WIA2007 Mobile Application Development
Semester 1, Session 2022/2023
Practical 8 (App Bar and Navigation View)

Task 1: Implement the App Bar, Side Navigation Bar, and Bottom Navigation Bar in Lecture 7

In this task, you are required to **use your project from Practical 7** to **implement the codes learned in Lecture 7.**

After the implementation of the codes given in Lecture 7, as a summary, your following files should be in this state:

- layout/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/DLMain"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/TBMainAct"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="?attr/colorPrimary"
        android:minHeight="?attr/actionBarSize" />

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/NHFMain"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        app:defaultNavHost="true"
        app:navGraph="@navigation/nav_animal_lover" />


<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/menu_bottom" >


</com.google.android.material.bottomnavigation.BottomNavigationView
>

    </LinearLayout>
```

```
    <com.google.android.material.navigation.NavigationView
        android:id="@+id/sideNav"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        app:headerLayout="@layout/nav_header"
        app:menu="@menu/menu_side" />

</androidx.drawerlayout.widget.DrawerLayout>
```

- menu/menu_bottom.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@id/DestHome"
        android:icon="@android:drawable/ic_menu_agenda"
        android:title="Home" />
    <item
        android:id="@id/DestAboutApp"
        android:icon="@android:drawable/ic_menu_info_details"
        android:title="@string/aboutapp" />
</menu>
```

- values/themes/theme.xml

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.NavGraph"
parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor"
tools:targetApi="l">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

- MainActivity.java

```
package com.example.navgraph;

import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.drawerlayout.widget.DrawerLayout;
```

```
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.fragment.NavHostFragment;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;

import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

import
com.google.android.material.bottomnavigation.BottomNavigationView;
import com.google.android.material.navigation.NavigationView;

public class MainActivity extends AppCompatActivity {
    private ActionBarDrawerToggle toggle;
    private DrawerLayout drawerLayout;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Toolbar toolbar = findViewById(R.id.TBMainAct);
        setSupportActionBar(toolbar);

        drawerLayout = findViewById(R.id.DLMain);
        toggle = new ActionBarDrawerToggle(
                this, drawerLayout, toolbar,
R.string.navigation_drawer_open, R.string.navigation_drawer_close);
        drawerLayout.addDrawerListener(toggle);
        toggle.syncState();

        NavHostFragment host = (NavHostFragment)
getSupportFragmentManager().findFragmentById(R.id.NHFMain);
        NavController navController = host.getNavController();

        AppBarConfiguration appBarConfiguration = new
AppBarConfiguration.Builder(navController.getGraph()).build();
        NavigationUI.setupActionBarWithNavController(this,
navController, appBarConfiguration);

        setupBottomNavMenu(navController);
        setupNavMenu(navController);
    }

    private void setupNavMenu(NavController navController){
        NavigationView sideNav = findViewById(R.id.sideNav);
        NavigationUI.setupWithNavController(sideNav,
navController);
    }

    private void setupBottomNavMenu(NavController navController) {
        BottomNavigationView bottomNav =
findViewById(R.id.bottom_nav_view);
        NavigationUI.setupWithNavController(bottomNav,
navController);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_bottom, menu);
```

```
            return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            try {
                Navigation.findNavController(this,
R.id.NHFMain).navigate(item.getItemId());
                return true;
            }
            catch (Exception ex)
            {
                return super.onOptionsItemSelected(item);
            }
        }
}
```

After the implementation, you should be able to see the following:

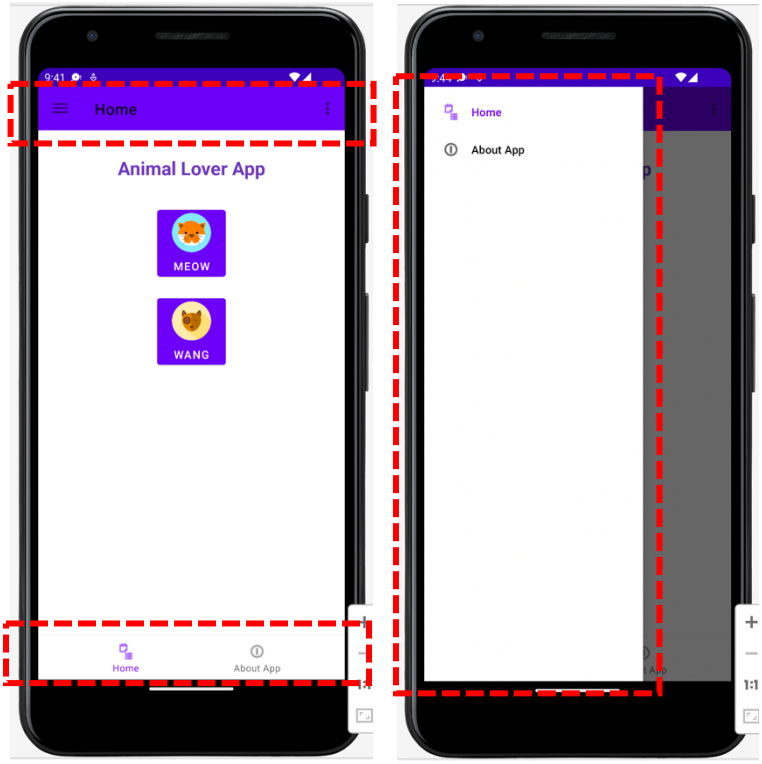

Figure 1: The left figure shows the App Bar and Bottom Navigation Bar while the right figure shows the Side Navigation Bar.

Once you have all the codes from Lecture 7 ready, let's continue to Task 2!

Task 2: Optimizing the Side Navigation View

The side Navigation Bar is often utilized to store many links (of different hierarchical levels) to enable various navigations in the mobile application. When we have many links on the Side Navigation Bar, we should organize or group the links to increase user accessibility to these paths.

In this mobile application, we have four paths to Home, About App, Cat and Dog fragments. Now, let's create a new menu resource file called menu_side.xml in the menu resource directory to start grouping them.

Tips: to create the new menu resource file, you can right-click on the menu resource directory to create a new menu resource file, enter the File name (menu_side) and click "OK" to create the xml file.

