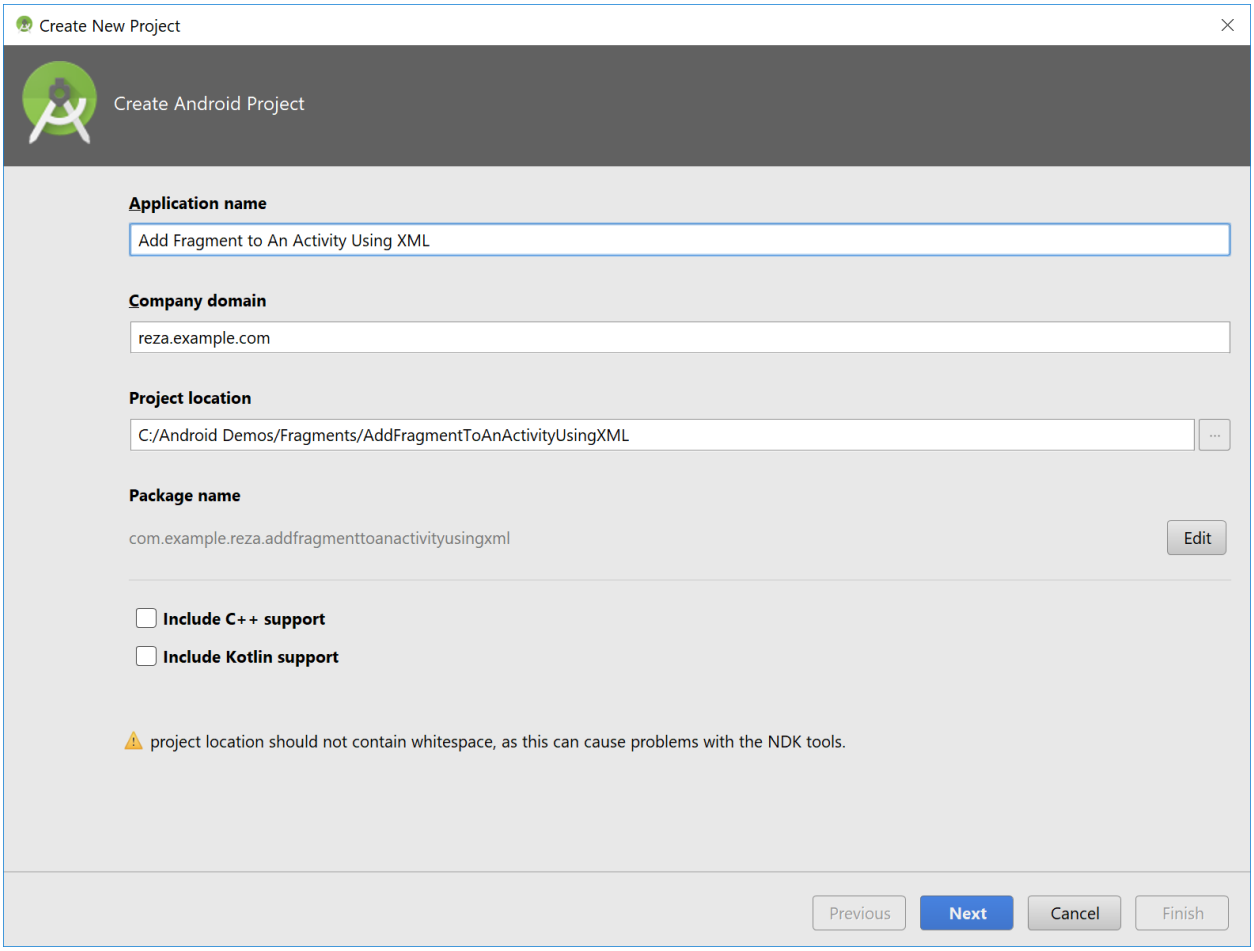
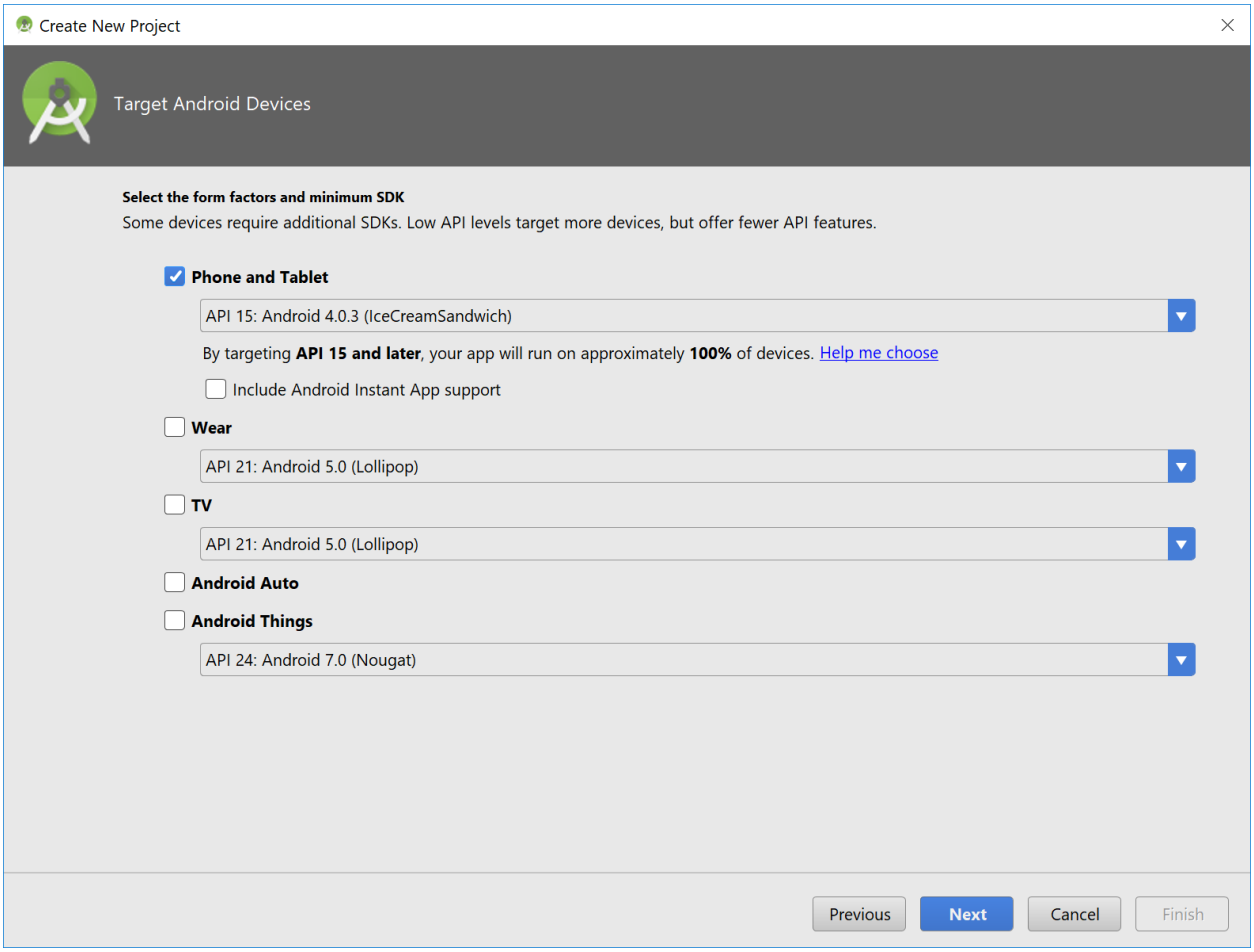
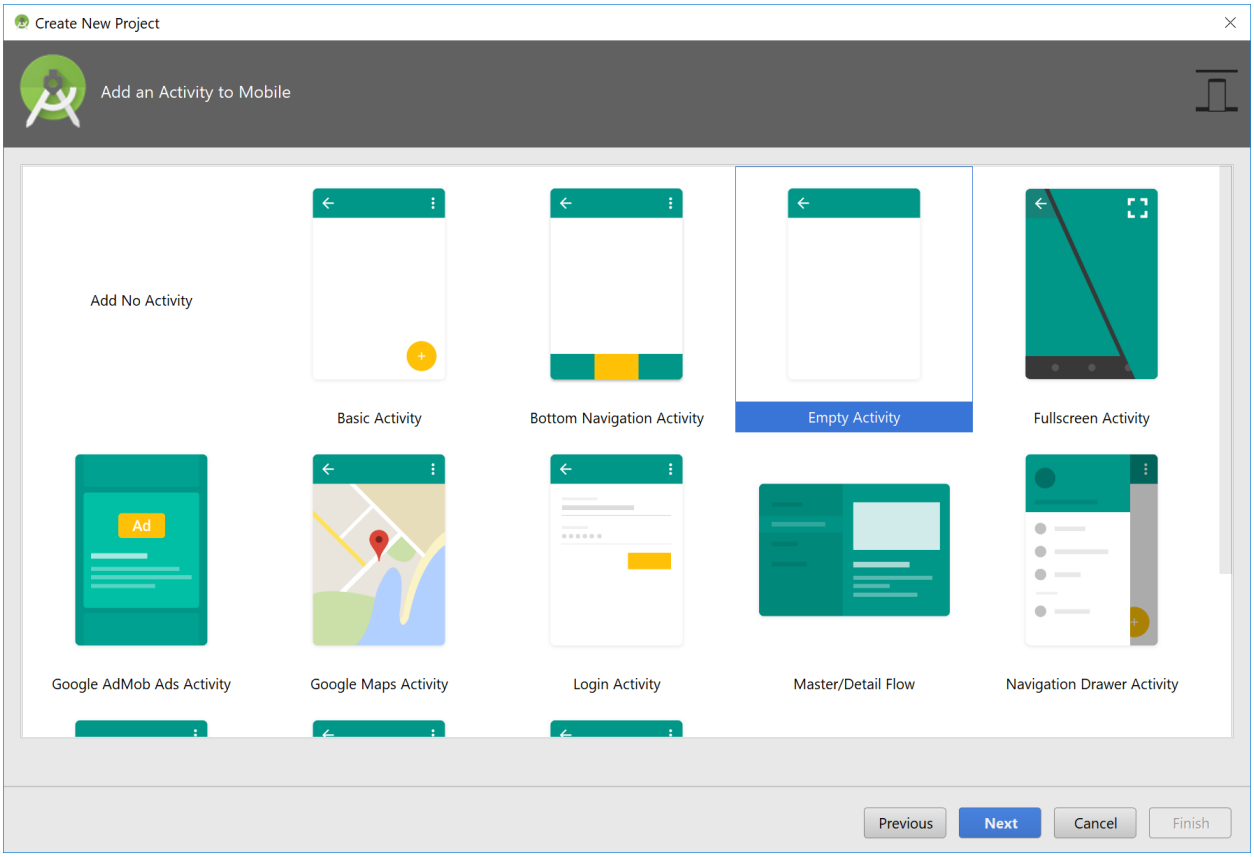
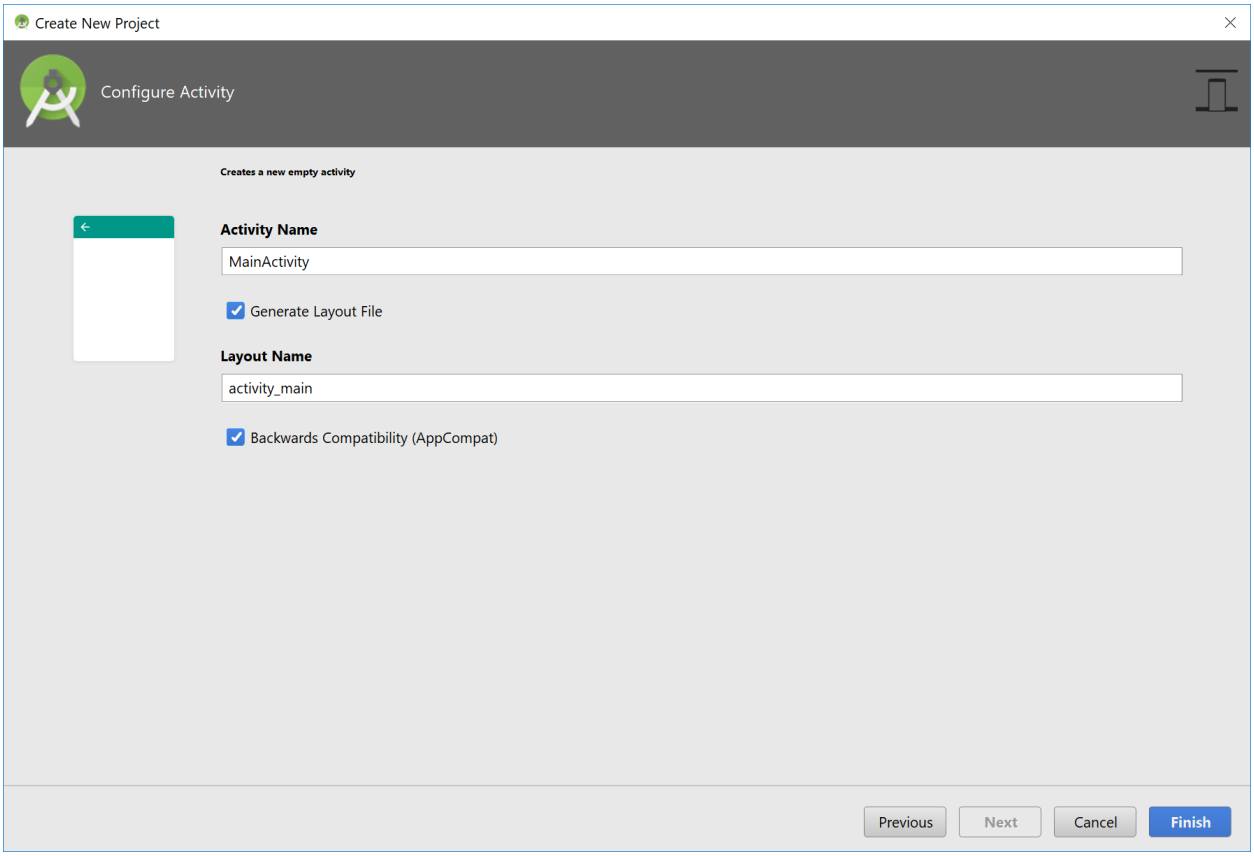
Adding a Fragment to an Activity using XML (dragging it into layout)

