Lab 3 - Advanced GUI Components

Due Oct 4 by 3:45pm **Points** 10 **Submitting** a file upload
Available Sep 27 at 10am - Oct 4 at 3:45pm 7 days

This assignment was locked Oct 4 at 3:45pm.

Introduction:

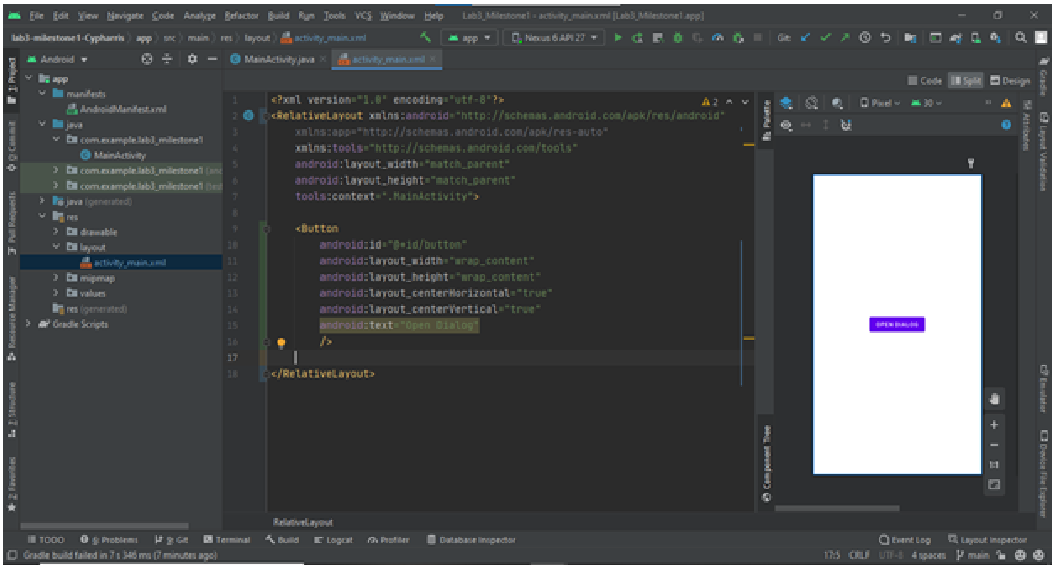
This lab will show you how to work with Advanced GUI components. You will learn how to create Dialogs, Menus, and Fragments. First, we will see how to create dialogs and display information in dialogs. Then we will learn how to create Menus and Fragments and switch between Fragments.

Directions:

Milestone 1 - Dialog Box

In this Milestone, we will be looking at creating a Dialog and displaying some information in it.

1. First, we are going to see how to use dialogs in Android
2. As you did in lab 2, clone the Lab 3 Milestone 1 repository by selecting File > New > Project from Version Control > Git
3. Then click the following link and accept the invitation: https://classroom.github.com/a/Tnguaa4F (https://classroom.github.com/a/Tnguaa4F)
4. You'll get a link in the form https://github.com/CS407-Fall-2021/lab3-milestone1-yourgithubnamehere
5. Copy this link in the URL text field in 'Clone' in Android Studio and click 'Clone'.
6. Change the "Constraint Layout" to "Relative Layout" (Line No. 2 and 18 in the picture below). Then set up a button at the center of the app as shown below. On pressing the button, we'll display a dialog box.

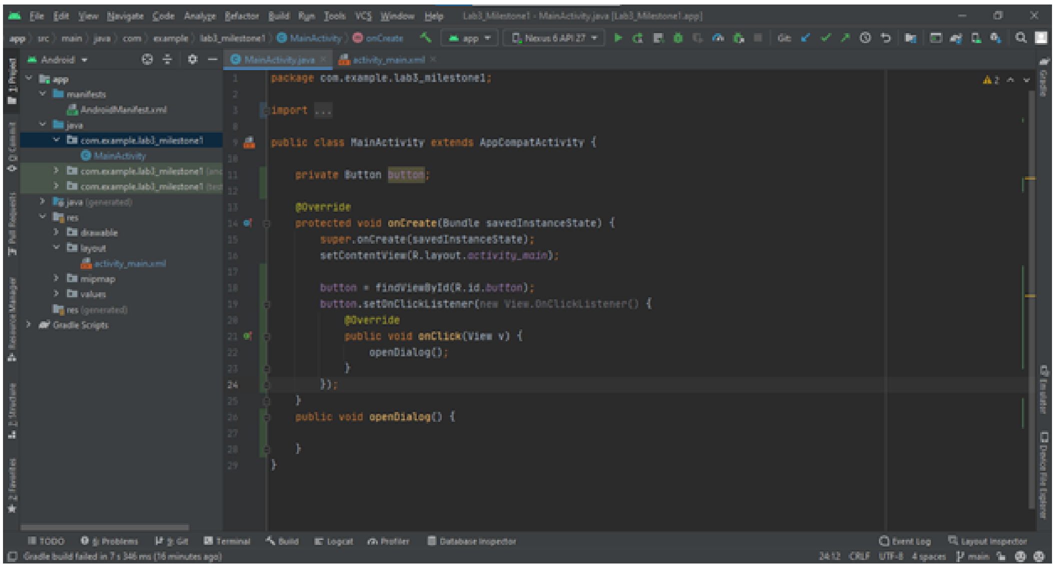


```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity">
8
9      <Button
10         android:id="@+id/button"
11         android:layout_width="wrap_content"
12         android:layout_height="wrap_content"
13         android:layout_centerHorizontal="true"
14         android:layout_centerVertical="true"
15         android:text="Open Dialog"
16         />
17
18 </RelativeLayout>
```

7. Next, switch to MainActivity.java. We will create an onClickListener() for the button in the onCreate() method. When a user presses the button, the button object receives an "on-click" event. In order to respond to this event, in the "MainActivity.java" file, we create an event handler

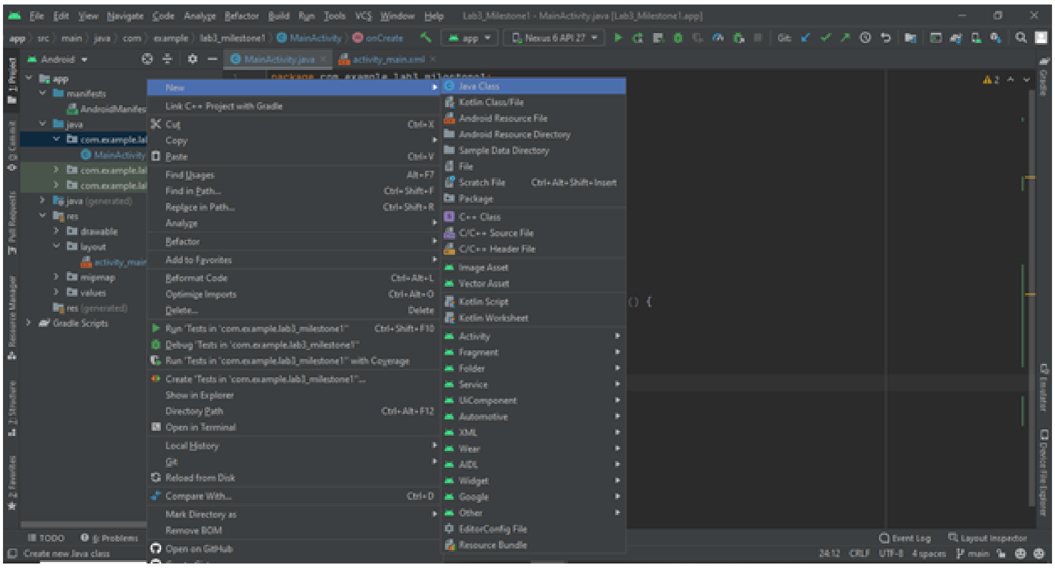
called onClickListener() for the button. This is added in the onCreate() method. You will notice that the onClickListener() contains an onClick() method. Recall from Lab-2 that the "onClick" method is used to specify the action that needs to be taken when the button is pressed.

