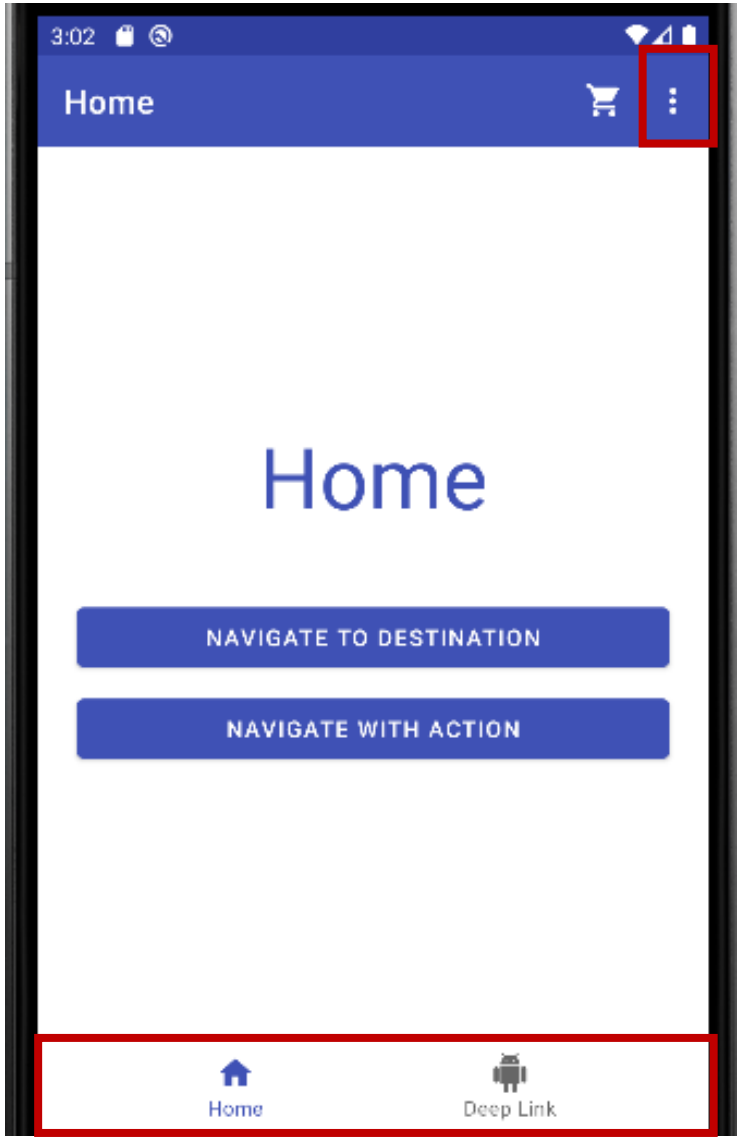
Lab 04
Android Navigation

Software Studio

DataLab, CS, NTHU

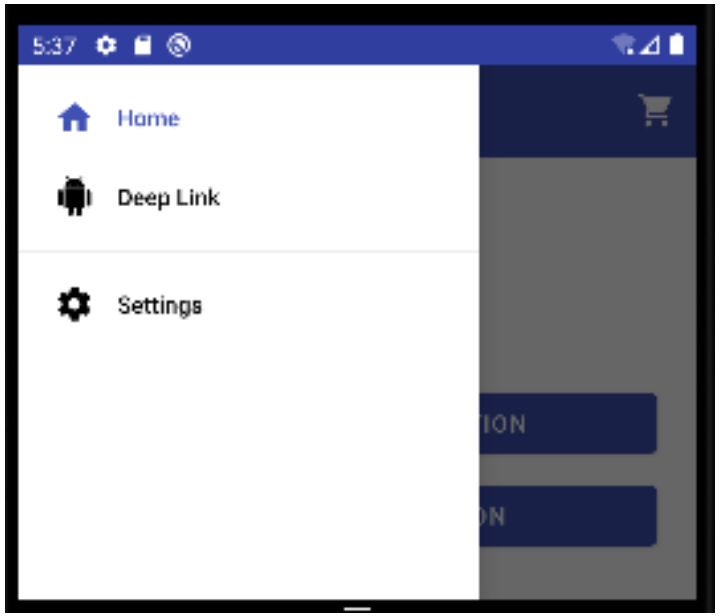
2022 spring

Navigation



1. Option Menu

3. Navigation Drawer

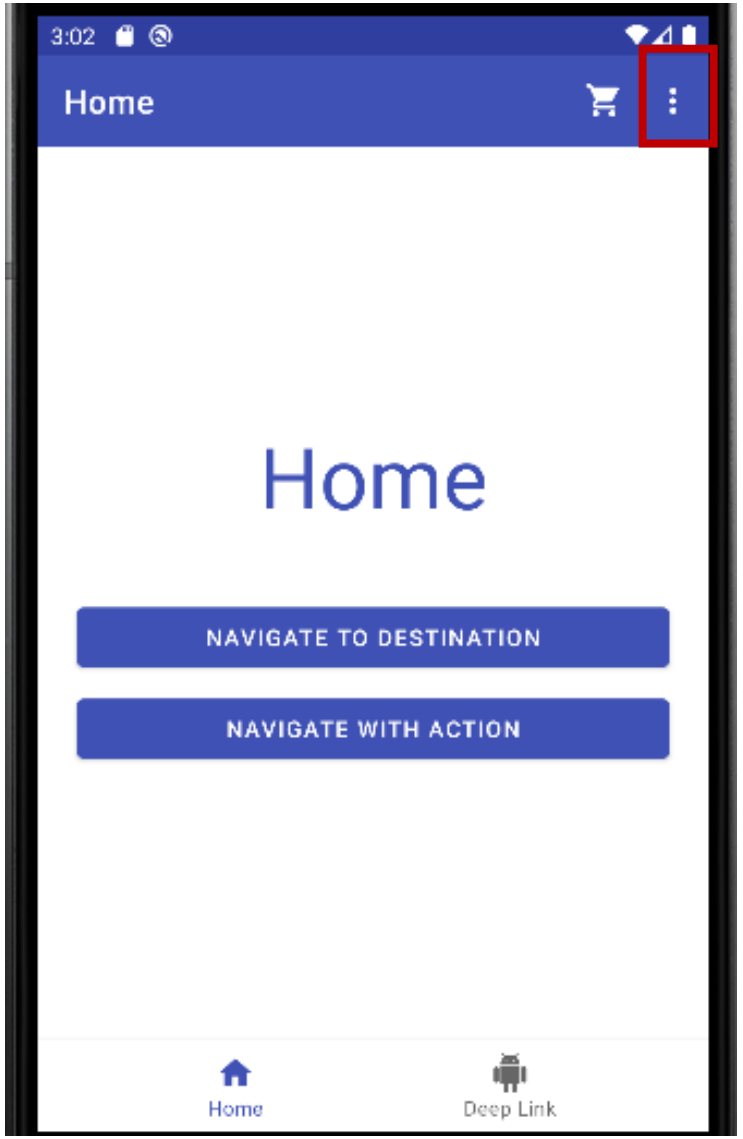


2. Bottom Navigation

Navigation UI and navigation-ui-ktx

- The Navigation Components include a NavigationUI class and the navigation-ui-ktx kotlin extensions.
- If NavigationUI finds a menu item with the same ID as a destination on the current graph, it configures the menu item to navigate to that destination.

Navigation



1. Option Menu

NavigationUI with an Options menu

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <!-- TODO Menu 1 - Add an item for the settings fragment -->
    <item
        android:id="@+id/settings_dest"
        android:icon="@drawable/ic_settings"
        android:menuCategory="secondary"
        android:title="Settings" />
    <!-- TODO END Menu 1 -->
</menu>
```

Res/menu/overflow_menu.xml

Res/navigation/mobile_navigation.xml