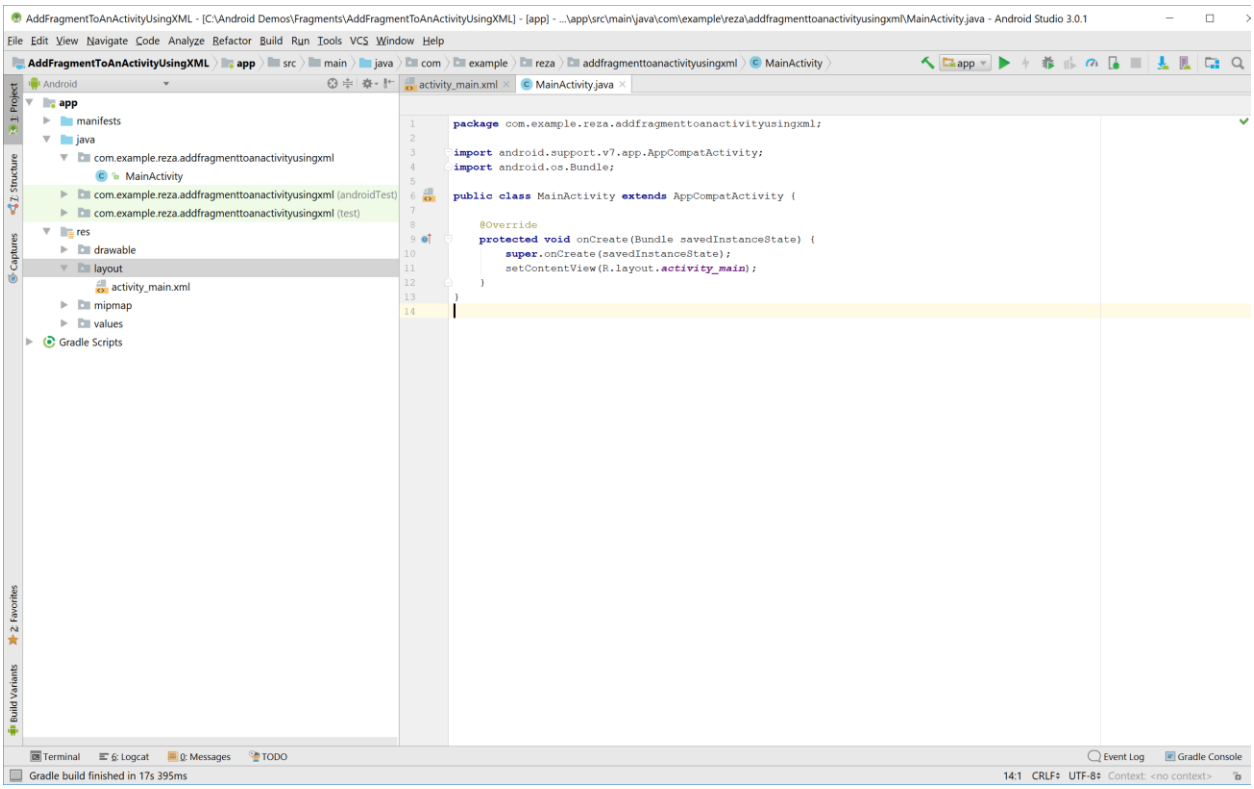
Create a new project as shown:

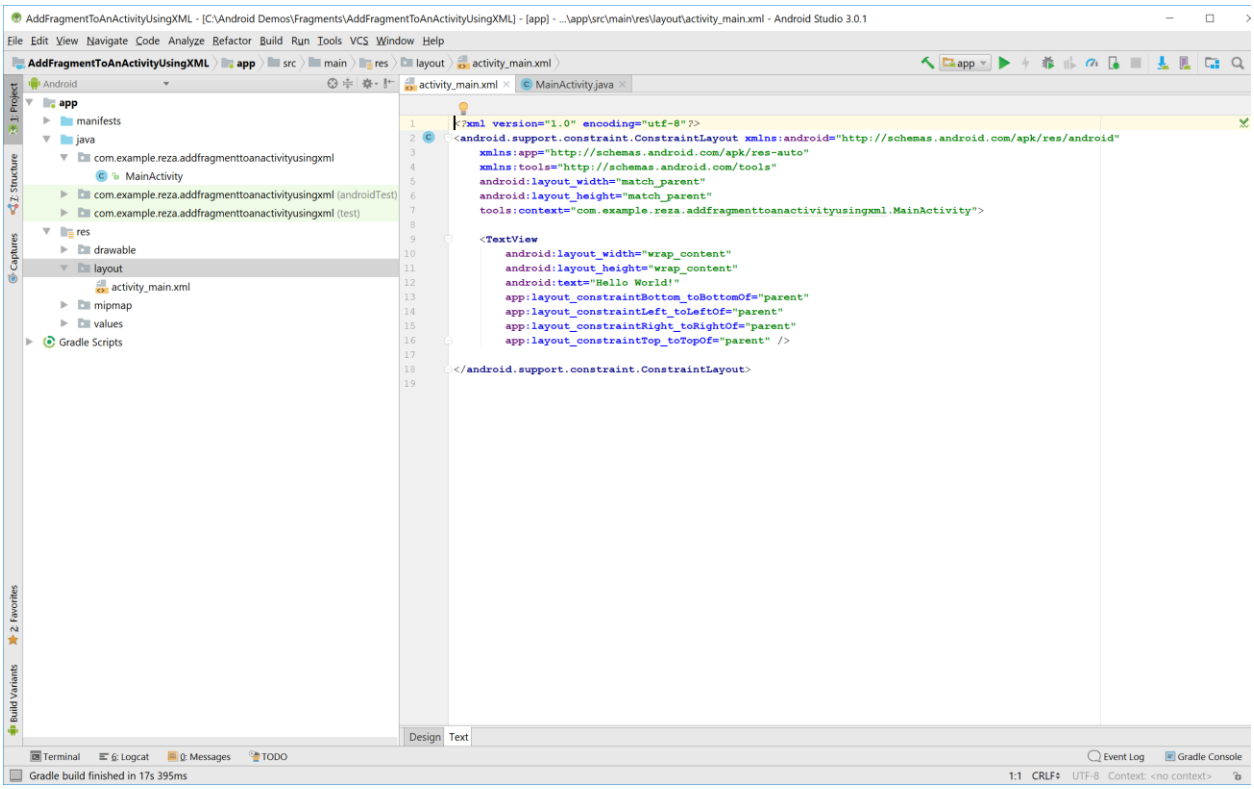



Create New Project
Target Android Devices
Select the form factors and minimum SDK
Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.
Phone and Tablet
API 15: Android 4.0.3 (IceCreamSandwich)
By targeting API 15 and later, your app will run on approximately 100% of devices. Help me choose
Include Android Instant App support
Wear
API 21: Android 5.0 (Lollipop)
TV
API 21: Android 5.0 (Lollipop)
Android Auto
Android Things
API 24: Android 7.0 (Nougat)
Previous
Next
Cancel
Finish

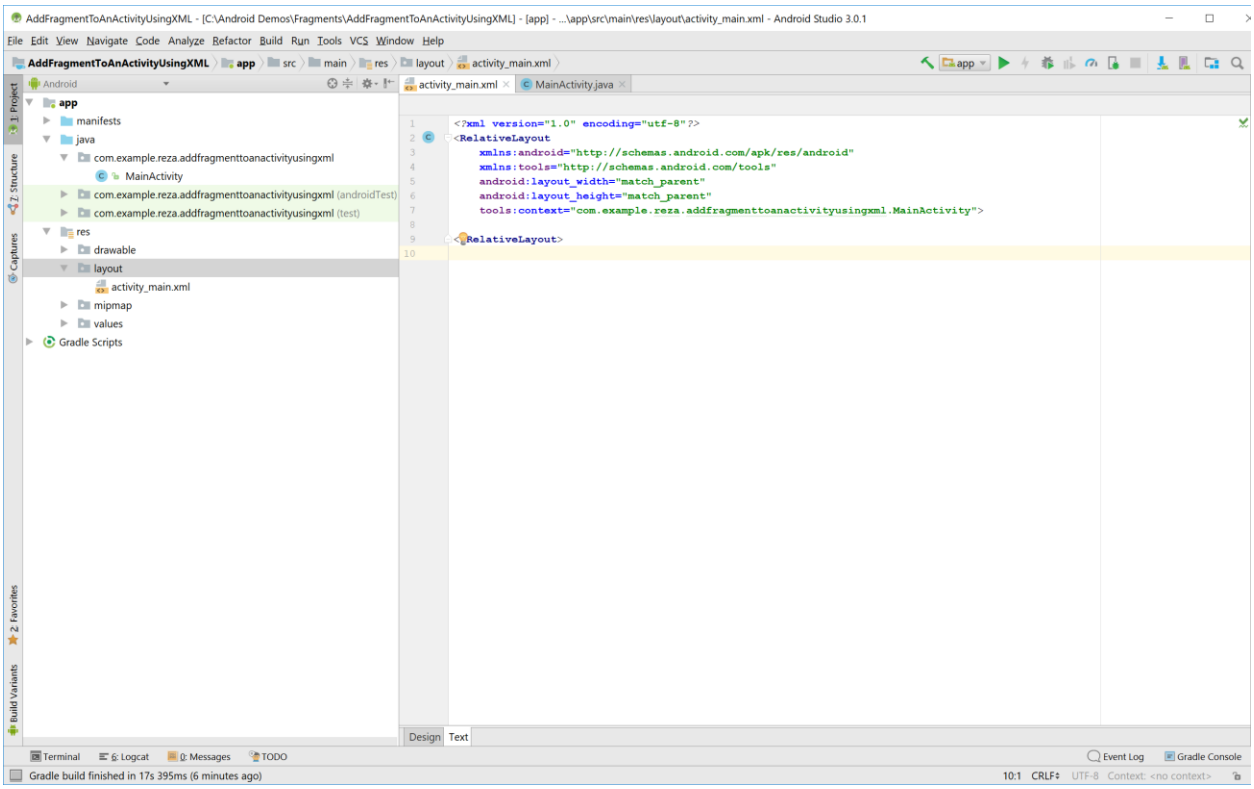

Create New Project
Add an Activity to Mobile
Add No Activity
Basic Activity
Bottom Navigation Activity
Empty Activity
Fullscreen Activity
Ad
Google AdMob Ads Activity
Google Maps Activity
Login Activity
Master/Detail Flow
Navigation Drawer Activity
Previous
Next
Cancel
Finish


