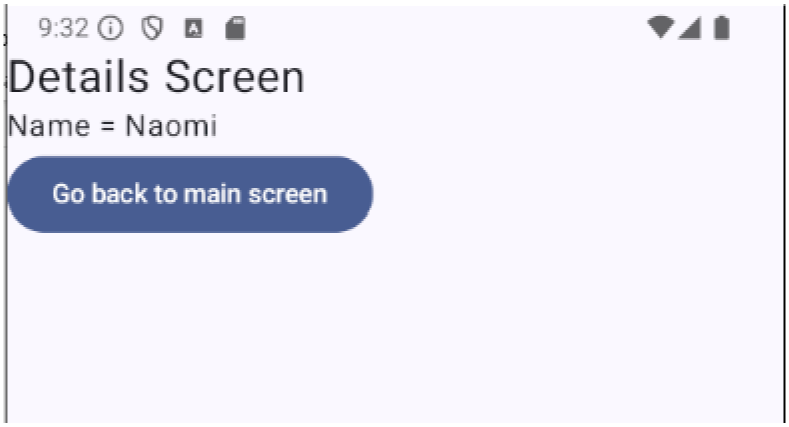
Update MainScreen

Do the following:

- Change the MainScreen function header so that it also takes a NavHostController as a parameter.
- Update the LazyColumn item click event handler. If an item is clicked in the LazyColumn it should navigate to the details screen (details screen described in the next section). It should pass the name in the item as data to the details screen. The details screen should then display the name that was passed to it. The NavHost is described in an upcoming section.

Set up the details screen

Create a Kotlin file name DetailsScreen.kt. Add a composable function named DetailsScreen. Add a parameter for a name string (this name should be displayed in the UI). Here is
what it should look like if “Naomi” was clicked on the main screen:



Details Screen

Name = Naomi

If the Go back to main screen button is pressed it should navigate back to the main screen.

IMPORTANT! It should pop the back stack to get back to main.

Setup the NavHost

Create a Kotlin file named Nav.kt. Add a composable function named Nav. Do the following in this function:

- Create a NavController instance.
- Define a NavHost which has both the main screen and the details screen as destinations. Need to set up the details screen destination so that it takes a parameter.
- Set the main screen as the starting destination in the NavHost.

Setup MainActivity

Update the MainActivity.onCreate so that it calls Nav (instead of MainScreen).

Run the App

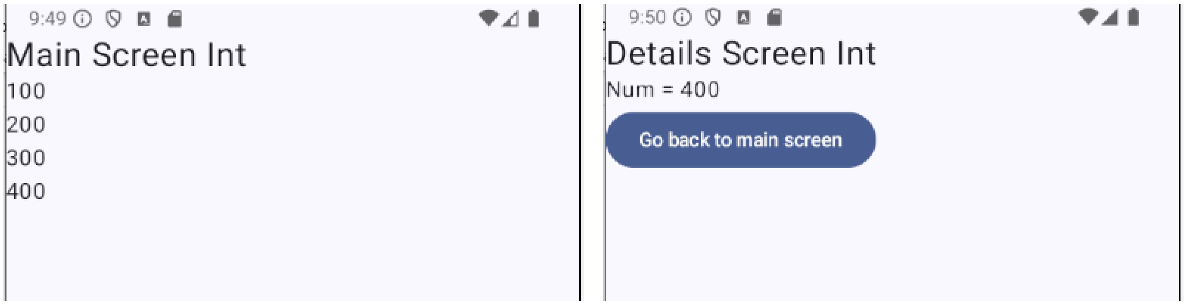
You should see the main screen appear when you run the app. Click a name and confirm that it navigates to the details screen. The name that was clicked should appear in the details screen.

Pass int data to another screen

Do the following:

- Create a function DetailsScreenInt. It should be like DetailsScreen except it takes an Int instead of a String as the second parameter.
- Create a function MainScreenInt. It should be like MainScreen except it contains a LazyColumn of Int data (instead of String data). The LazyColumn event handler should navigate to DetailsScreenInt (not DetailsScreen).
  Note: If you are still displaying a toast, make sure to call toString on the Int data that is passed to the makeText function. If you do not, then it will treat the int as a string resource id (instead of a normal Int) and it will throw an exception.
- If an item is clicked in the LazyColumn of Int it should navigate to DetailsScreenInt. It should pass the Int as data to DetailsScreenInt. It should then display the Int that was passed to it. You can use the following for the list:
  var numList = mutableListOf(100, 200, 300, 400)
- Update the NavHost as follows:
  - Add a composable for DetailsScreenInt. Use a navArgument with type NavType.IntType.
  - Add a composable for MainScreenInt.
  - Set the startDestination to MainScreenInt.

Here are screenshots before and after clicking 400:



Main Screen Int

100
200
300
400

Details Screen Int

Num = 400