```
    <fragment
        android:id="@+id/settings_dest"
        android:name="com.example.android.codelabs.navigation.SettingsFragment"
        android:label="Settings"
        tools:layout="@layout/settings_fragment" />
```

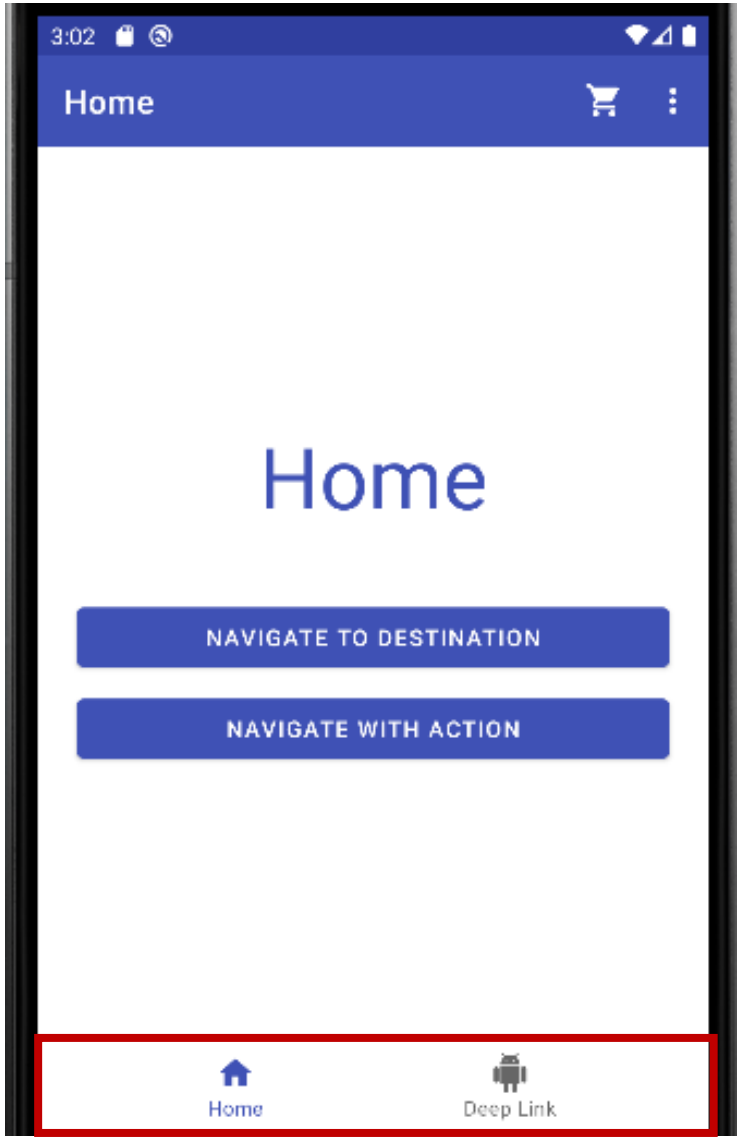
NavigationUI with an Options menu

- Implement the `onOptionsItemSelected`

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // TODO MENU 1 - Have Navigation UI Handle the item selection
    // Have the NavigationUI look for an action or destination matching the menu
    // item id and navigate there if found.
    // Otherwise, bubble up to the parent.
    return item.onNavDestinationSelected(findNavController(R.id.my_nav_host_fragment))
            || super.onOptionsItemSelected(item)
    // TODO END Menu 1
}
```

MainActivity.kt

Navigation



2. Bottom Navigation

Navigation UI to configure Bottom Navigation

```
<com.google.android.material.bottomnavigation.BottomNavigationView
    android:id="@+id/bottom_nav_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:menu="@menu/bottom_nav_menu" />
```

res/layout/navigation_activity/navigation_activity.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@id/home_dest"
        android:icon="@drawable/ic_home"
        android:title="Home" />
    <item
        android:id="@id/deeplink_dest"
        android:icon="@drawable/ic_android"
        android:title="@string/deeplink" />
</menu>
```

their ids match the destinations of navigation graph destinations

bottom_nav_menu.xml

Navigation UI to configure Bottom Navigation

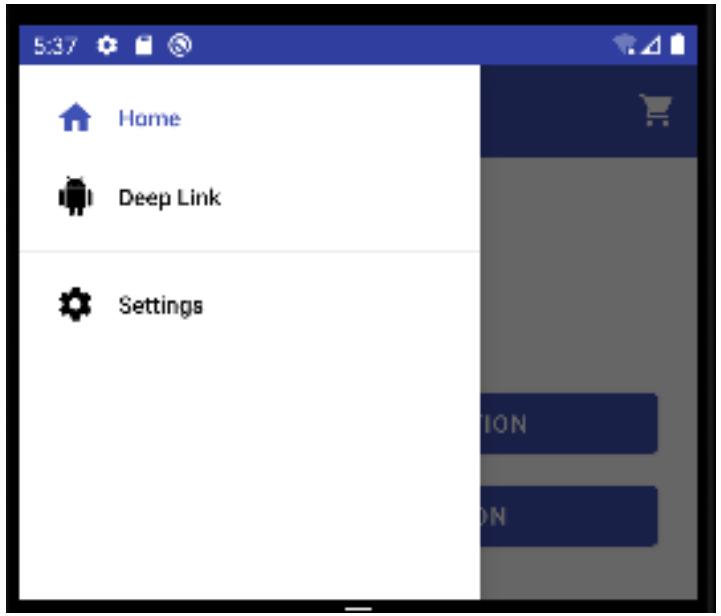
- Implement the `setupBottomNavMenu`