Create New Project
Configure Activity
Creates a new empty activity
Activity Name
MainActivity
Generate Layout File
Layout Name
activity_main
Backwards Compatibility (AppCompat)
Previous
Next
Cancel
Finish

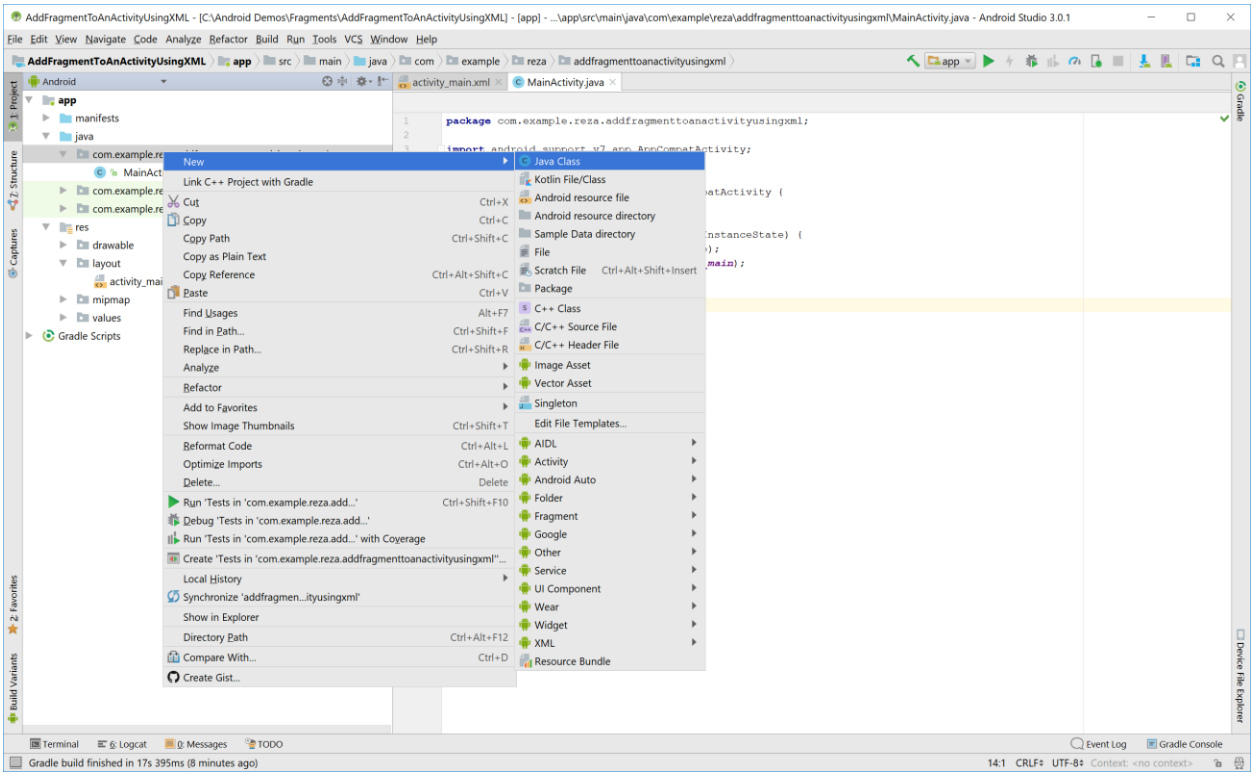

AddFragmentToAnActivityUsingXML - [C:\Android Demos\Fragments\AddFragmentToAnActivityUsingXML] - [app] - ...\app\src\main\java\com\example\reza\addfragmenttoanactivityusingxml\MainActivity.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AddFragmentToAnActivityUsingXML app src main java com example reza addfragmenttoanactivityusingxml MainActivity
activity_main.xml
MainActivity.java
app
manifests
java
com.example.reza.addfragmenttoanactivityusingxml
MainActivity
com.example.reza.addfragmenttoanactivityusingxml (androidTest)
com.example.reza.addfragmenttoanactivityusingxml (test)
res
drawable
layout
activity_main.xml
mipmap
values
Gradle Scripts
package com.example.reza.addfragmenttoanactivityusingxml;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
public class MainActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
}
Terminal
6: Logcat
0: Messages
TODO
Event Log
Gradle Console
Gradle build finished in 17s 395ms
14:1 CRLF UTF-8 Context: <no context>

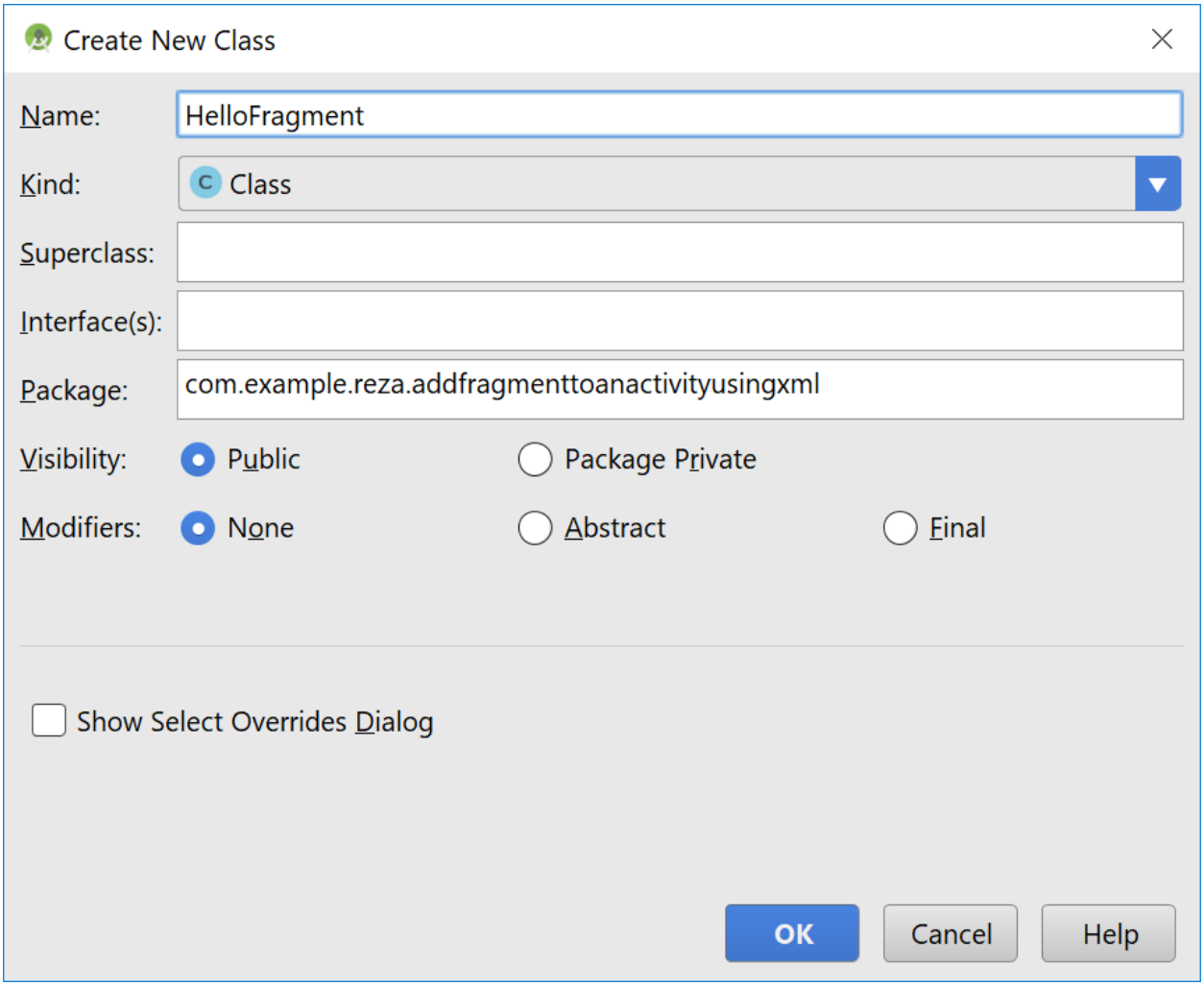

AddFragmentToAnActivityUsingXML - [C:\Android Demos\Fragments\AddFragmentToAnActivityUsingXML] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AddFragmentToAnActivityUsingXML app src main res layout activity_main.xml
activity_main.xml
MainActivity.java
app
manifests
java
com.example.reza.addfragmenttoanactivityusingxml
MainActivity
com.example.reza.addfragmenttoanactivityusingxml (androidTest)
com.example.reza.addfragmenttoanactivityusingxml (test)
res
drawable
layout
activity_main.xml
mipmap
values
Gradle Scripts
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout 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="com.example.reza.addfragmenttoanactivityusingxml.MainActivity">
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Hello World!"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintLeft_toLeftOf="parent"
app:layout_constraintRight_toRightOf="parent"
app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
Design
Text
Terminal
6: Logcat
0: Messages
TODO
Event Log
Gradle Console
Gradle build finished in 17s 395ms
1:1 CRLF UTF-8 Context: <no context>

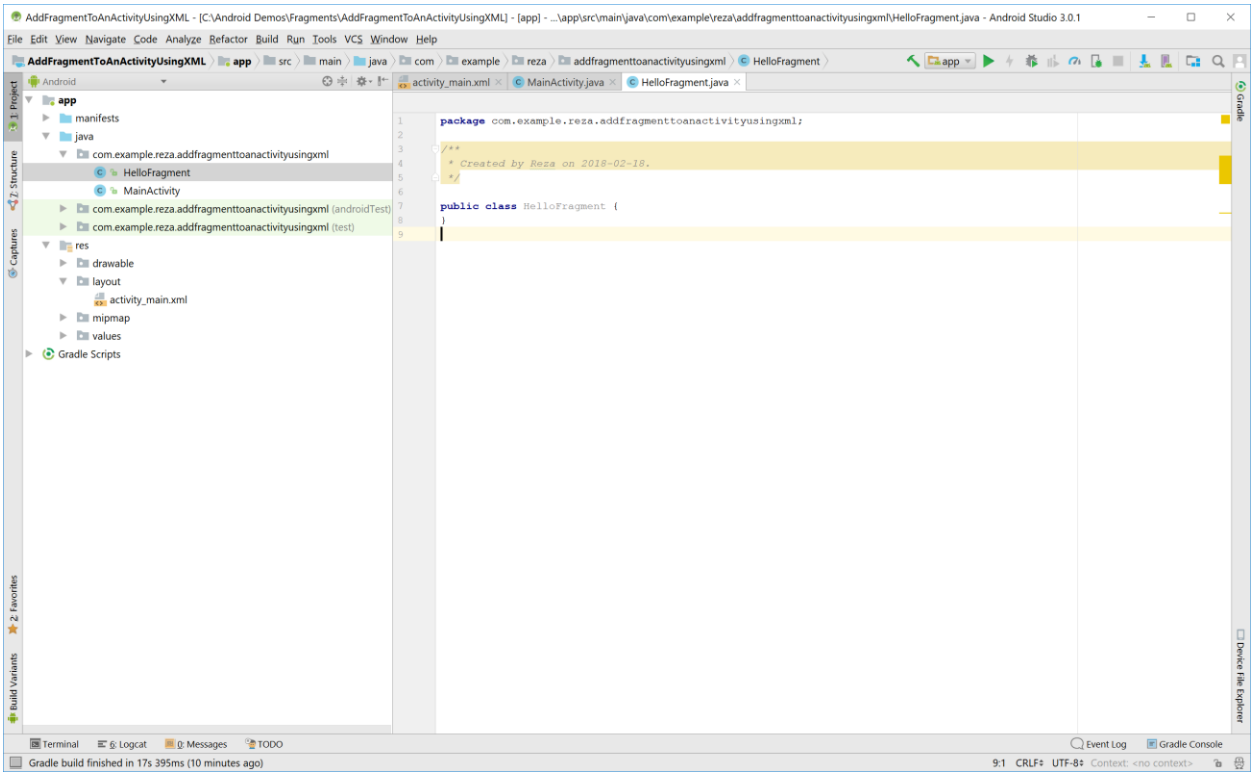
Change the layout file (activity_main.xml) as shown:

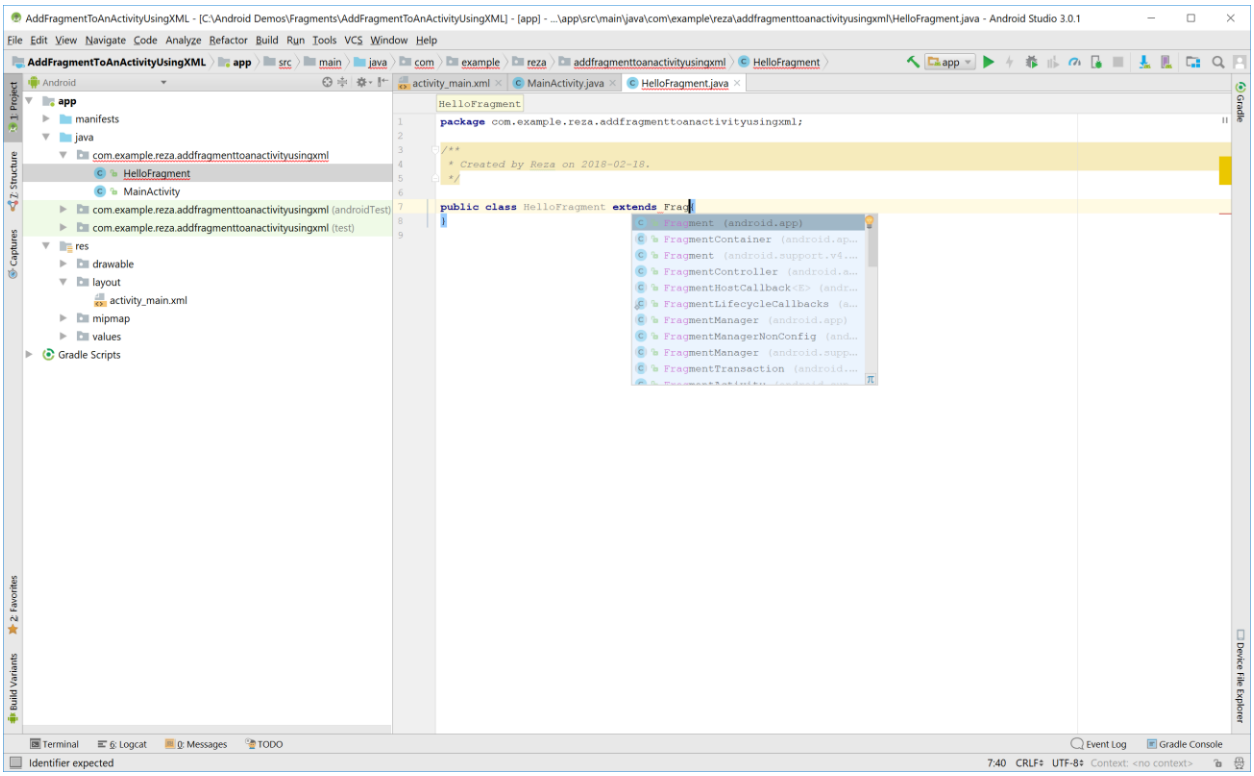


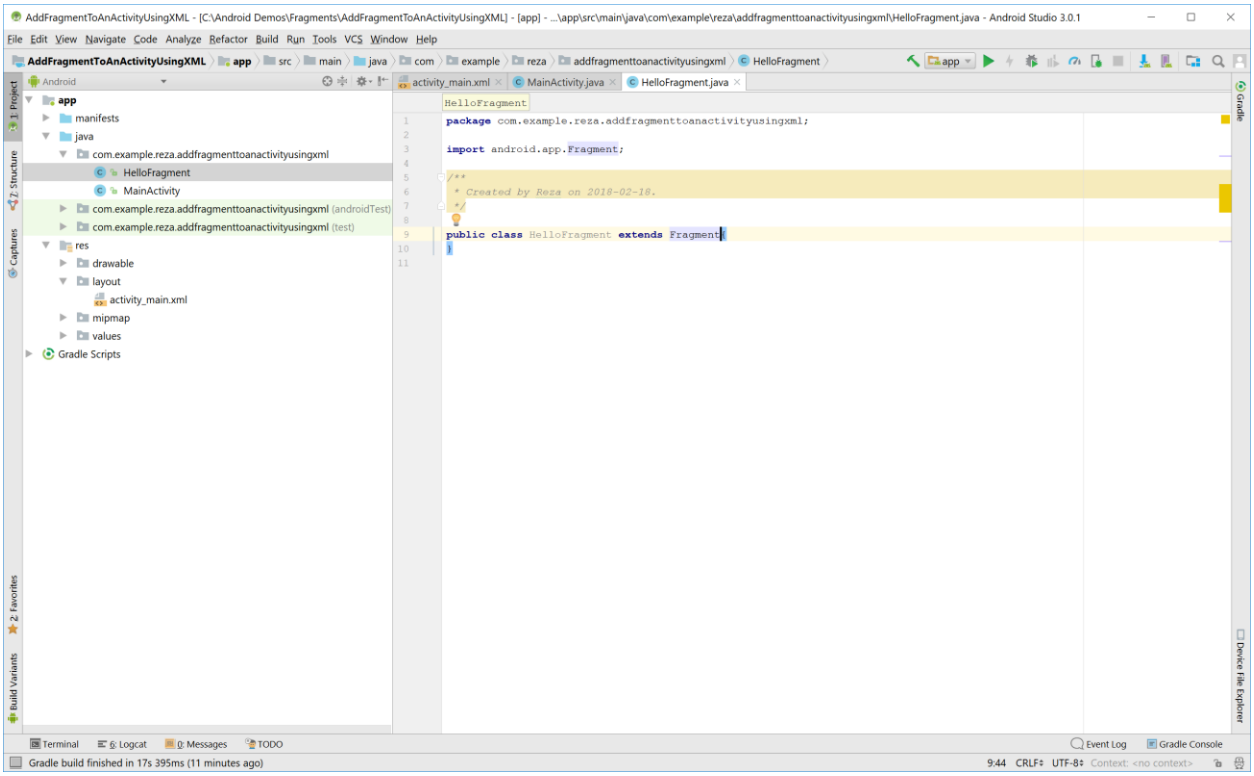
Now you need to create your fragment which is a subclass of Fragment class. So create a new Java class:



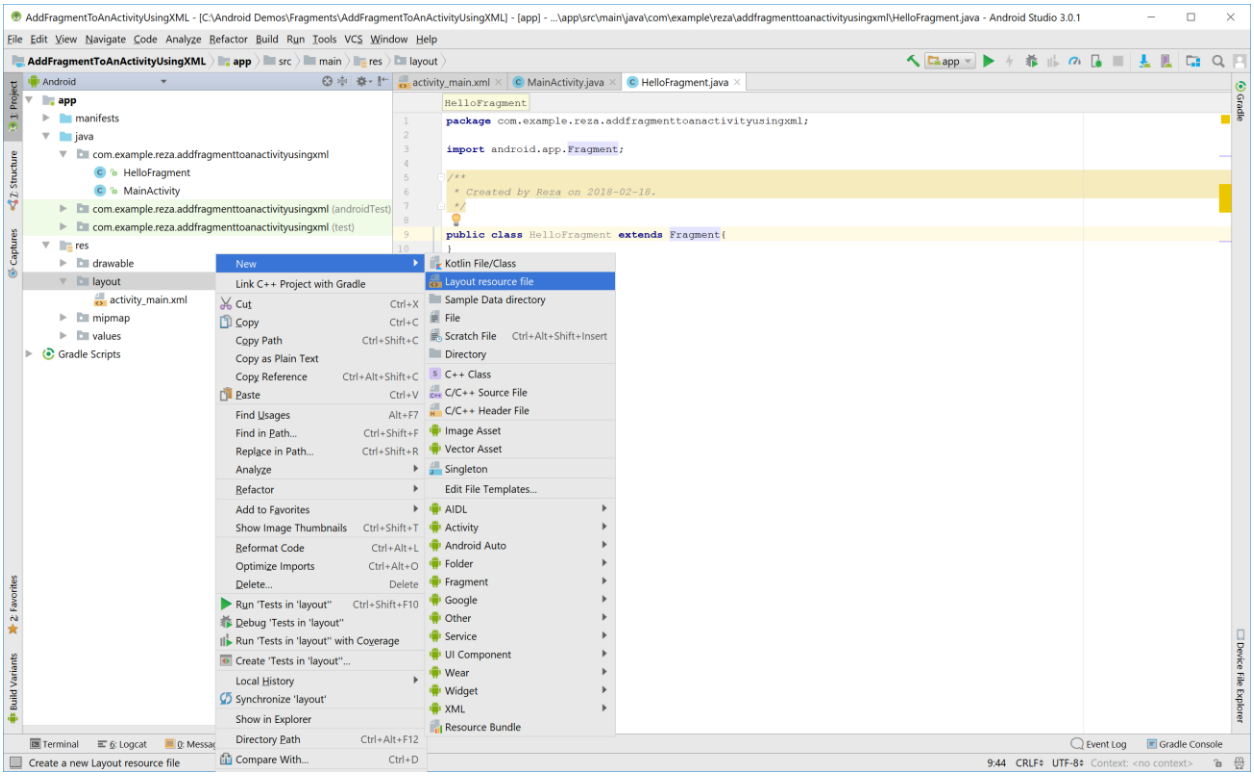

Create New Class
Name:
HelloFragment
Kind:
Class
Superclass:
Interface(s):
Package:
com.example.reza.addfragmenttoanactivityusingxml
Visibility:
Public
Package Private
Modifiers:
None
Abstract
Final
Show Select Overrides Dialog
OK
Cancel
Help