8. Create a new method openDialog() which will create the dialog. We will define the body of this method in a moment. Call this openDialog() method within the onClick method we created in the previous step. Your code after step-7 and step-8 should look something like this:



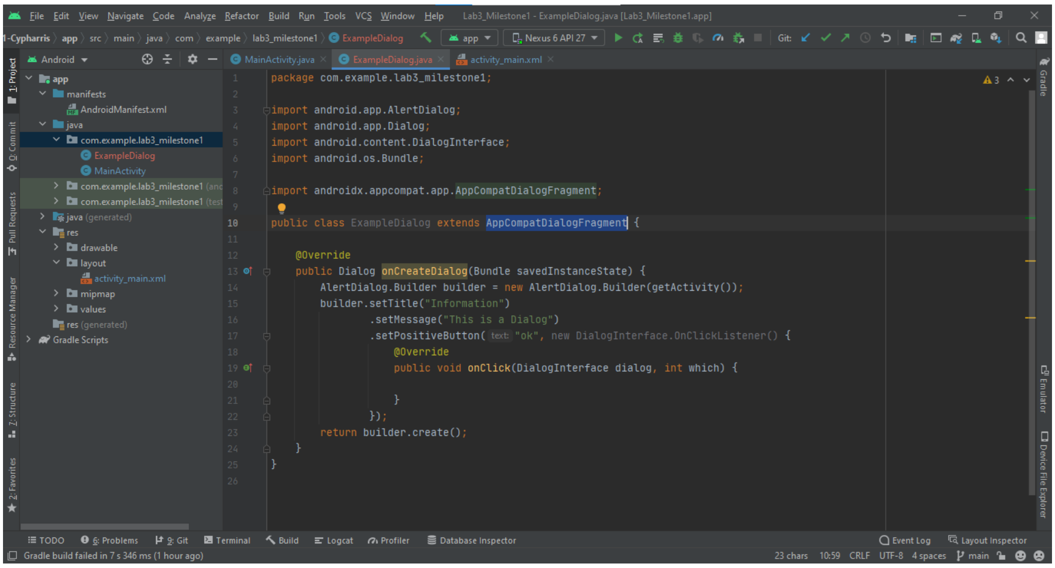
```
package com.example.lab3_milestone1;

import ...

public class MainActivity extends AppCompatActivity {

    private Button button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openDialog();
            }
        });
    }
    public void openDialog() {

    }
}
```

9. Now, we will create a new class. To create a new class, right-click on com.example.your-project-name in the “Project” tab on the left side of Android studio. Select New → Java Class. Name the class “ExampleDialog” and press Enter. Then a new file name “ExampleDialog.java” will be generated:



10. Java provides the reserved word “extends” for specifying a hierarchical relationship between two classes. We extend our class to be able to use the functionality of “AppCompatDialogFragment” class (Line No. 10 in SS below). In this relationship, “AppCompatDialogFragment” is known as the base class, parent class, or superclass; and “ExampleDialog” is known as the derived class, child class, or subclass. Then, we override the onCreateDialog() method within our new class. Create a new AlertDialog instance within this onCreateDialog() method. We set a title and a message for the dialog box. We also set a button that the user can click to make the dialog box disappear. For this example, we leave the onClick() method for this button empty. Here is a snapshot of how the

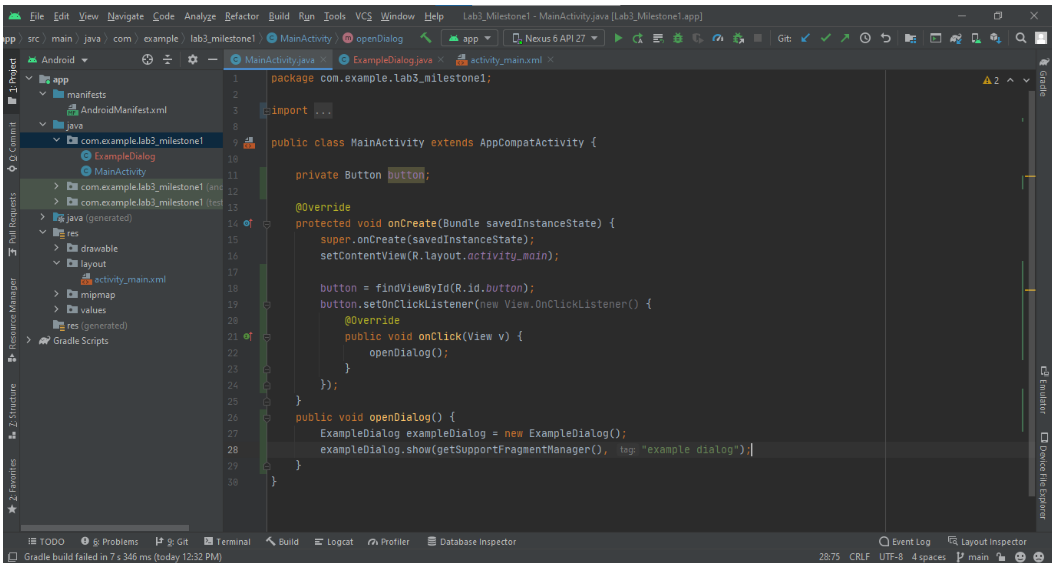
code should look like this. Make sure that you have the import statements added correctly.



```java
package com.example.lab3_milestone1;

import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.os.Bundle;

import androidx.appcompat.app.AppCompatDialogFragment;

public class ExampleDialog extends AppCompatDialogFragment {

    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
        builder.setTitle("Information")
                .setMessage("This is a Dialog")
                .setPositiveButton( text: "ok", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

                    }
                });
        return builder.create();
    }
}
```

Lab3_Milestone1 - ExampleDialog.java [Lab3_Milestone1.app]
MainActivity.java
ExampleDialog.java
activity_main.xml
Gradle build failed in 7 s 346 ms (1 hour ago)

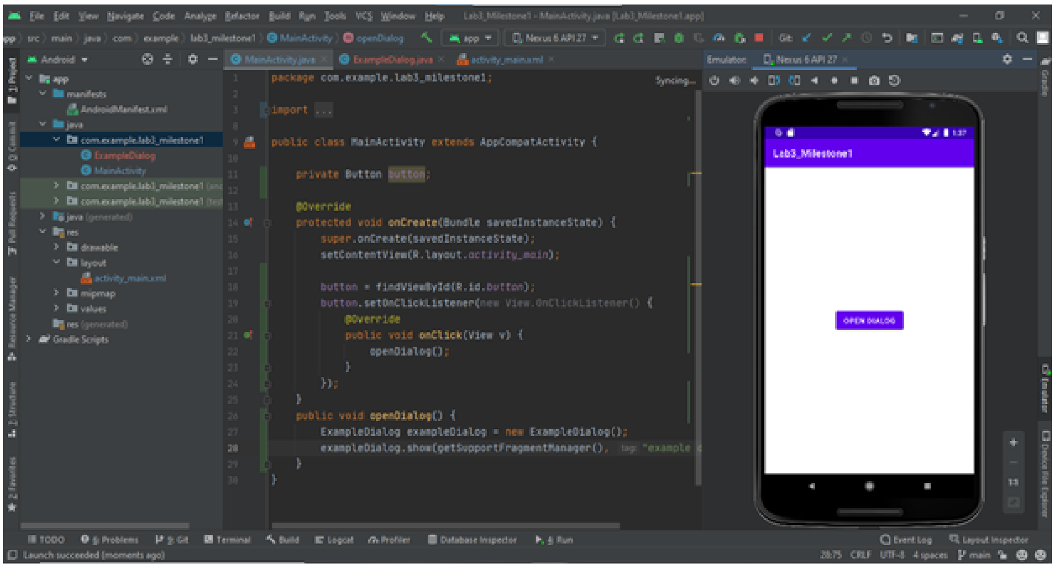
11. Switch to MainActivity.java and in the openDialog() method we created earlier, create an instance of our ExampleDialog class. Now show the dialog.