```
private fun setupBottomNavMenu(navController: NavController) {
    // TODO MENU 2 - Use NavigationUI to set up Bottom Nav
    val bottomNav = findViewById<BottomNavigationView>(R.id.bottom_nav_view)
    bottomNav?.setupWithNavController(navController)
    // TODO END MENU 2
}
```

MainActivity.kt

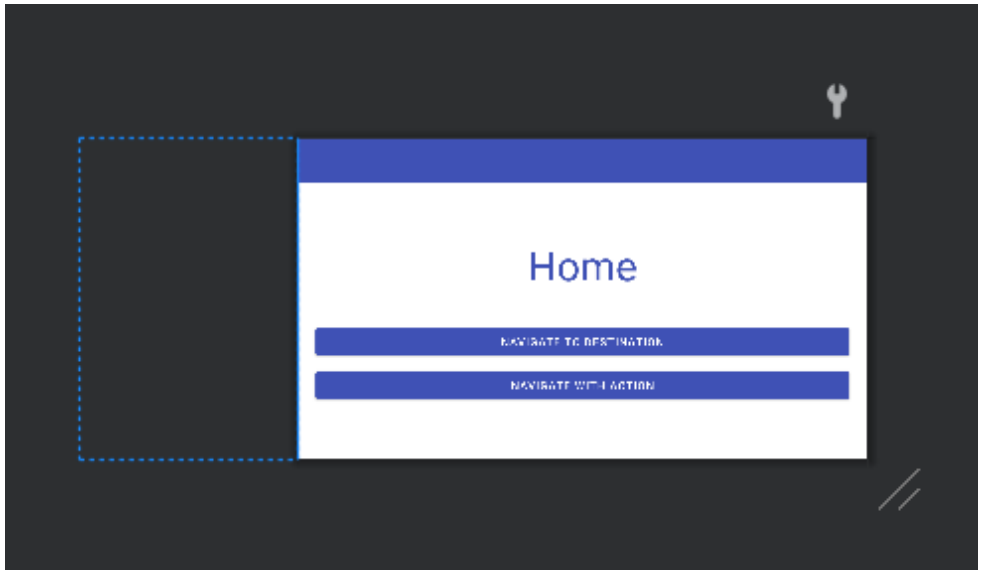
Navigation

3. Navigation Drawer

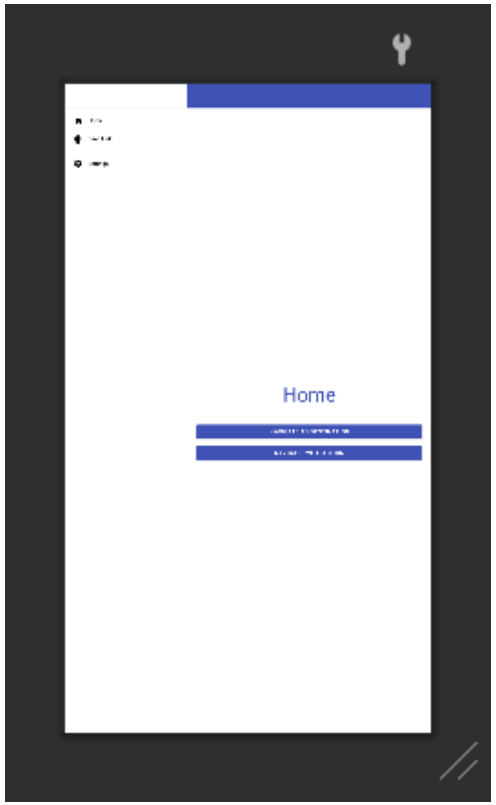


Navigation UI to configure Navigation Drawer

- You'll see this if you've got a large enough screen or if the screen is too short for bottom navigation.



navigation_activity.xml



navigation_activity.xml (w960dp)

Navigation UI to configure Navigation Drawer

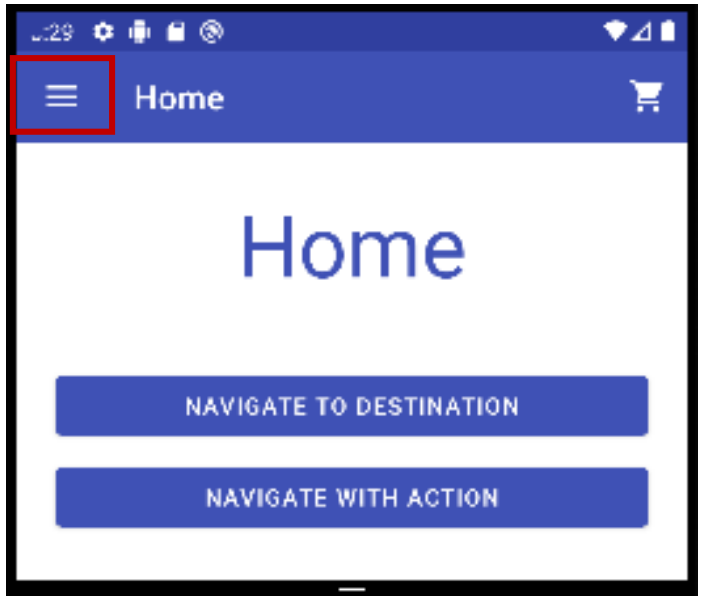
- Implement the `setupNavigationMenu`

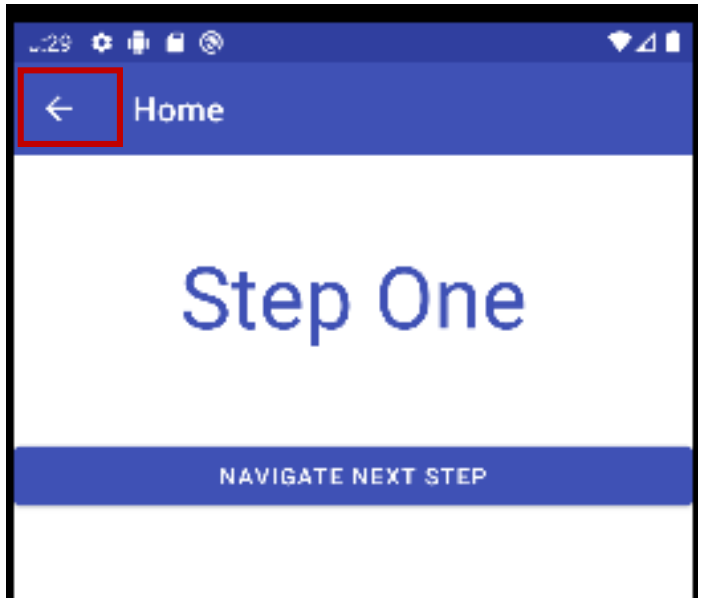
```
private fun setupNavigationMenu(navController: NavController) {
    // TODO MENU 3 - Use NavigationUI to set up a Navigation View
    // In split screen mode, you can drag this view out from the left
    // This does NOT modify the actionbar
    val sideNavView = findViewById<NavigationView>(R.id.nav_view)
    sideNavView?.setupWithNavController(navController)
    // TODO END MENU 3
}
```

MainActivity.kt

Navigation UI to configure Navigation Drawer

- Set up the **ActionBar** requires creating an instance of AppBarConfiguration.
- Purpose of AppBarConfiguration is to specify the configuration options for toolbars.





Navigation UI to configure Navigation Drawer

- In MainActivity - onCreate