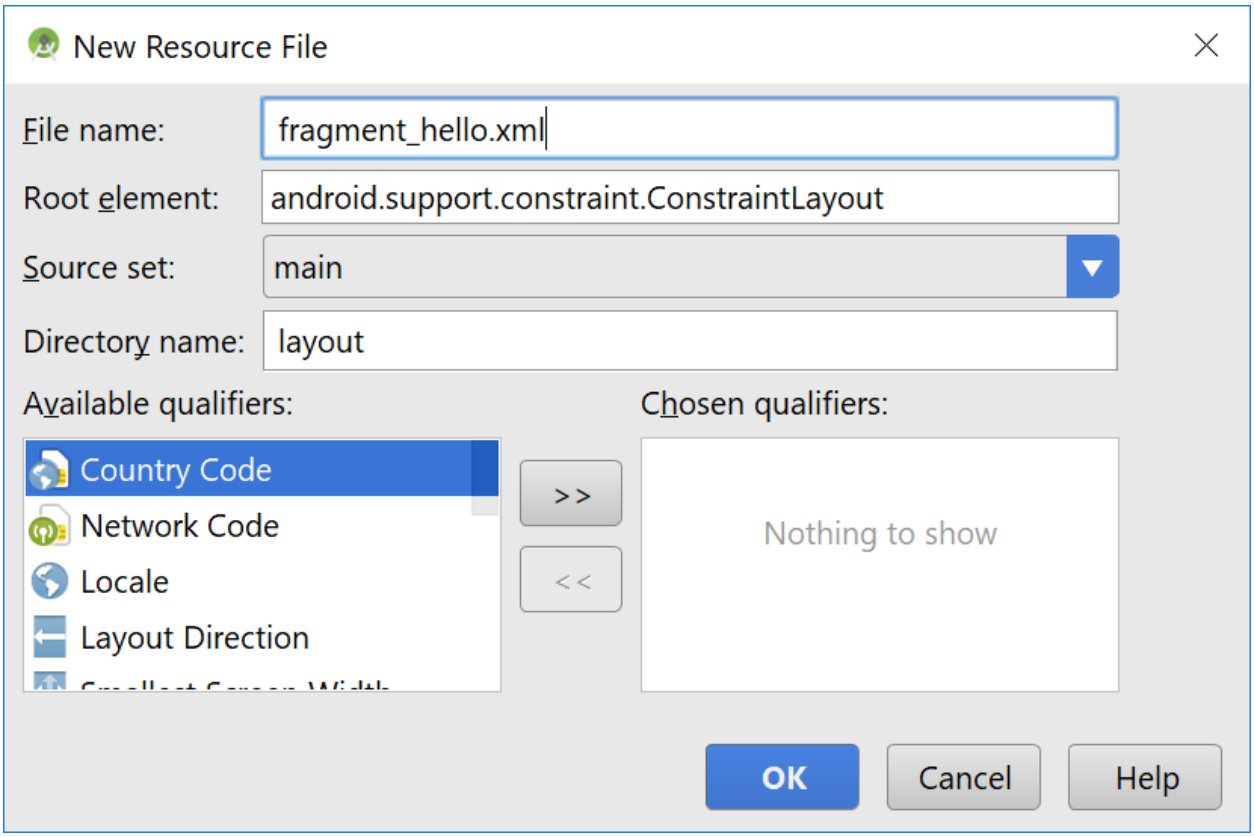

AddFragmentToAnActivityUsingXML - [C:\Android Demos\Fragments\AddFragmentToAnActivityUsingXML] - [app] - ...\app\src\main\java\com\example\reza\addfragmenttoanactivityusingxml\HelloFragment.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AddFragmentToAnActivityUsingXML app src main java com example reza addfragmenttoanactivityusingxml HelloFragment
activity_main.xml MainActivity.java HelloFragment.java
app
manifests
java
com.example.reza.addfragmenttoanactivityusingxml
HelloFragment
MainActivity
com.example.reza.addfragmenttoanactivityusingxml (androidTest)
com.example.reza.addfragmenttoanactivityusingxml (test)
res
drawable
layout
activity_main.xml
mipmap
values
Gradle Scripts
package com.example.reza.addfragmenttoanactivityusingxml;
/**
* Created by Reza on 2018-02-18.
*/
public class HelloFragment {
}
Terminal 6: Logcat 0: Messages TODO
Event Log Gradle Console
Gradle build finished in 17s 395ms (10 minutes ago)
9:1 CRLF UTF-8 Context: <no context>


AddFragmentToAnActivityUsingXML - [C:\Android Demos\Fragments\AddFragmentToAnActivityUsingXML] - [app] - ...\app\src\main\java\com\example\reza\addfragmenttoanactivityusingxml\HelloFragment.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AddFragmentToAnActivityUsingXML app src main java com example reza addfragmenttoanactivityusingxml HelloFragment
activity_main.xml MainActivity.java HelloFragment.java
HelloFragment
app
manifests
java
com.example.reza.addfragmenttoanactivityusingxml
HelloFragment
MainActivity
com.example.reza.addfragmenttoanactivityusingxml (androidTest)
com.example.reza.addfragmenttoanactivityusingxml (test)
res
drawable
layout
activity_main.xml
mipmap
values
Gradle Scripts
package com.example.reza.addfragmenttoanactivityusingxml;
/**
* Created by Reza on 2018-02-18.
*/
public class HelloFragment extends Frag
}
Fragment (android.app)
FragmentContainer (android.ap...
Fragment (android.support.v4....
FragmentController (android.s...
FragmentHostCallback<E> (andr...
FragmentLifecycleCallbacks (a...
FragmentManager (android.app)
FragmentManagerNonConfig (and...
FragmentManager (android.supp...
FragmentTransaction (android....
Terminal 6: Logcat 0: Messages TODO
Event Log Gradle Console
Identifier expected
7:40 CRLF UTF-8 Context: <no context>

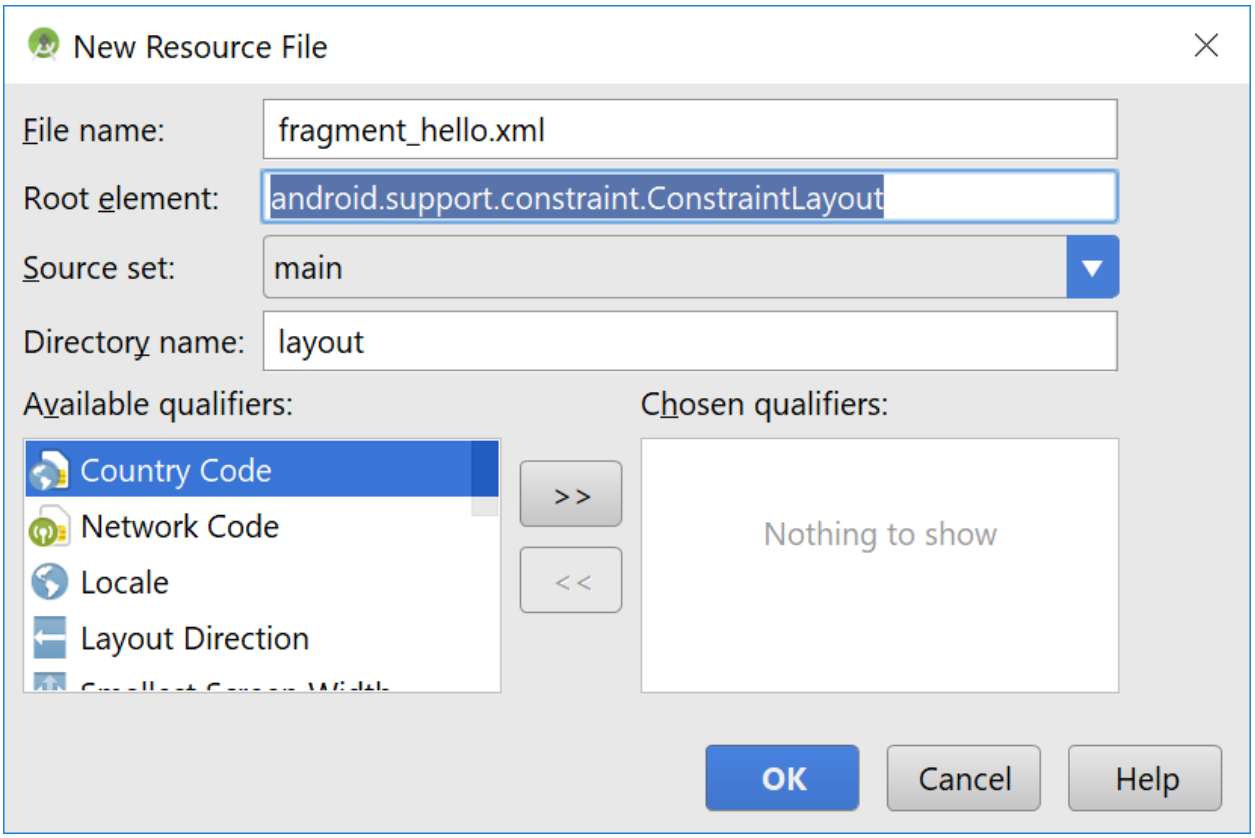


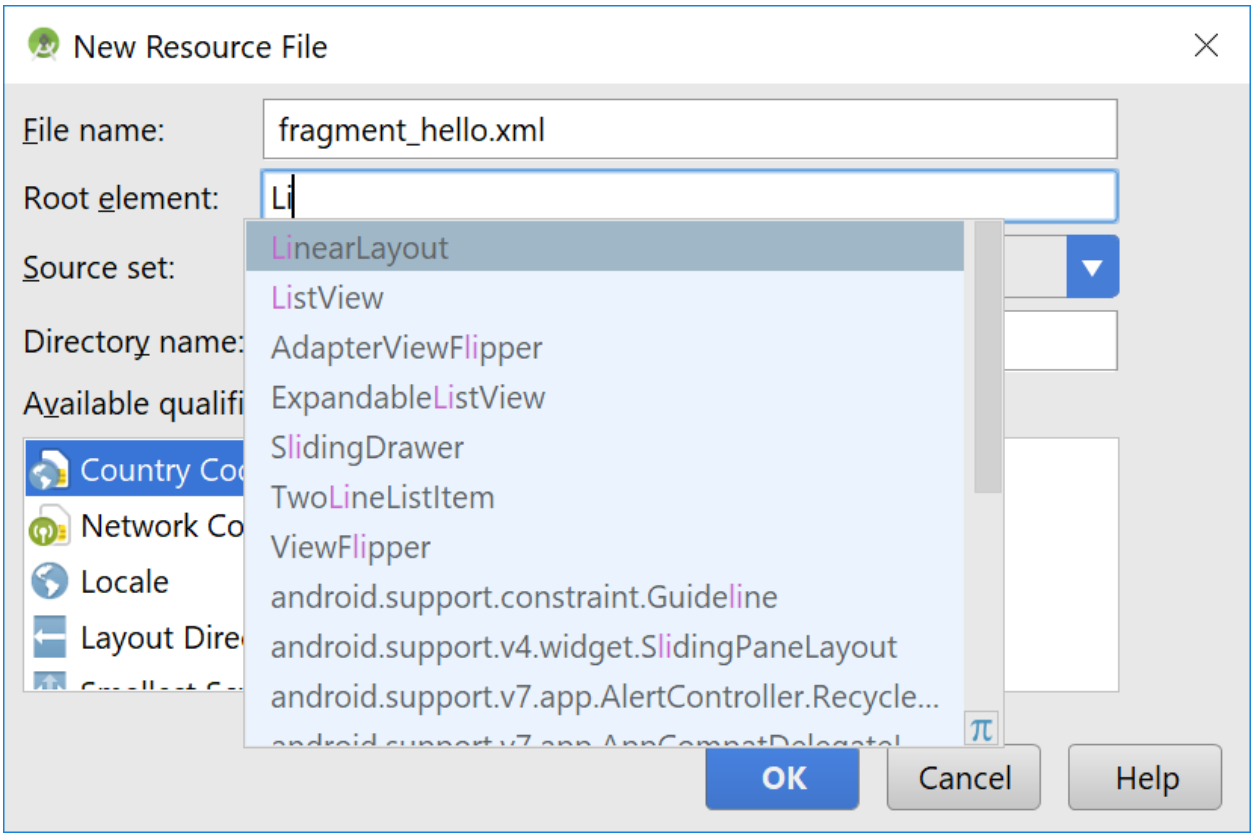
Next create a layout for your fragment class as shown:

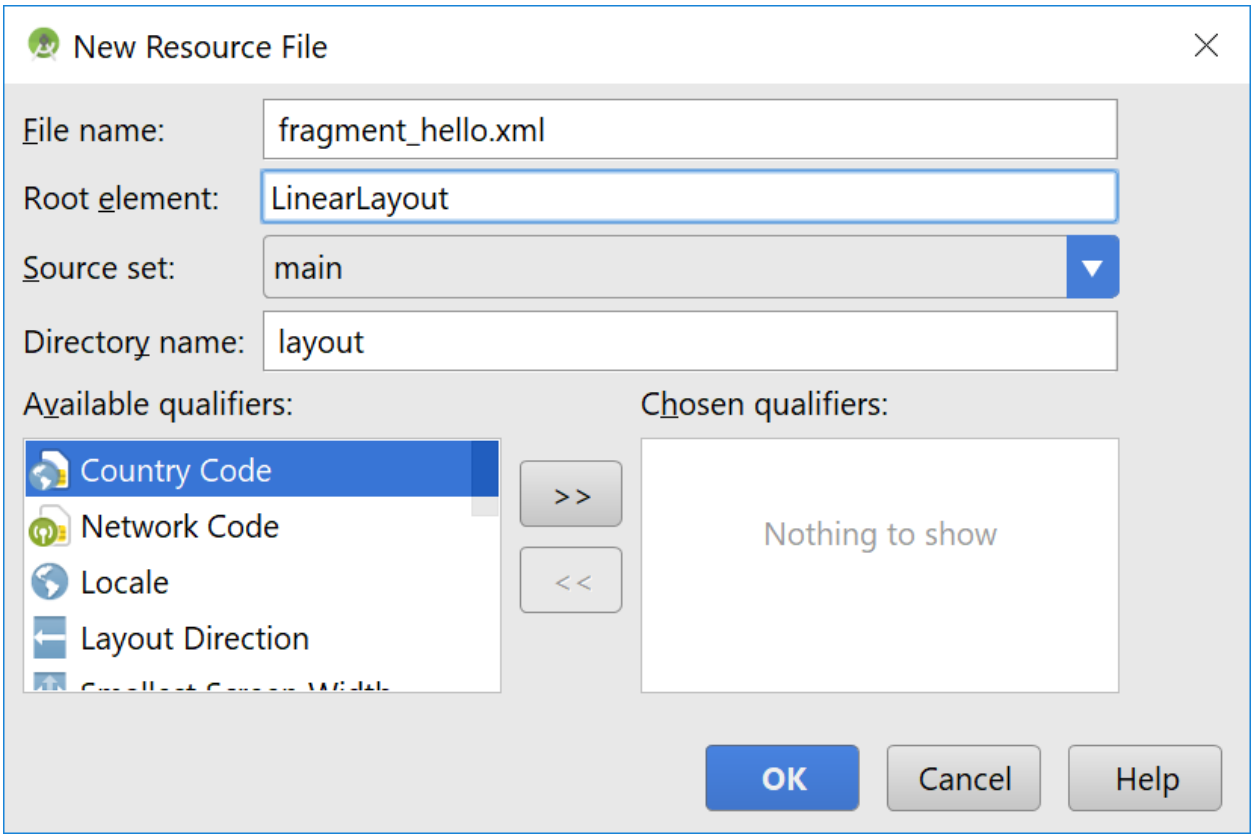


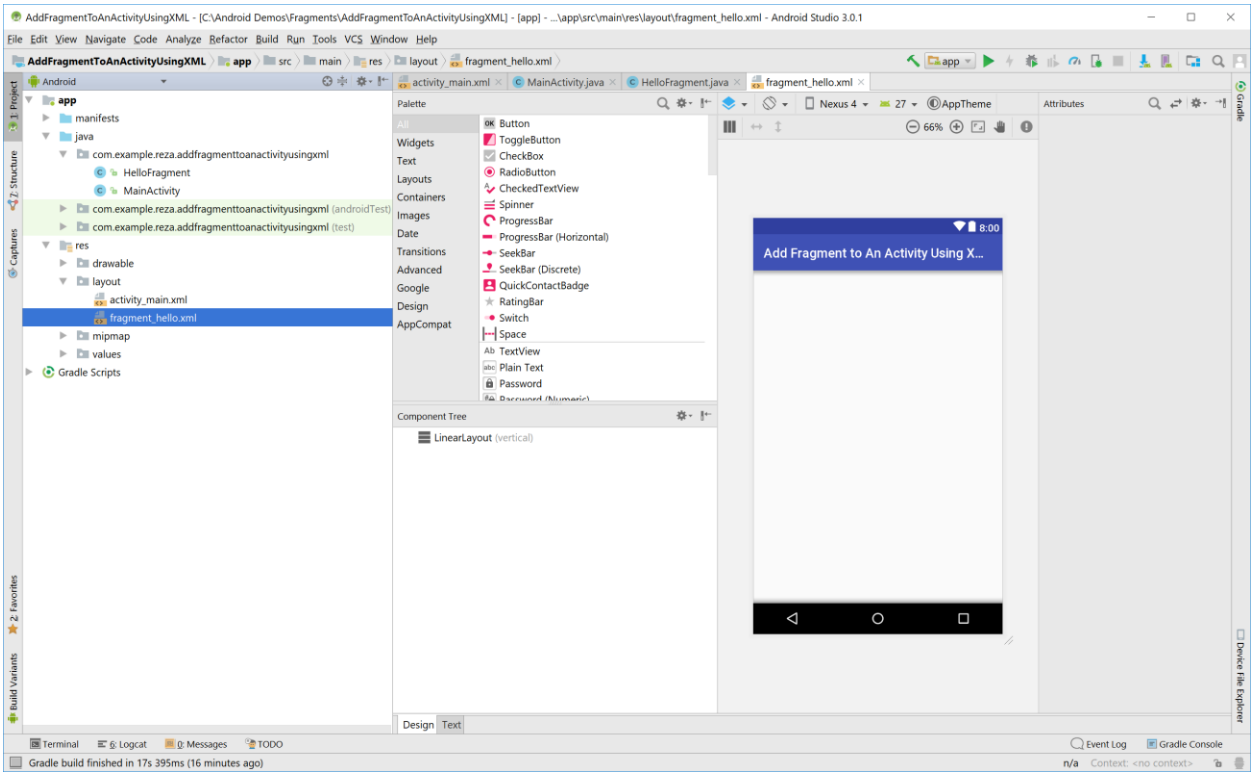

New Resource File
File name:
fragment_hello.xml
Root element:
android.support.constraint.ConstraintLayout
Source set:
main
Directory name:
layout
Available qualifiers:
Country Code
Network Code
Locale
Layout Direction
>>
<<
Chosen qualifiers:
Nothing to show
OK
Cancel
Help

We are going to change the root layout to a LinearLayout as shown:

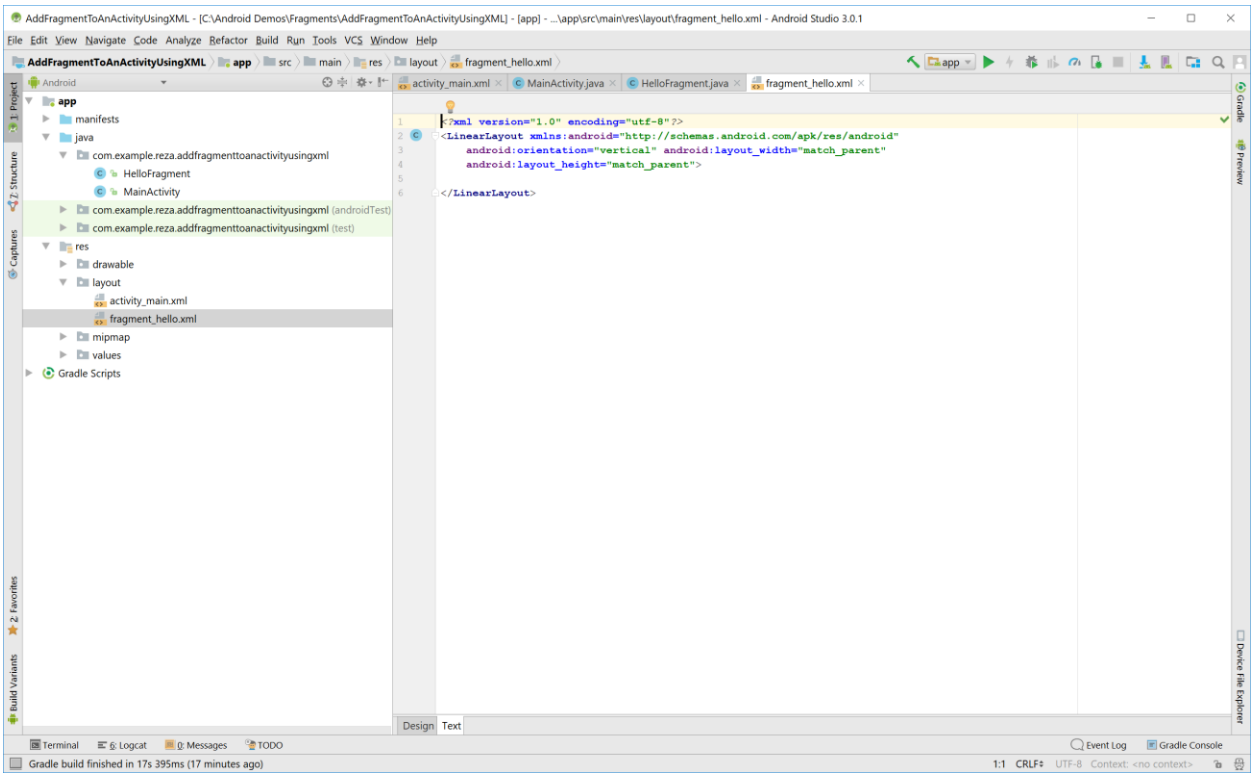



New Resource File
File name:
fragment_hello.xml
Root element:
Li
Source set:
Directory name:
Available qualifi
Country Co
Network Co
Locale
Layout Dire
LinearLayout
ListView
AdapterViewFlipper
ExpandableListView
SlidingDrawer
TwoLineListItem
ViewFlipper
android.support.constraint.Guideline
android.support.v4.widget.SlidingPaneLayout
android.support.v7.app.AlertController.Recycle...
OK
Cancel
Help

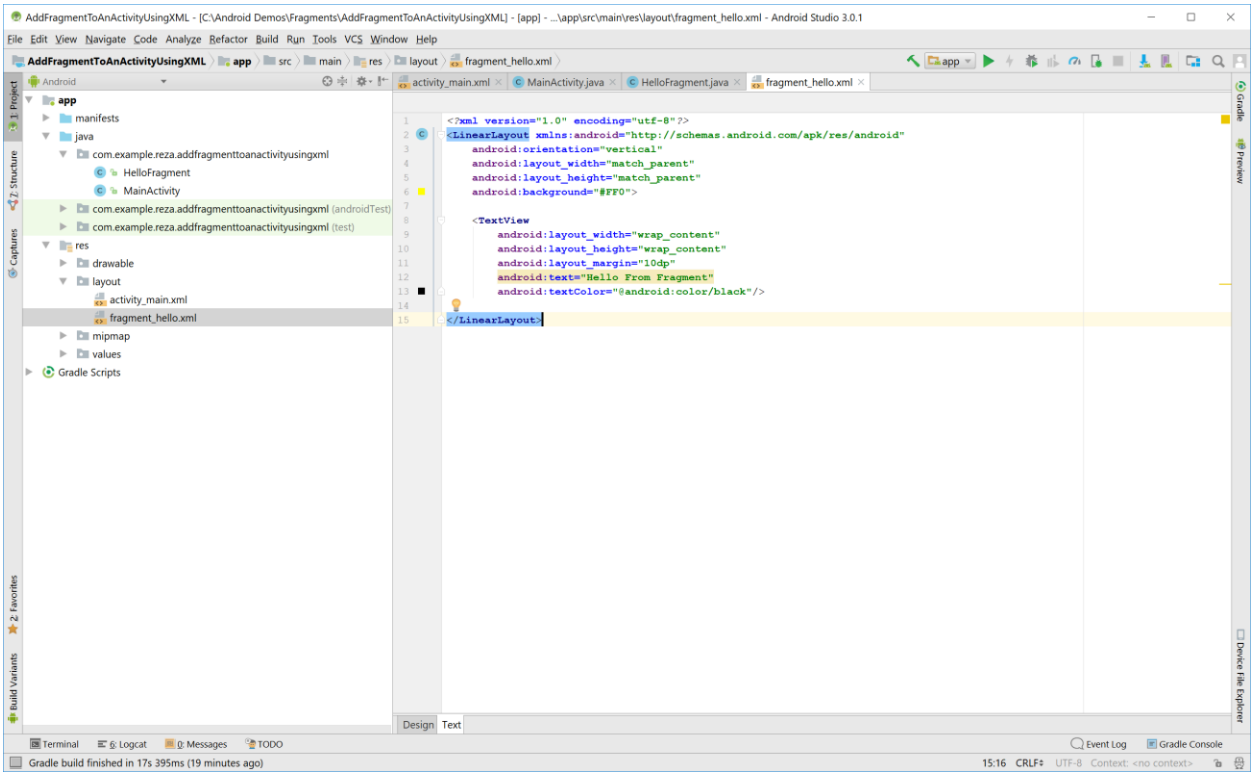

New Resource File
File name:
fragment_hello.xml
Root element:
LinearLayout
Source set:
main
Directory name:
layout
Available qualifiers:
Chosen qualifiers:
Country Code
Network Code
Locale
Layout Direction
>>
<<
Nothing to show
OK
Cancel
Help