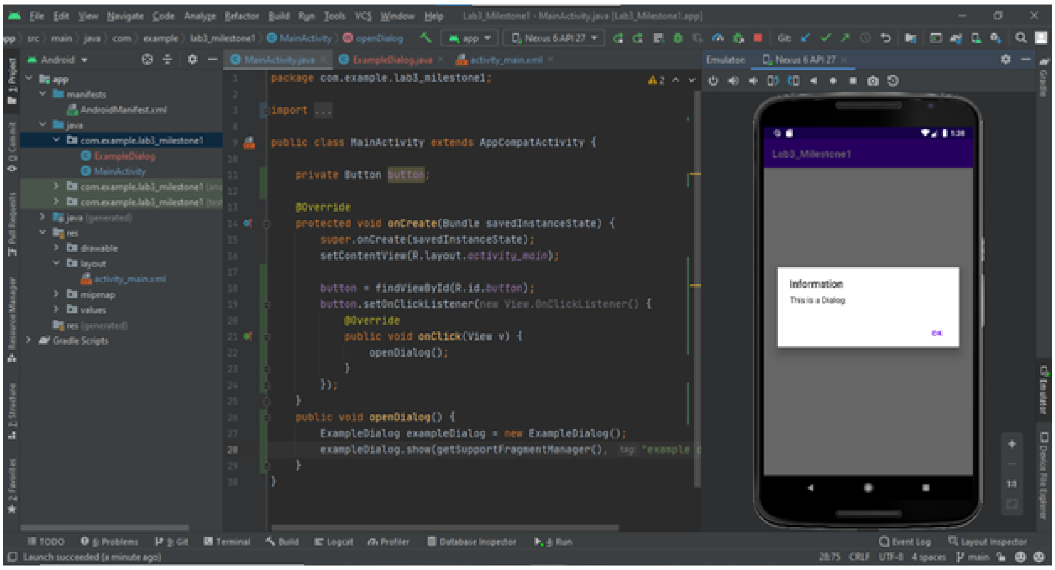
```java
package com.example.lab3_milestone1;

import ...

public class MainActivity extends AppCompatActivity {

    private Button button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openDialog();
            }
        });
    }
    public void openDialog() {
        ExampleDialog exampleDialog = new ExampleDialog();
        exampleDialog.show(getSupportFragmentManager(),  tag: "example dialog");
    }
}
```

Lab3_Milestone1 - MainActivity.java [Lab3_Milestone1.app]
Gradle build failed in 7 s 346 ms (today 12:32 PM)

12. Your app should look like this:



```java
package com.example.lab3_milestone1;

import ...

public class MainActivity extends AppCompatActivity {

    private Button button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openDialog();
            }
        });
    }
    public void openDialog() {
        ExampleDialog exampleDialog = new ExampleDialog();
        exampleDialog.show(getSupportFragmentManager(),  tag: "example
    }
}
```



```
package com.example.lab3_milestone1;

import ...

public class MainActivity extends AppCompatActivity {

    private Button button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openDialog();
            }
        });
    }
    public void openDialog() {
        ExampleDialog exampleDialog = new ExampleDialog();
        exampleDialog.show(getSupportFragmentManager(), tag: "example d
    }
}
```

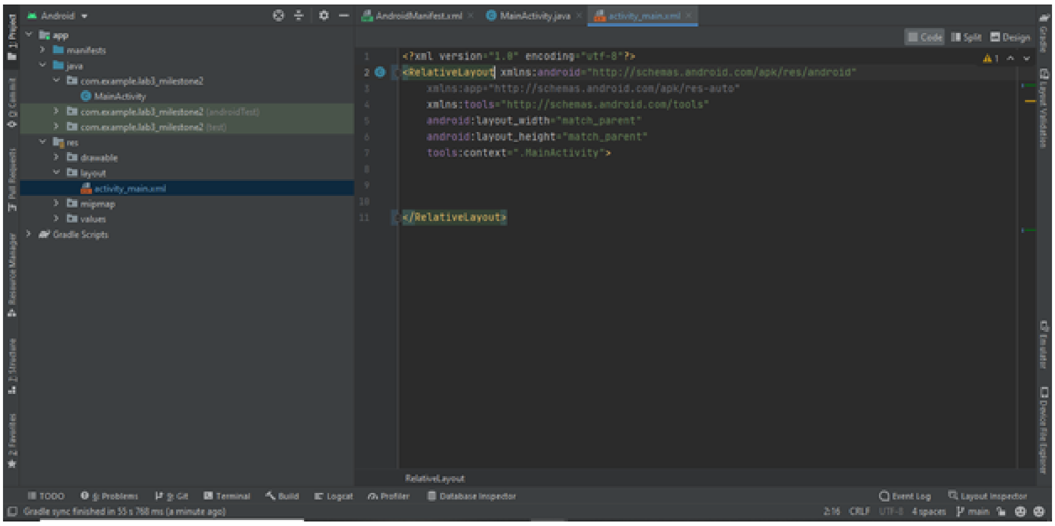
DELIVERABLES:

- Show the correct functioning of the Dialog Box to a peer mentor or TA
- Commit and push the project to the GitHub classroom

Milestone 2:

In this Milestone, we will be creating Menu and Fragments.

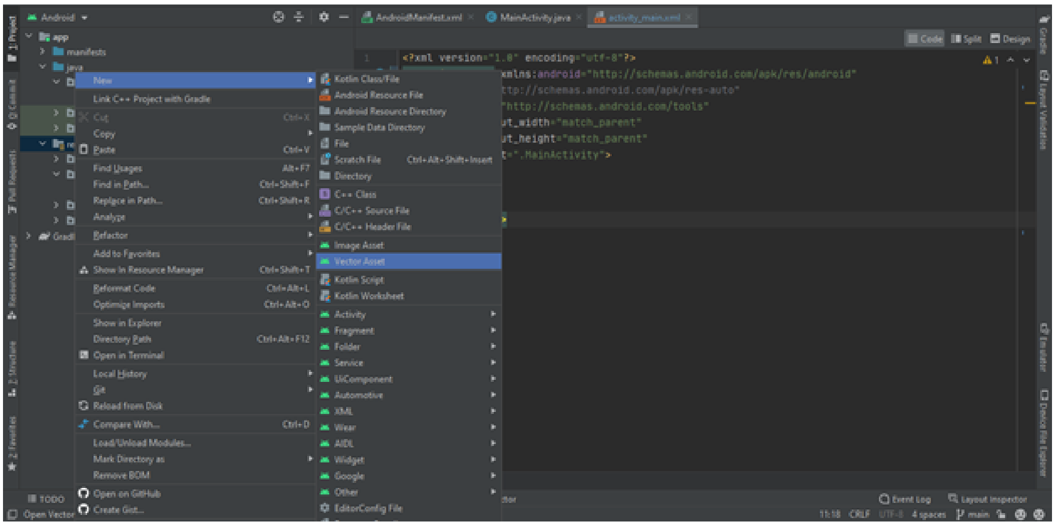
1. As you did in the first milestone, clone the Lab 3 Milestone 2 repository by selecting File > New > Project from Version Control > Git
2. Then click the following link and accept the invitation: https://classroom.github.com/a/uT9EyqZ8 (https://classroom.github.com/a/uT9EyqZ8)
3. You'll get a link in the form https://github.com/CS407-Fall-2021/lab3-milestone2-yourgithubnamehere
4. Copy this link in the URL text field in 'Clone' in Android Studio and click 'Clone'.
5. Change the "Constraint Layout" to "Relative Layout" (Line No. 2 & 11 as shown in the picture below):