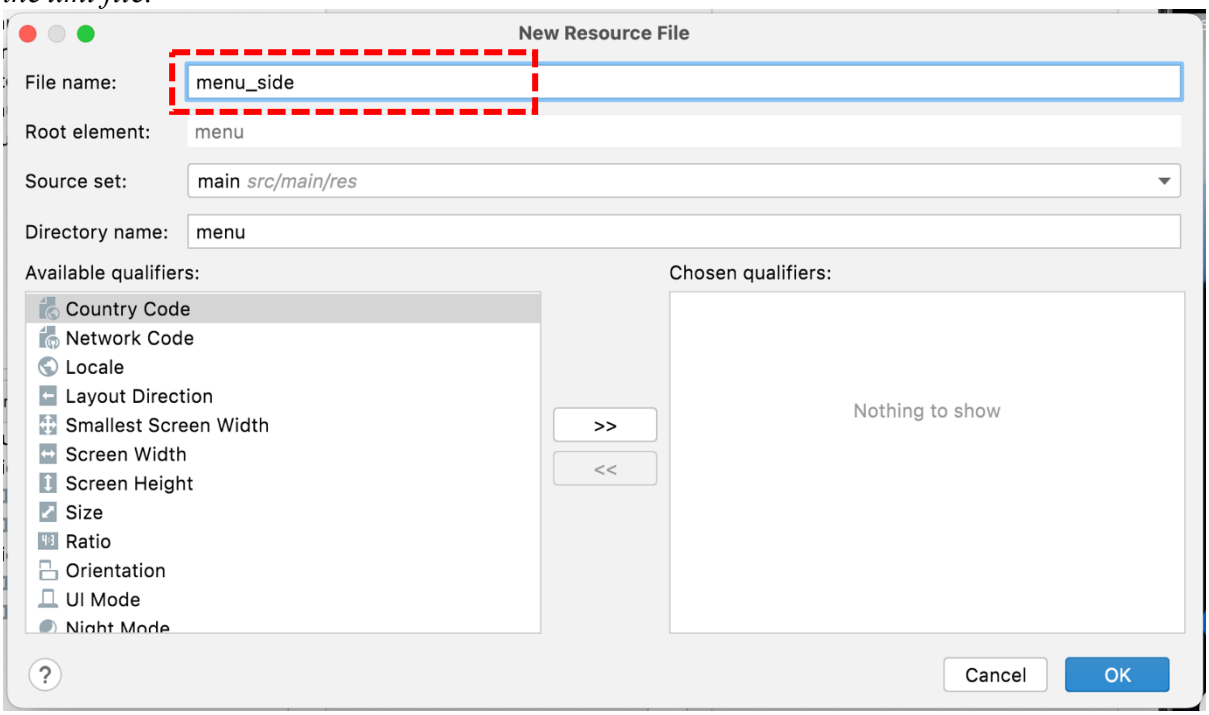

Figure 2: Creating menu_side.xml.

We will divide the into two groups – the **main** group that can access to the top-level navigation (Home and About App), and the **animal** group that can access to the animal fragments (Cat and Dog).

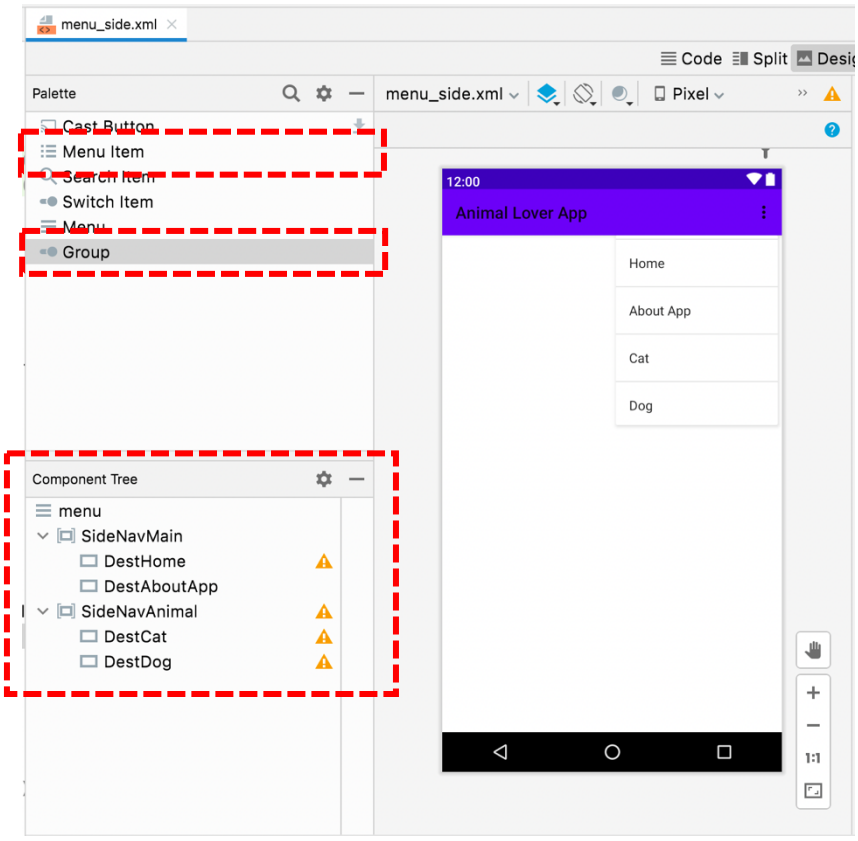

Figure 3: Group the menu item.

On Figure 3, there is a "Group" widget available in the Palette Panel. Drag and Drop two "Group" widgets on the menu_side.xml canvas, or to the Component Tree Panel. Then, drag and drop the "Menu Item" for all fragments. You can move the menu items in the Component Tree Panel to group them. (Note: make sure to change the name of the menu items to DestHome, DestAboutApp, DestCat and DestDog to allow the navigation controller to locate these fragment for navigation purposes.)

Then, on the activity_main.xml file, find the Side Navigation View, and change the menu source file to @menu/menu_side.