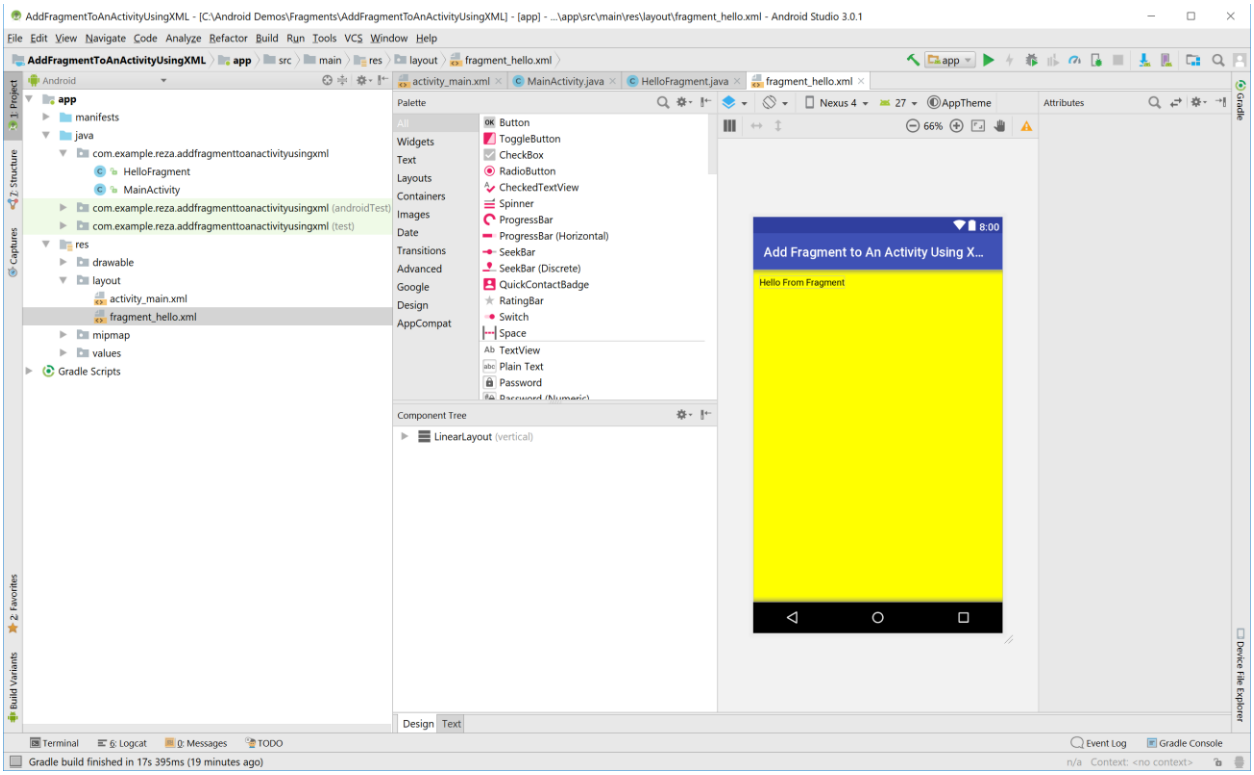


Here is the layout:

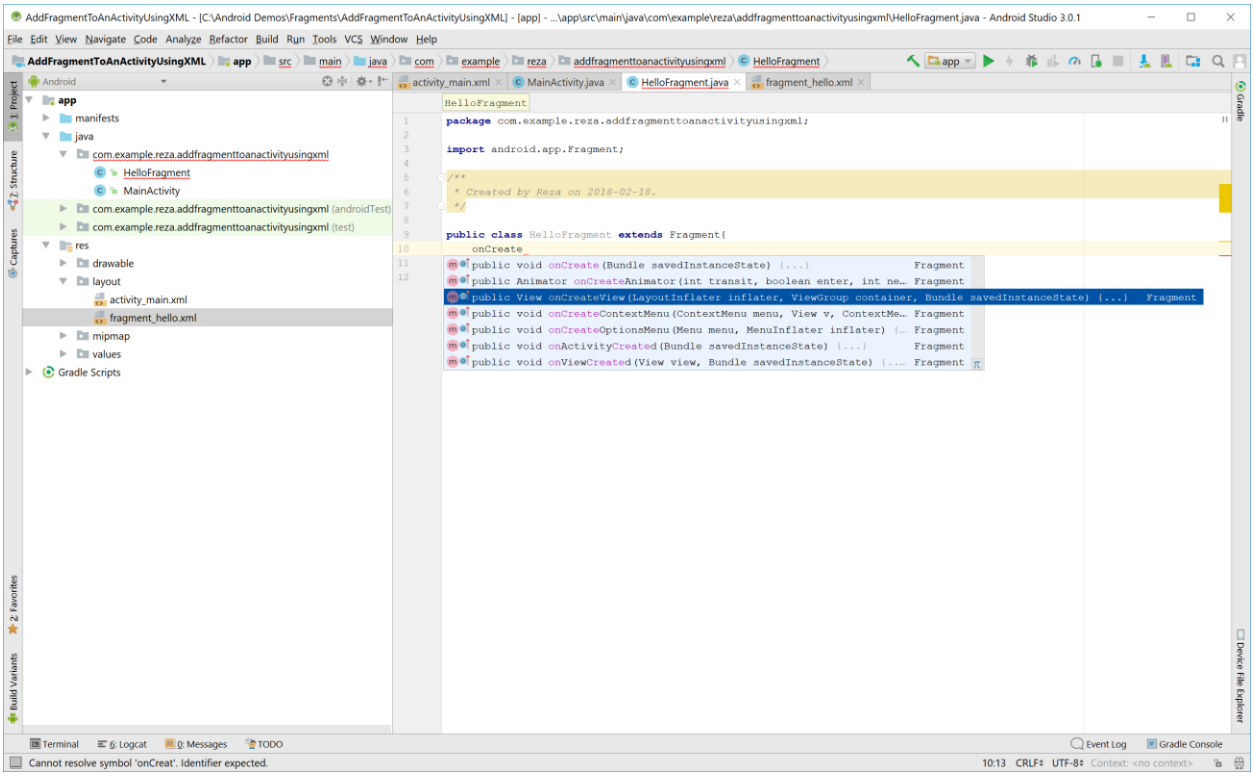


Add an element and set the background color so it is easy to see the fragment when displayed in the activity:

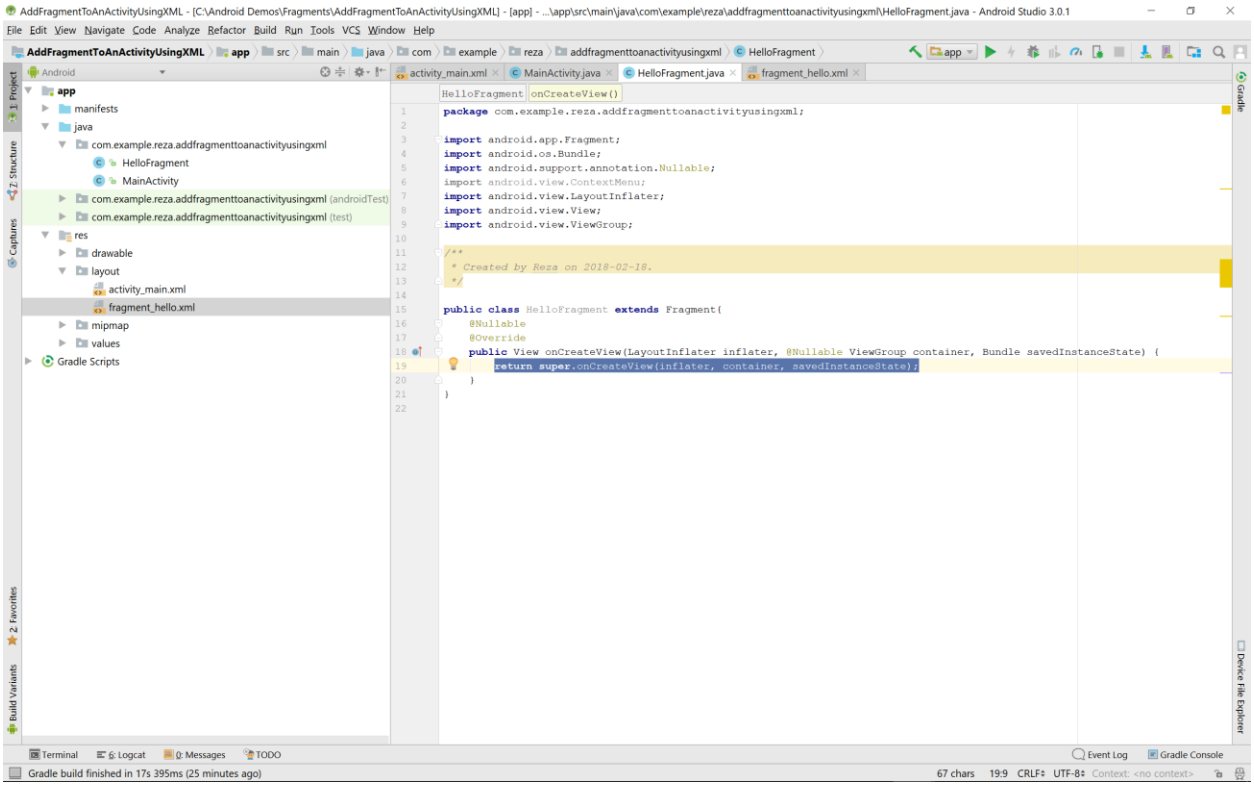


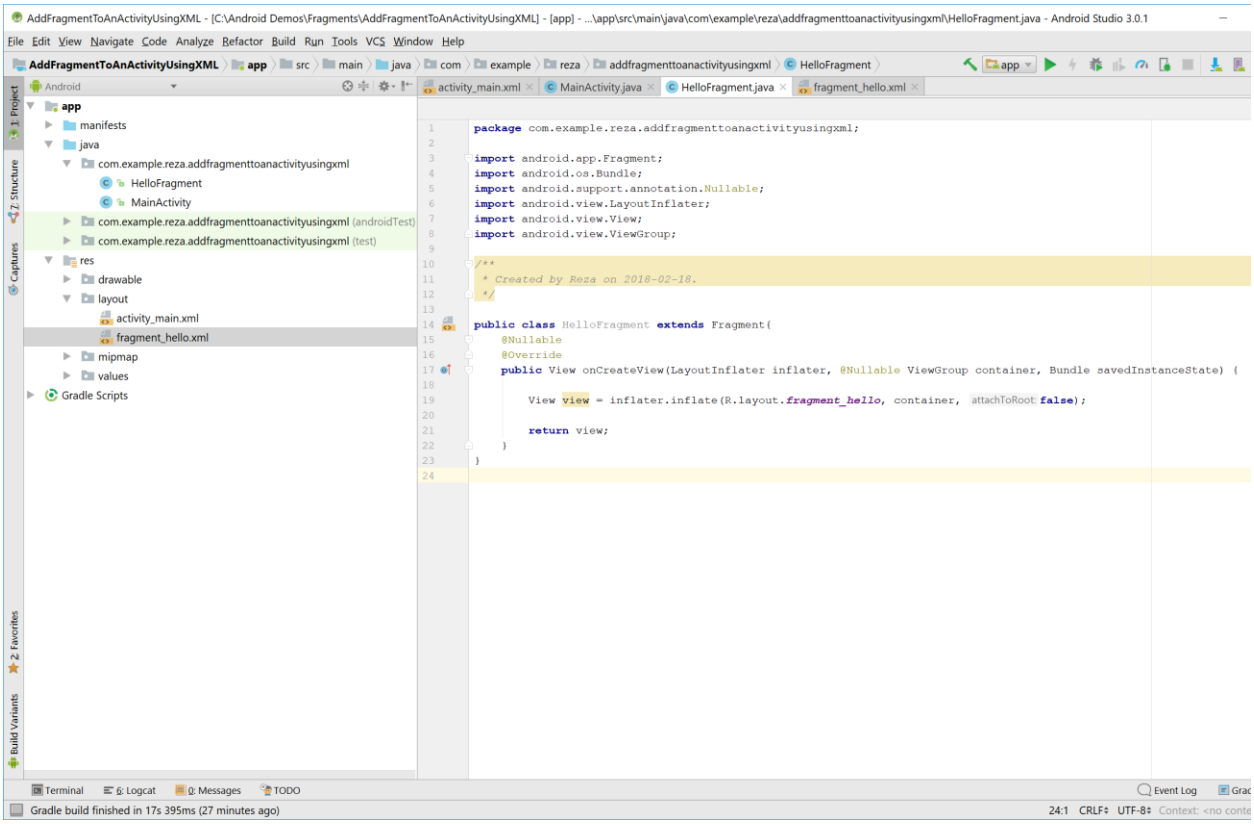


Now go back to HelloFragment.java code and use the onCreate () method to create the connection between the java code and its layout file.