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity">
8
9
10
11 </RelativeLayout>
```

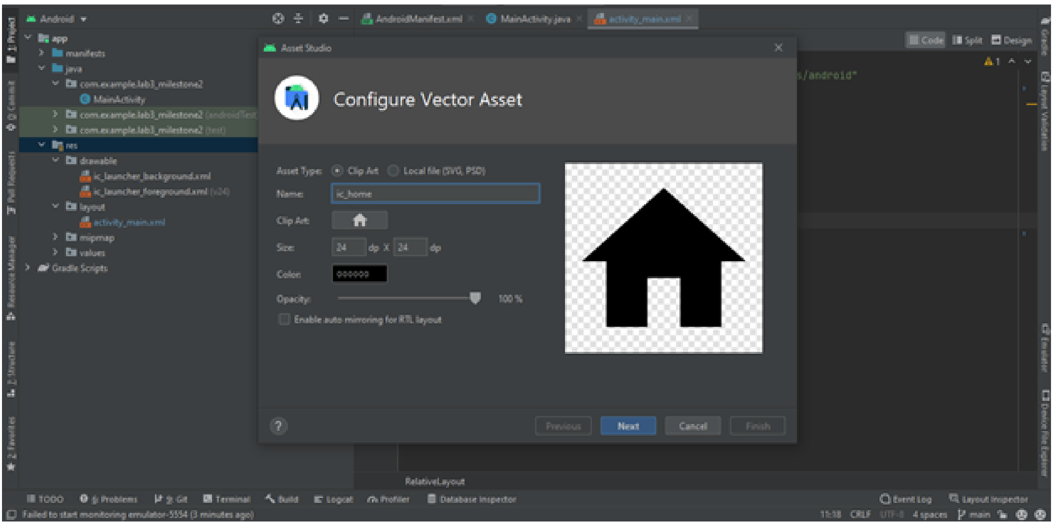
6. Now we will see how to create menus for your app.

7. First, we will add a vector asset to the app. Right-click res -> New -> Vector Asset.



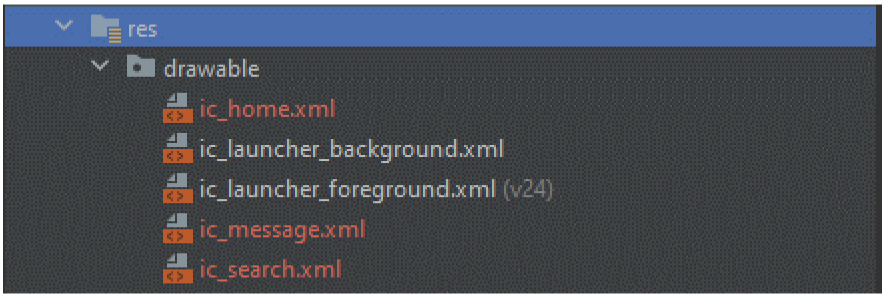
```
<?xml version="1.0" encoding="utf-8"?>
xmlns:android="http://schemas.android.com/apk/res/android"
ttp://schemas.android.com/apk/res-auto"
"http://schemas.android.com/tools"
ut_width="match_parent"
ut_height="match_parent"
t=".MainActivity">
```

8. Select ‘Clip Art’ and choose “Home” icon. Change the name to ‘ic_home’ and click next and then Finish.



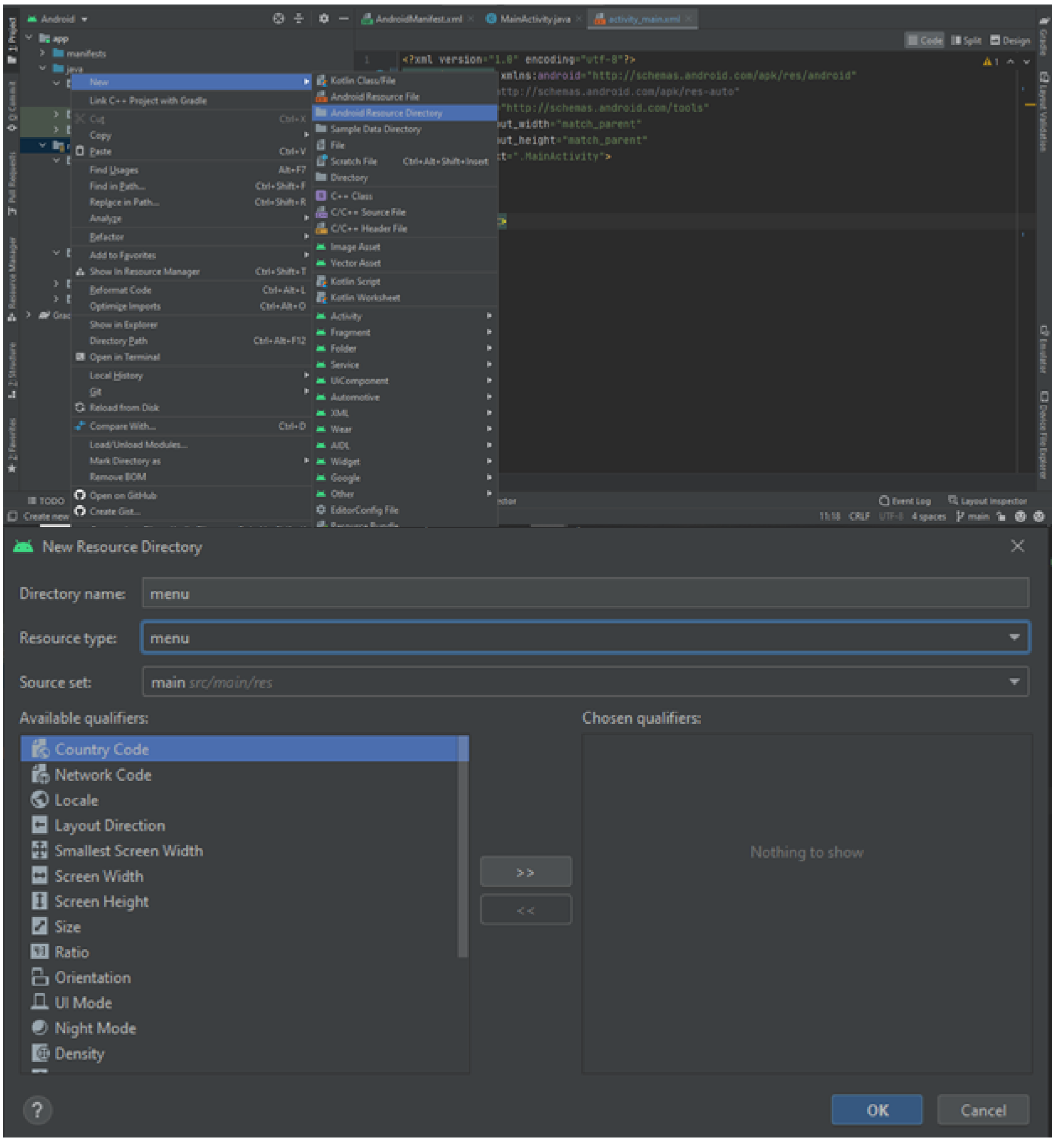
Asset Type: ● Clip Art ○ Local file (SVG, PSD)
Name: ic_home
Size: 24 dp X 24 dp
Color: 000000
Opacity: 100 %
□ Enable auto mirroring for RTL layout

9. Now repeat the same process for “Message” and “Search” icons and name them “ic_message” and “ic_search” respectively. After these steps, your “res” folder should look like this:



res
drawable
ic_home.xml
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
ic_message.xml
ic_search.xml

10. Now, we will create an Android Resource Directory for menus. Right Click res -> New -> Android Resource Directory. Under the ‘Resource type’ select the menu and click “OK”.

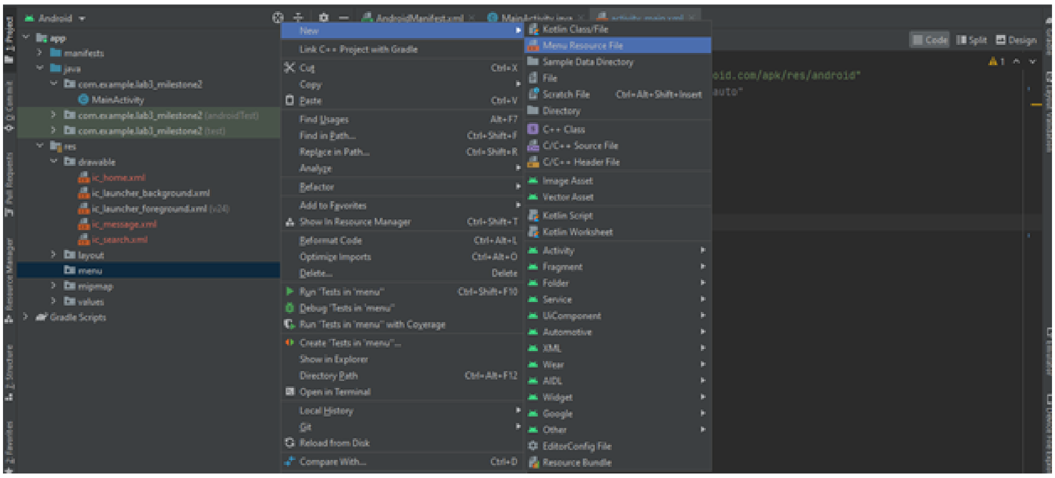


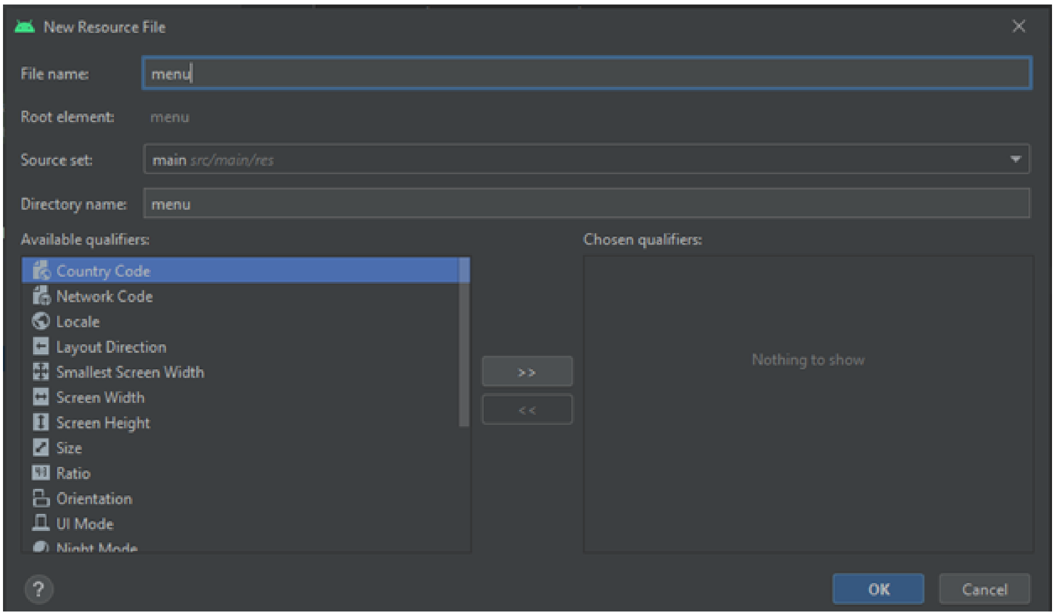
Directory name: menu

Resource type: menu

Source set: main src/main/res

11. Now right-click on ‘menu’ -> New -> Menu Resource File. Name it “menu” and then click “OK”.





New Resource File

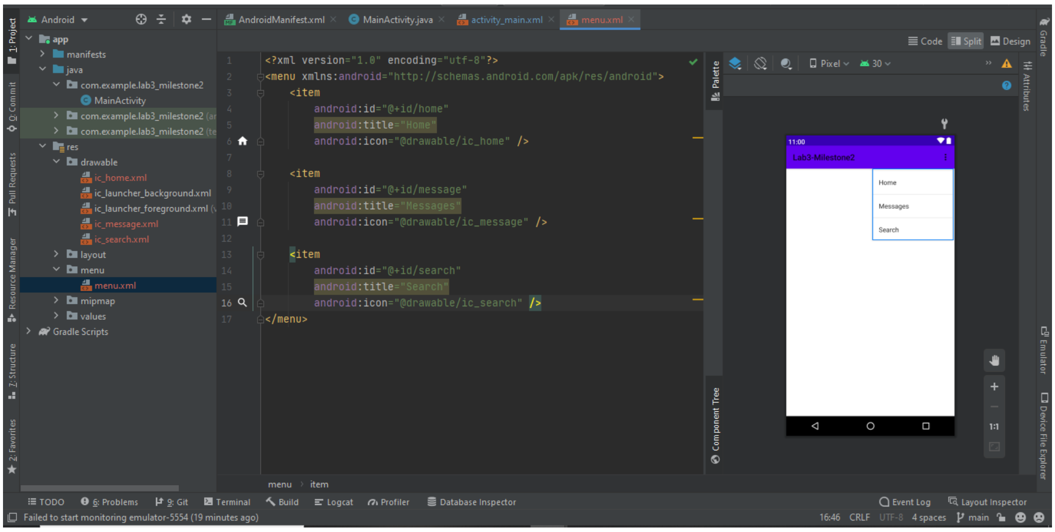
File name: menu

Root element: menu

Source set: main src/main/res

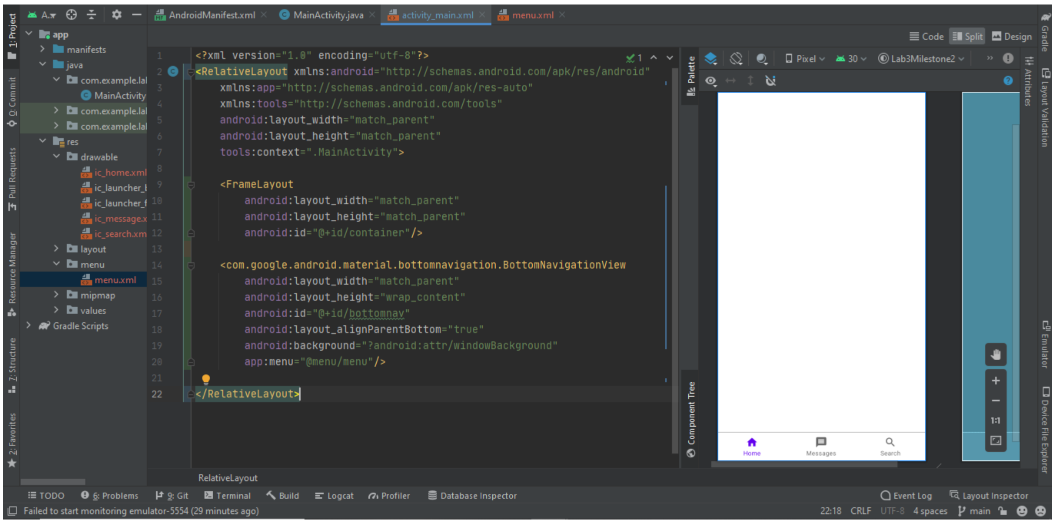
Directory name: menu

12. Now in the menu.xml file we just created, we will add items to our menu. We can assign the icons we created in steps 8 & 9 to menu items. Check out the code below:



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/home"
        android:title="Home"
        android:icon="@drawable/ic_home" />

    <item
        android:id="@+id/message"
        android:title="Messages"
        android:icon="@drawable/ic_message" />

    <item
        android:id="@+id/search"
        android:title="Search"
        android:icon="@drawable/ic_search" />
</menu>
```