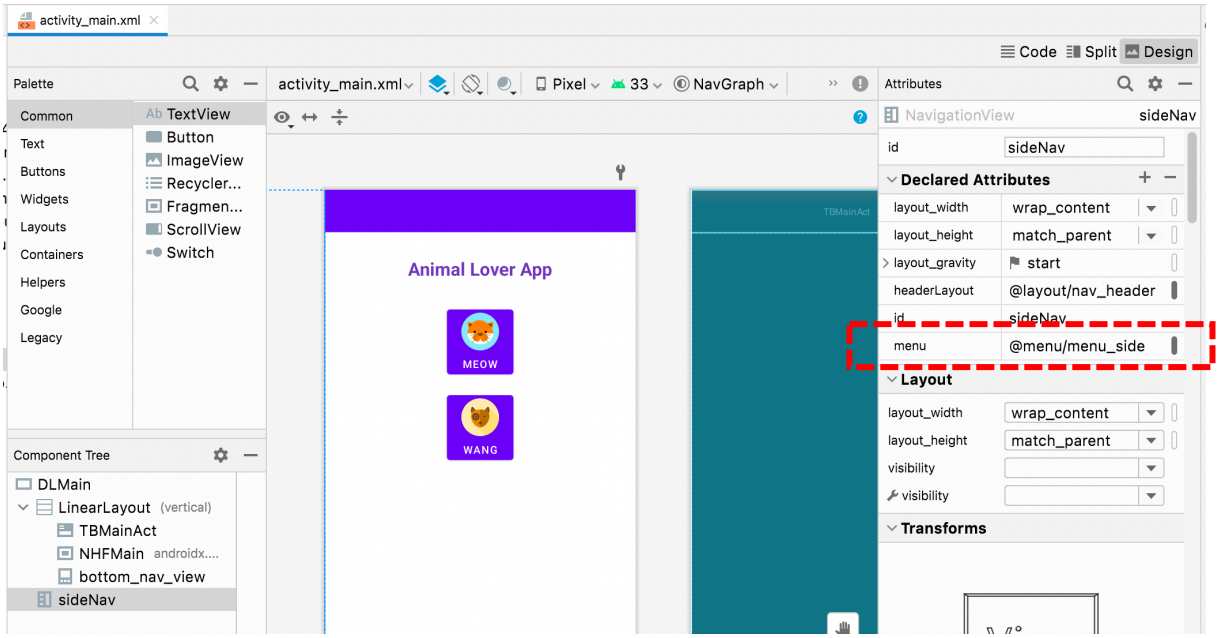

Figure 4: Update the menu source for the side navigation view.

Now, try run the emulator and you shall see the items in the Side Navigation Bar is grouped.

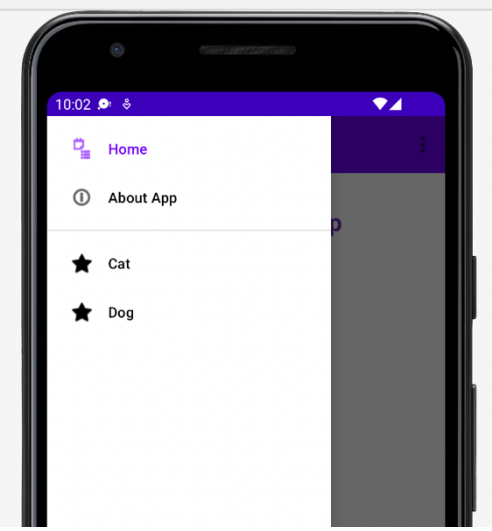

Figure 5: The grouped Side Navigation Bar.

Task 3: Header Layout for Side Navigation View

A Side Navigation Bar normally contains two major components – the header view and the menu. The header view will be displayed on the top part of the Side Navigation Bar, using a separate layout file, to display the Profile Picture, descriptions, etc.

Now, let's create a simple layout file for the header. On the layout resource directory, right click to create a new layout resource file, and let's name it as nav_header.xml.