```
// TODO MENU 3 - Create an AppBarConfiguration with the correct top-level destinations
val drawerLayout : DrawerLayout? = findViewById(R.id.drawer_layout)
appBarConfiguration = AppBarConfiguration(
        setOf(R.id.home_dest, R.id.deeplink_dest),
        drawerLayout)
// TODO END MENU 3
```

Set top-level destinations

MainActivity.kt

Navigation UI to configure Navigation Drawer

- Show a title in the ActionBar based off of the destination's label
- Display the Up button whenever you're **not** on a top-level destination
- Display a drawer icon (hamburger icon) when you're on a top-level destination

```
private fun setupActionBar(navController: NavController,
                           appBarConfig : AppBarConfiguration) {
    // TODO MENU 3 - Have NavigationUI handle what your ActionBar displays
    // This allows NavigationUI to decide what label to show in the action bar
    // By using appBarConfig, it will also determine whether to
    // show the up arrow or drawer menu icon
    setupActionBarWithNavController(navController, appBarConfig)
    // TODO END MENU 3
}
```

MainActivity.kt

Navigation UI to configure Navigation Drawer

- Implement the `onSupportNavigateUp`

```kotlin
// TODO MENU 3 - Have NavigationUI handle up behavior in the ActionBar
override fun onSupportNavigateUp(): Boolean {
    // Allows NavigationUI to support proper up navigation or the drawer layout
    // drawer menu, depending on the situation
    return findNavController(R.id.my_nav_host_fragment).navigateUp(appBarConfiguration)
}
// TODO END MENU 3
```

MainActivity.kt

Navigation Transition

```
val options = navOptions {
    anim {
        enter = R.anim.slide_in_right
        exit = R.anim.slide_out_left
        popEnter = R.anim.slide_in_left
        popExit = R.anim.slide_out_right
    }
}
view.findViewById<Button>(R.id.navigate_destination_button)?.setOnClickListener{
    findNavController().navigate(R.id.flow_step_one_dest, null, options)
}
```