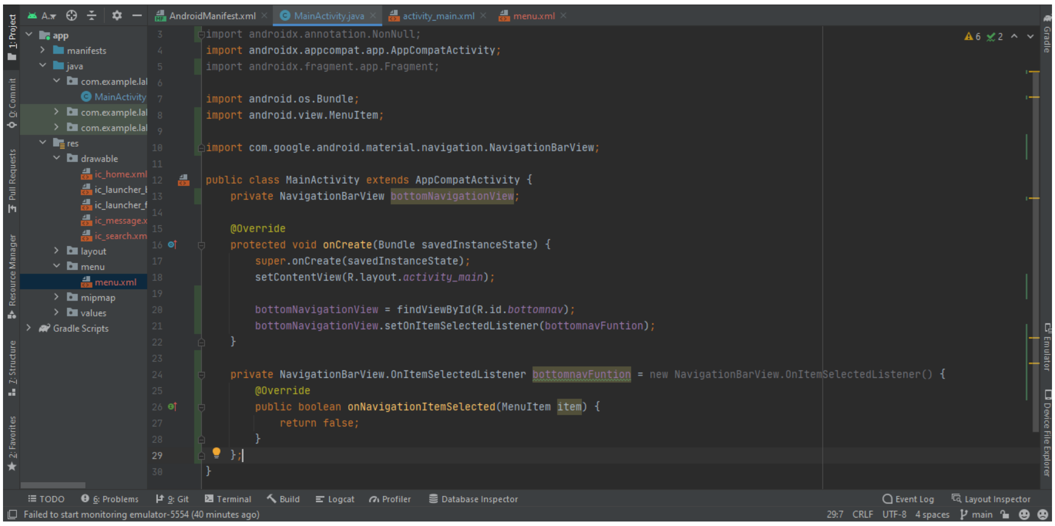
13. Now, let's add FrameLayout and convert our menu into a bottom navigation bar. Add this code in your "activity_main.xml" file:



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/container"/>

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/bottomnav"
        android:layout_alignParentBottom="true"
        android:background="?android:attr/windowBackground"
        app:menu="@menu/menu"/>

</RelativeLayout>
```

14. Now, we will go to MainActivity.java to set up the functionality of the bottom navigation bar. We'll define a function "bottomnavFunction" which will be called when an item is clicked on the bottom navigation bar. Inside this function, we'll override the method "onNavigationItemSelected". Your

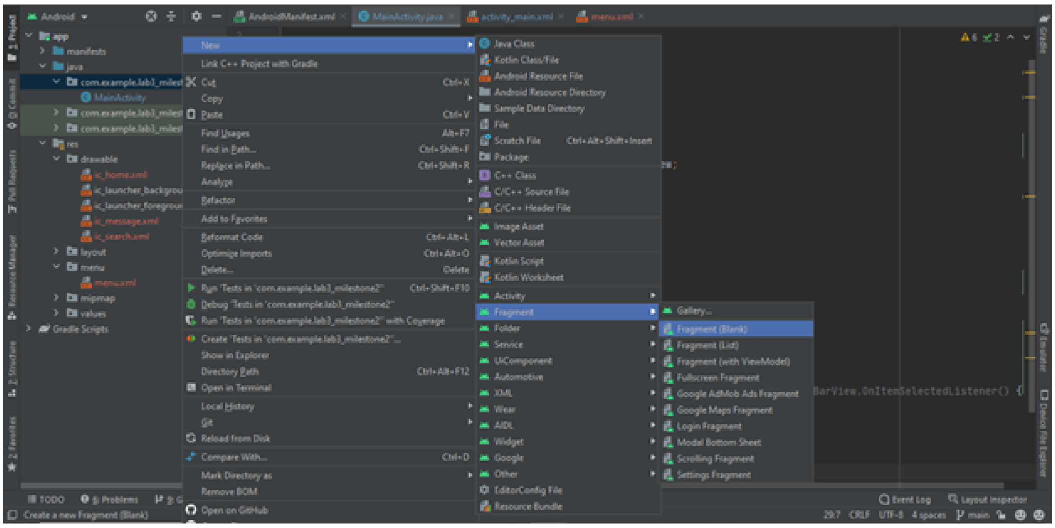
code should look like this. We’ll fill in the remaining code in step 24.



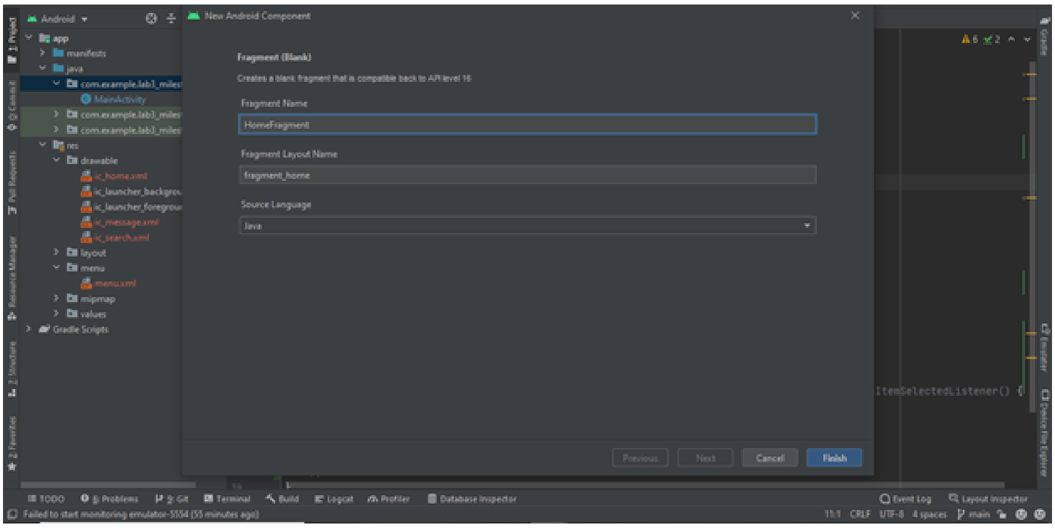
```java
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;

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

import com.google.android.material.navigation.NavigationBarView;

public class MainActivity extends AppCompatActivity {
    private NavigationBarView bottomNavigationView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        bottomNavigationView = findViewById(R.id.bottomnav);
        bottomNavigationView.setOnItemSelectedListener(bottomnavFuntion);
    }