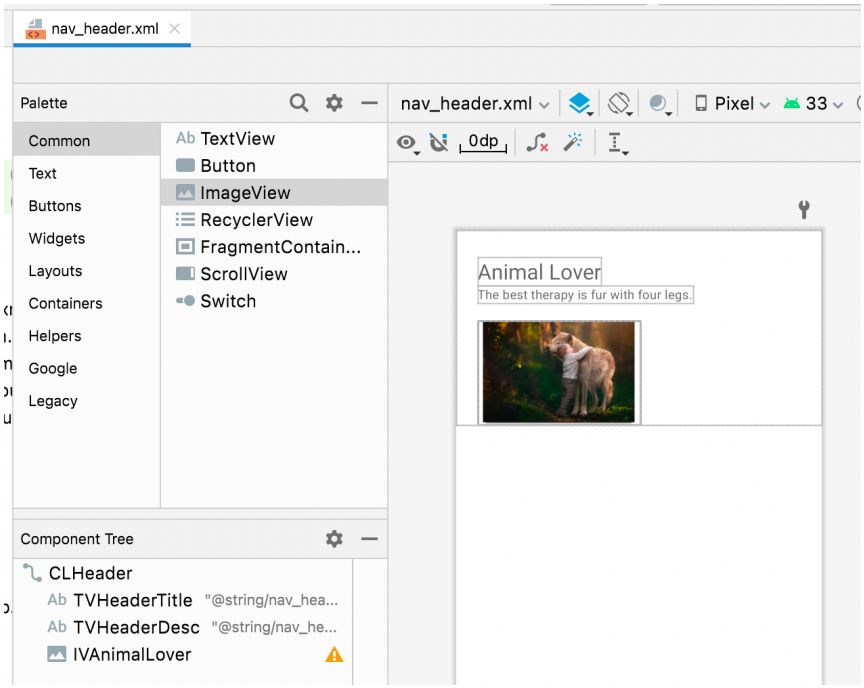

Figure 6: The simple header view for Side Navigation Bar.

We are creating a simple header layout file for demonstration purposes here. In Figure 6, it shows that the header view has two TextView (for the title and description of the App), and an ImageView.

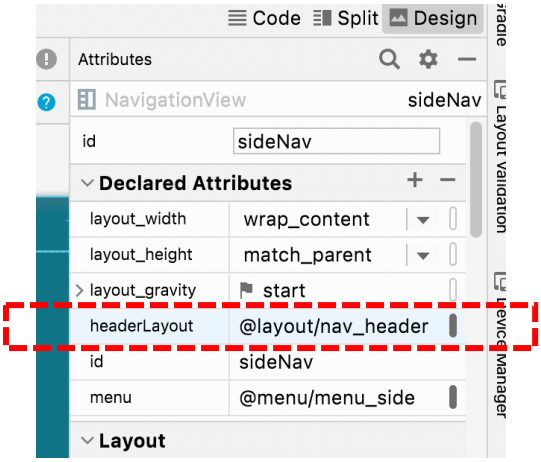

Figure 7: Link the header layout file to the Side Navigation Bar.

After that, on the activity_main.xml, locate the Side Navigation View, and update the headerLayout attribute to add the header view to the Side Navigation Bar.

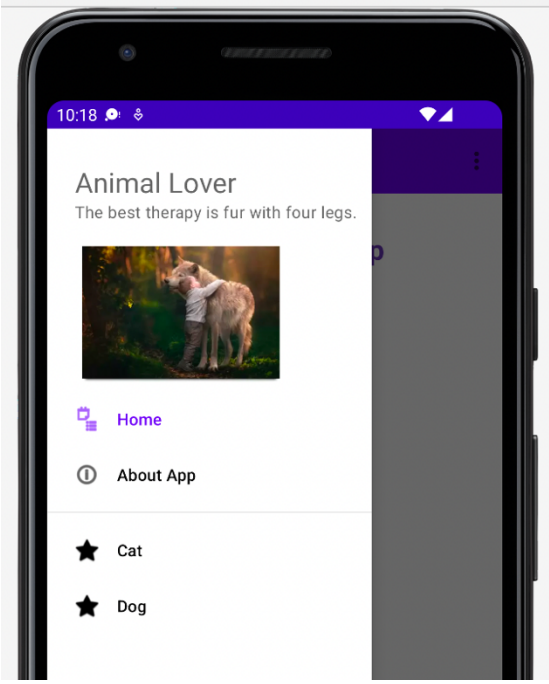

Figure 7: The Side Navigation Bar with header view.

Now, try run your mobile application and you shall see this the header view as shown in Figure 7.

Task 4: Action on the App Bar

In the App Bar, some important functions for the particular activity will be placed to allow the handy movement when user is using the mobile application. These important functions are often coming from the Overflow menu.

On the menu resource file, you can use the attribute called showAsAction to decide whether to show the menu item as Action that will be displayed on the App Bar. If you set showAsAction to "ifRoom", the menu item will be placed on the App Bar if there is room available (depending on the screen size), otherwise if you set it to "never", then it will be always showed as menu item inside the Overflow menu. There are few more settings you can set to the showAsAction attribute (refer to Figure 8).

`android:showAsAction`

Keyword. When and how this item should appear as an action item in the app bar. A menu item can appear as an action item only when the activity includes an app bar. Valid values:

Value	Description
`ifRoom`	Only place this item in the app bar if there is room for it. If there is not room for all the items marked "`ifRoom`", the items with the lowest `orderInCategory` values are displayed as actions, and the remaining items are displayed in the overflow menu.
`withText`	Also include the title text (defined by `android:title`) with the action item. You can include this value along with one of the others as a flag set, by separating them with a pipe |.
`never`	Never place this item in the app bar. Instead, list the item in the app bar's overflow menu.
`always`	Always place this item in the app bar. Avoid using this unless it's critical that the item always appear in the action bar. Setting multiple items to always appear as action items can result in them overlapping with other UI in the app bar.
`collapseActionView`	The action view associated with this action item (as declared by `android:actionLayout` or `android:actionViewClass`) is collapsible. Introduced in API Level 14.

Figure 8: The possible settings for showAsAction attribute.

menu_bottom.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:app="http://schemas.android.com/apk/res-auto"
3      xmlns:android="http://schemas.android.com/apk/res/android">
4
5      <item
6          android:id="@id/DestHome"
7          android:icon="@android:drawable/ic_menu_agenda"
8          android:title="Home"
9          app:showAsAction="never"/>
10     <item
11         android:id="@id/DestAboutApp"
12         android:icon="@android:drawable/ic_menu_info_details"
13         android:title="About App"
14         app:showAsAction="ifRoom"/>
15 </menu>
16
```

Figure 9: Setting the items showAsAction attribute.

Now, let's try experimenting with the settings. Figure 9 shows that the menu_bottom.xml file is used to set the showAsAction to "never" and "ifRoom". When you test run the mobile application, you should see the About App menu item is moved to the App Bar, while the Home menu item is reside the Overflow menu (see Figure 10).

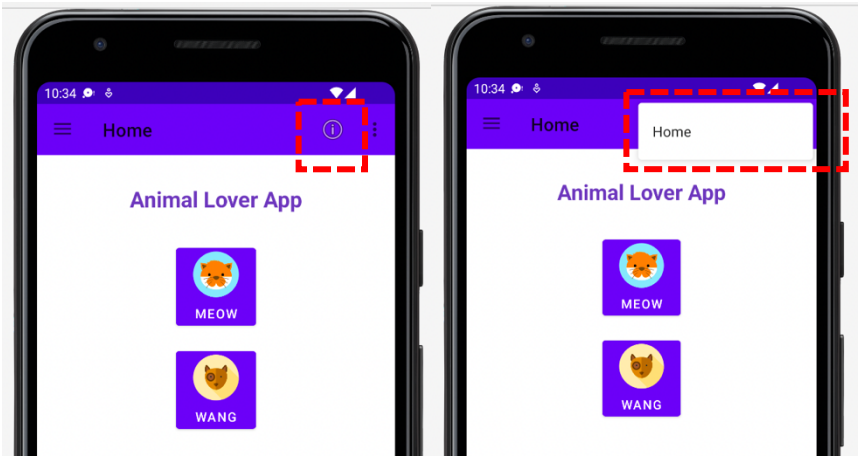

Figure 10: The About App menu item is shown as Action (see the left figure) while the Overflow menu only contains the Home item.

Submission

There are many attributes that you can explore in the App Bar and these navigation views. Beautify the app and complete these tasks in the practical exercise.

You are required to submit the Android Studio Project to Spectrum before the next Tutorial class.