    private NavigationBarView.OnItemSelectedListener bottomnavFuntion = new NavigationBarView.OnItemSelectedListener() {
        @Override
        public boolean onNavigationItemSelected(MenuItem item) {
            return false;
        }
    };
}
```

15. Now, we’ll create three fragments that will be displayed when an item is selected on the bottom navigation bar.

16. Create the first fragment: click on app package -> New -> Fragment -> Fragment (Blank).



17. Name it “HomeFragment” and then click “Finish”.



New Android Component

Fragment (Blank)

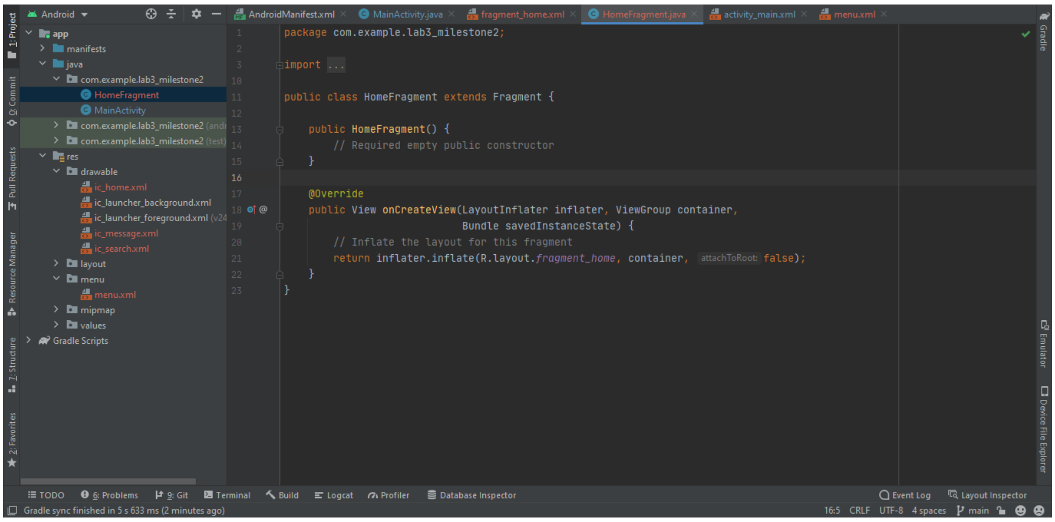
Creates a blank fragment that is compatible back to API level 16

Fragment Name: HomeFragment

Fragment Layout Name: fragment_home

Source Language: Java

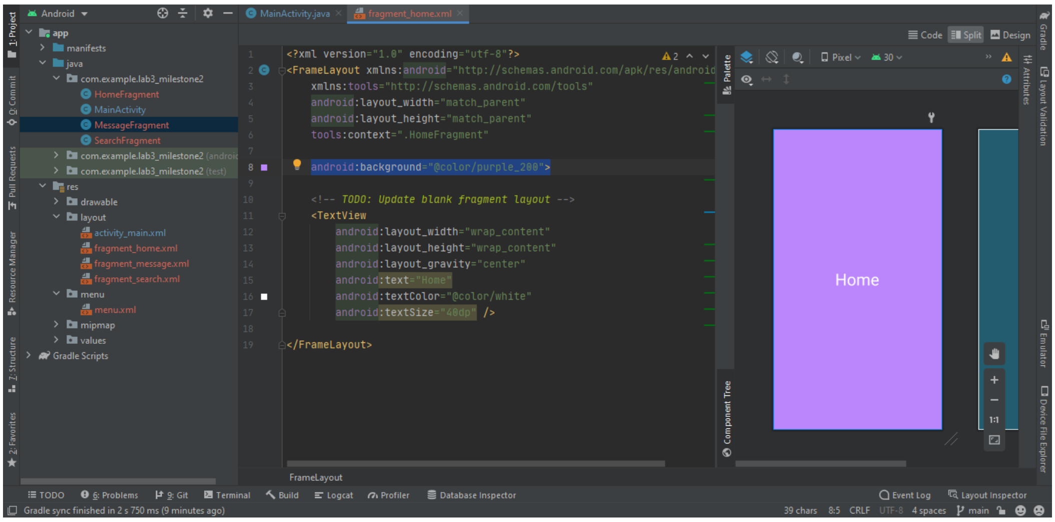
18. A new “HomeFragment.java” file will be created. From that file, remove all the extra code and just keep these two methods.



```java
package com.example.lab3_milestone2;

import ...

public class HomeFragment extends Fragment {

    public HomeFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_home, container, attachToRoot: false);
    }
}
```

Gradle sync finished in 5 s 633 ms (2 minutes ago)

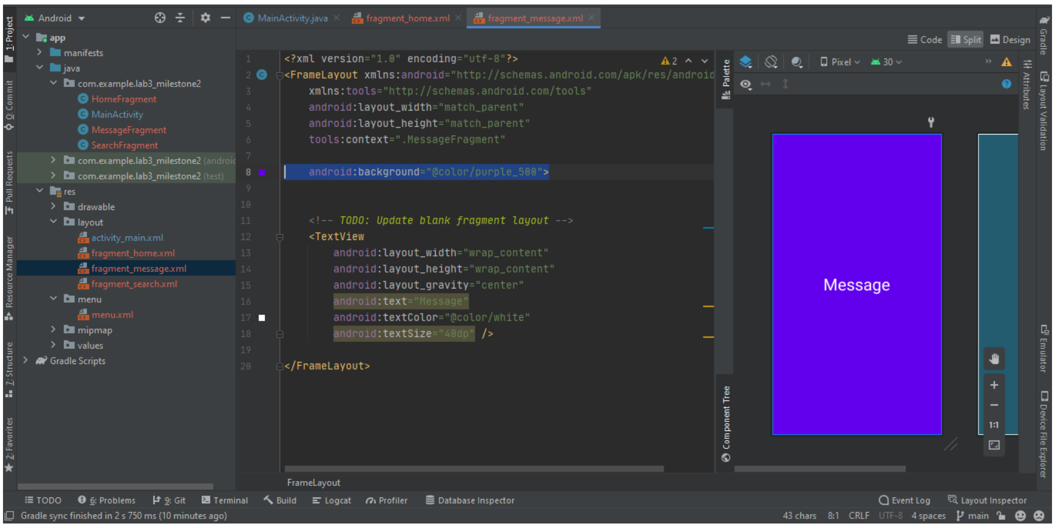
19. Now repeat steps 16, 17 & 18 twice to create two more Fragments named “MessageFragment” and “SearchFragment”.
20. Along with the Java files of Fragments, three XML files will also be created automatically, in the layout folder, for the corresponding Fragments. We’ll edit those XML files to set up the UI of the Fragments. We’ll just change the background colors and add some text in the center of the screen for every fragment.
21. For “fragment_home.xml”, your code should look like this:



```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".HomeFragment"

    android:background="@color/purple_200">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Home"
        android:textColor="@color/white"
        android:textSize="40dp" />

</FrameLayout>
```

Gradle sync finished in 2 s 750 ms (9 minutes ago)

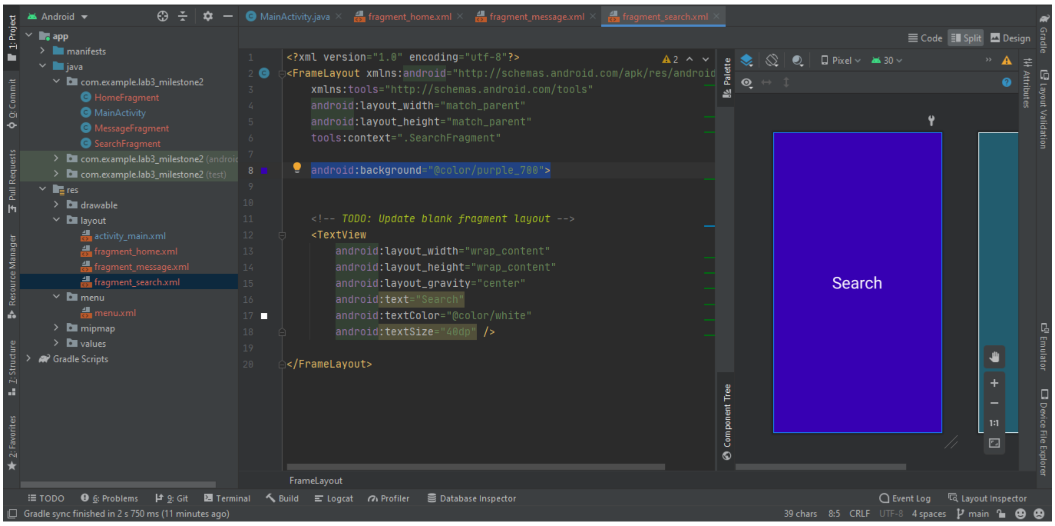
22. For “fragment_message.xml”, your code should look like this:



```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MessageFragment"

    android:background="@color/purple_500">


    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Message"
        android:textColor="@color/white"
        android:textSize="40dp" />

</FrameLayout>
```

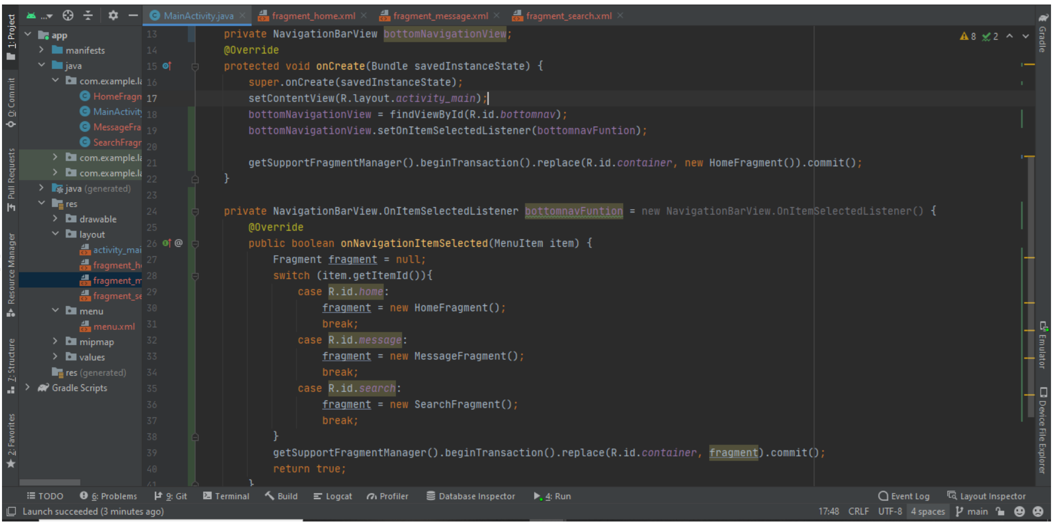
Gradle sync finished in 2 s 750 ms (10 minutes ago)

23. For “fragment_search.xml”, your code should look like this:



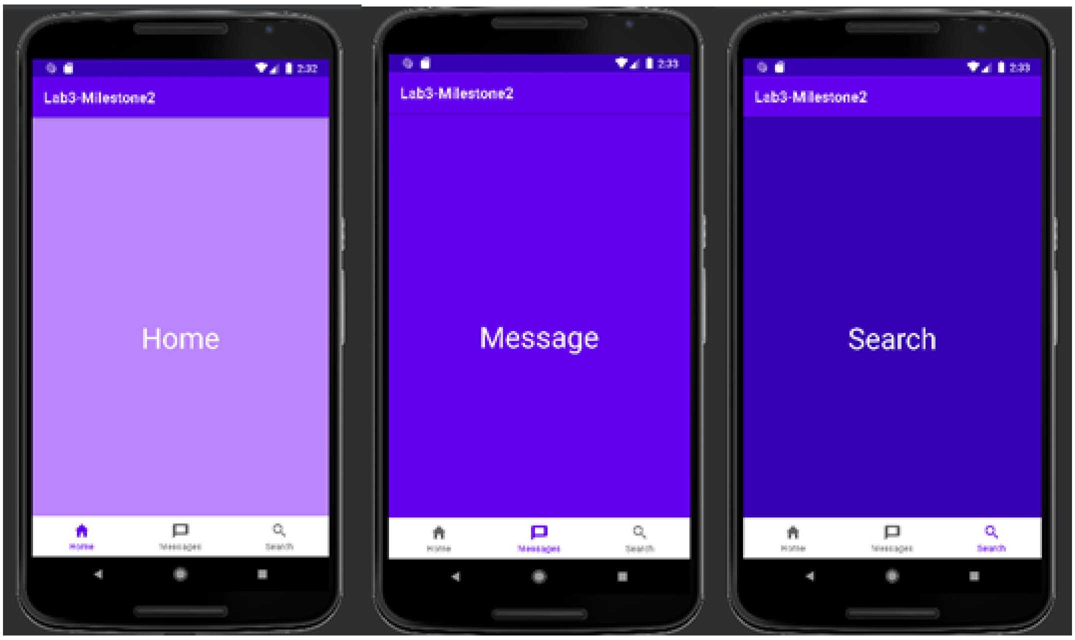
```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SearchFragment"

    android:background="@color/purple_700">


    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Search"
        android:textColor="@color/white"
        android:textSize="40dp" />

</FrameLayout>
```

24. After we are done creating the Fragments, we’ll go to MainActivity.java to fill the code in the bottomnavFunction we defined earlier in step 14. In this function we will create a switch() to see which menu item was pressed and once we get that information, we replace the current Layout with the Fragment corresponding to the selected item using Fragment Manager, a class responsible for performing actions on your app's fragments, such as adding, removing, or replacing them, and adding them to the back stack. Don’t forget to add line No. 21, this is to show the HomeFragment instead of the main activity when the app starts. Your code should look like this:



```java
13      private NavigationBarView bottomNavigationView;
14      @Override
15      protected void onCreate(Bundle savedInstanceState) {
16          super.onCreate(savedInstanceState);
17          setContentView(R.layout.activity_main);
18          bottomNavigationView = findViewById(R.id.bottomnav);
19          bottomNavigationView.setOnItemSelectedListener(bottomnavFuntion);
20
21          getSupportFragmentManager().beginTransaction().replace(R.id.container, new HomeFragment()).commit();
22      }
23
24      private NavigationBarView.OnItemSelectedListener bottomnavFuntion = new NavigationBarView.OnItemSelectedListener() {
25          @Override
26          public boolean onNavigationItemSelected(MenuItem item) {
27              Fragment fragment = null;
28              switch (item.getItemId()){
29                  case R.id.home:
30                      fragment = new HomeFragment();
31                      break;
32                  case R.id.message:
33                      fragment = new MessageFragment();
34                      break;
35                  case R.id.search:
36                      fragment = new SearchFragment();
37                      break;
38              }
39              getSupportFragmentManager().beginTransaction().replace(R.id.container, fragment).commit();
40              return true;
            }
```

25. Your Final app should look like this:

26. 

DELIVERABLES:

- Show the correct functioning of the Fragments to a peer mentor or TA
- The app should look presentable
- Switching between fragments should work correctly
- Commit and push the project to the GitHub classroom

Conclusion:

Great job on getting through this lab. There are a lot more things you can do using these components in Android Studio. Explore these components as much as you can because it will be a huge help for your projects.

References:

https://developer.android.com/guide/fragments (https://developer.android.com/guide/fragments)

Lab 3 Rubric

Criteria	Ratings		Pts
Milestone 1 - Clicking the "Open Dialog" opens the Dialog Box	**1.5 pts** **Full Marks**	**0 pts** **No Marks**	1.5 pts
Milestone 1 - Clicking the "OK" button on Dialog Box closes it	**1.5 pts** **Full Marks**	**0 pts** **No Marks**	1.5 pts
Milestone 1 - Commit and push changes to GitHub classroom repo	**1 pts** **Full Marks**	**0 pts** **No Marks**	1 pts
Milestone 2 - "Home" button shows the Home Fragment	**1 pts** **Full Marks**	**0 pts** **No Marks**	1 pts
Milestone 2 - "Message" button shows the Message Fragment	**1 pts** **Full Marks**	**0 pts** **No Marks**	1 pts
Milestone 2 - "Search" button shows the Search Fragment	**1 pts** **Full Marks**	**0 pts** **No Marks**	1 pts
Milestone 2 - Navigation Bar in a presentable format	**2 pts** **Full Marks**	**0 pts** **No Marks**	2 pts
Milestone 2 - Commit and push changes to GitHub classroom repo	**1 pts** **Full Marks**	**0 pts** **No Marks**	1 pts
			Total Points: 10