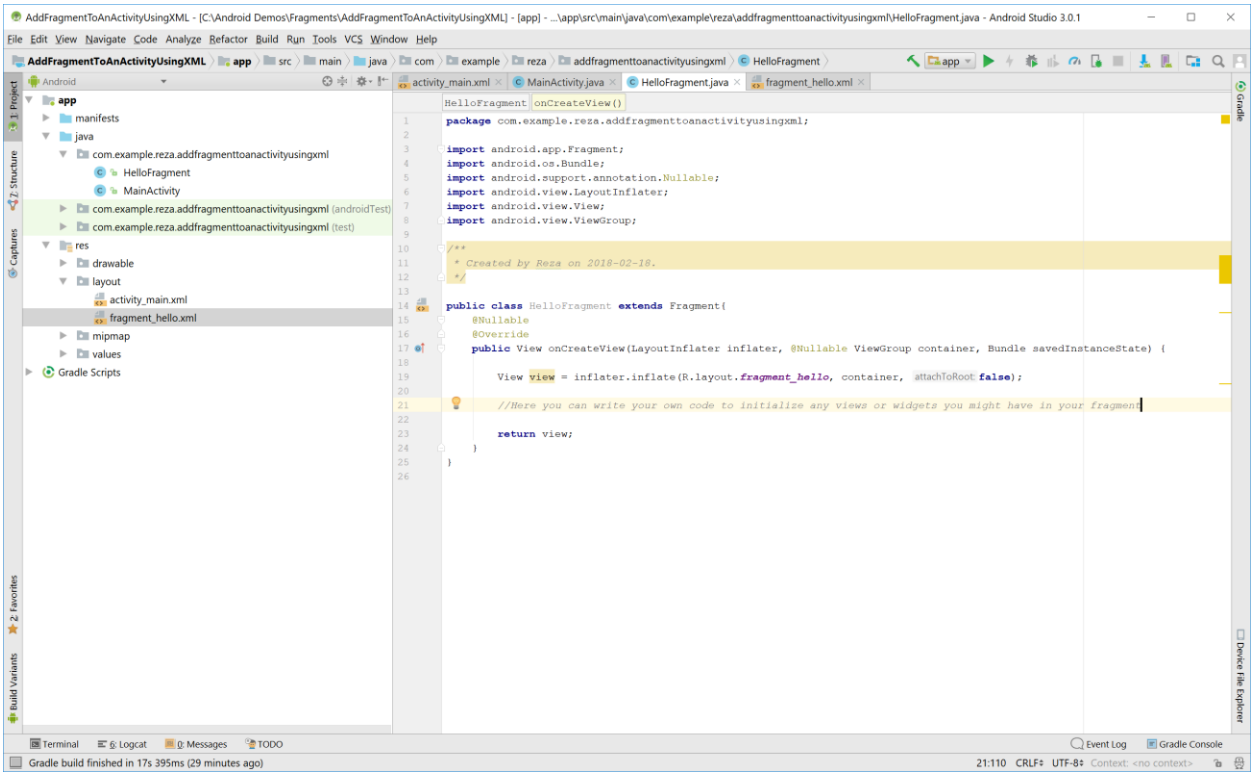


You need to replace the highlighted code as shown:



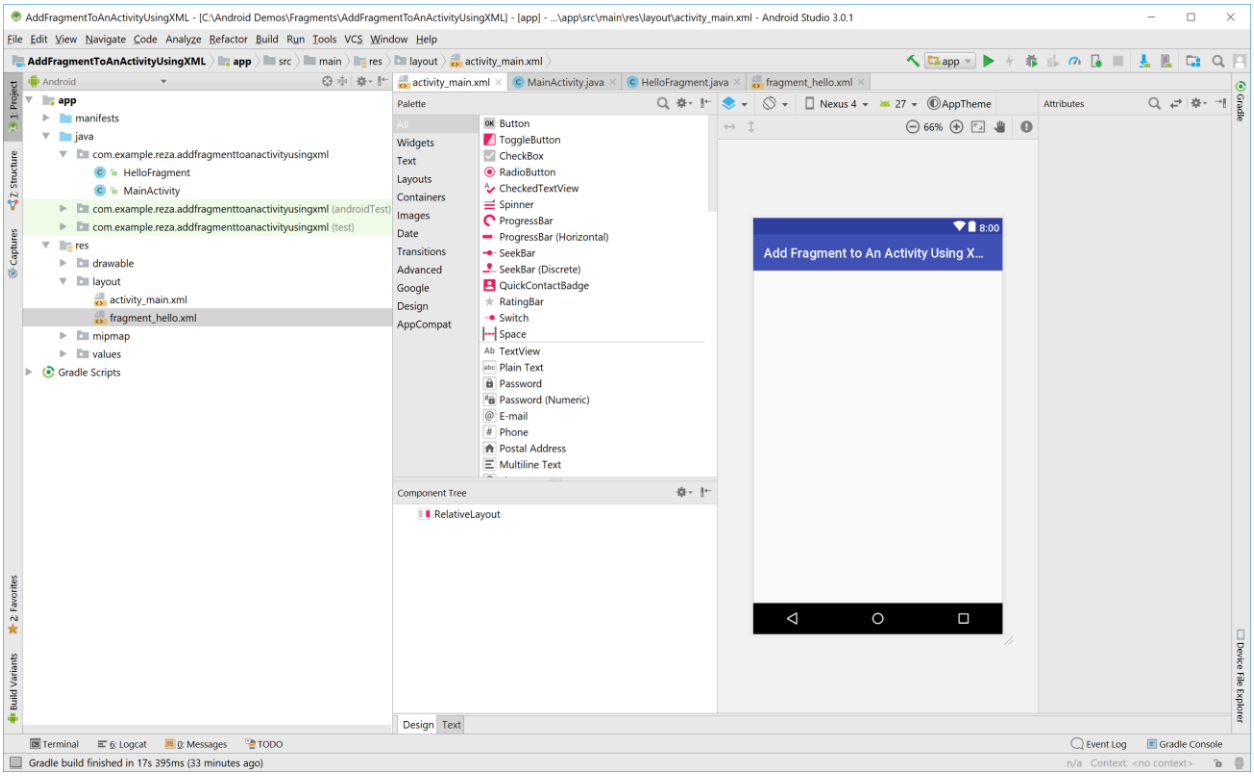


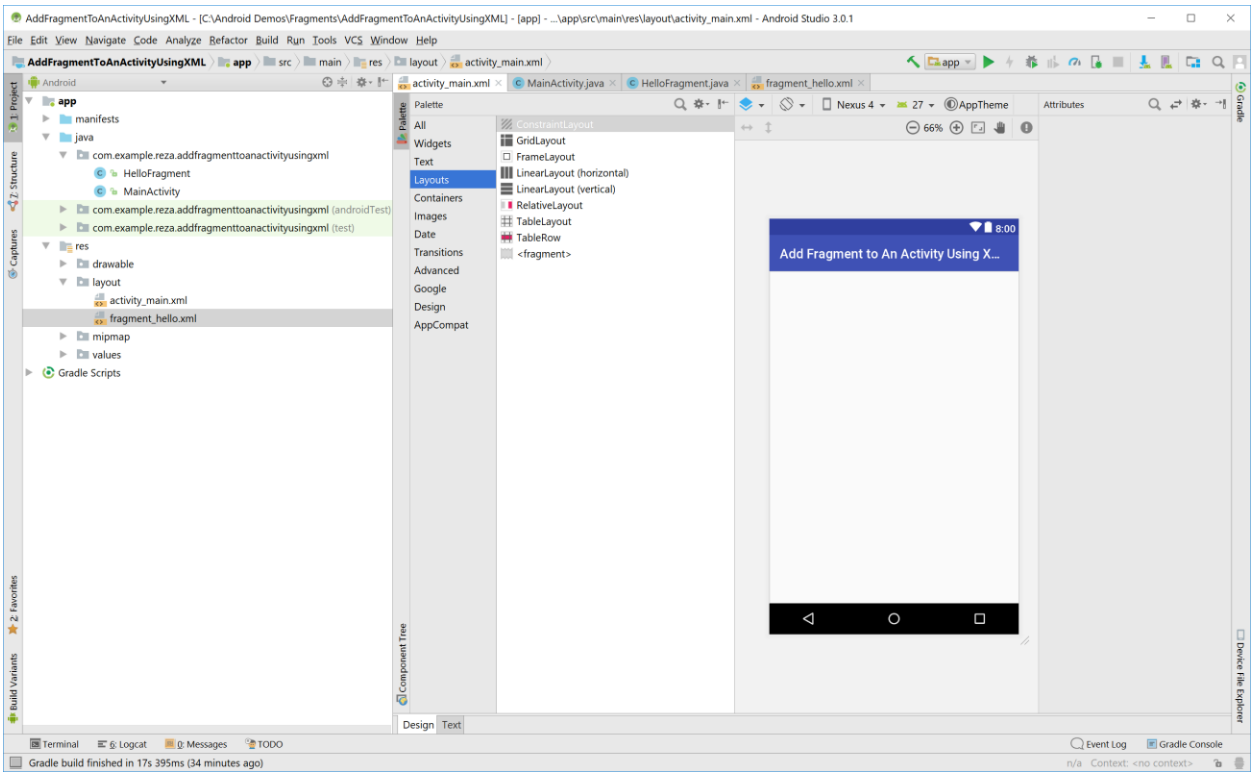
In the comment section you can write the code to initialize any other views you might have in your fragment:



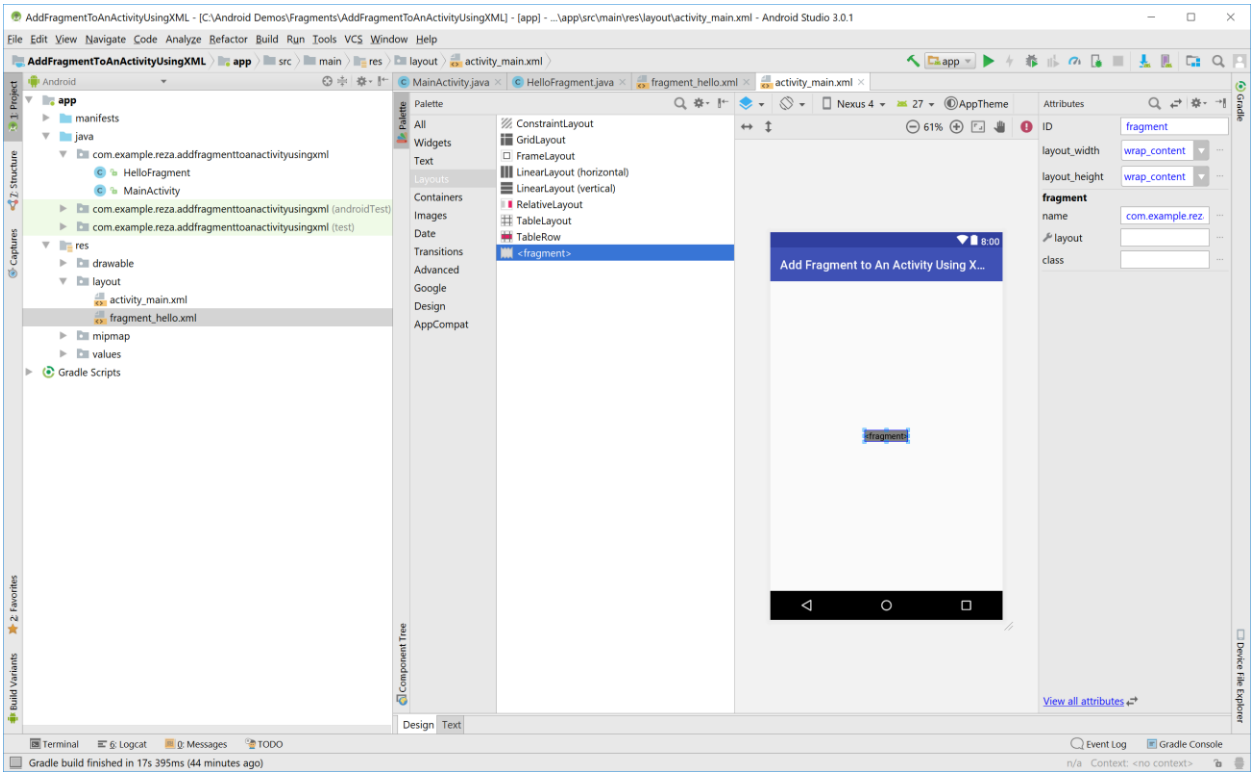
Next you need to add this fragment to the layout for your main activity. The process is the same as adding a TextView or EditText to the main activity layout.

So select activity_main.xml in the design view and under Palette select Layouts and as shown:

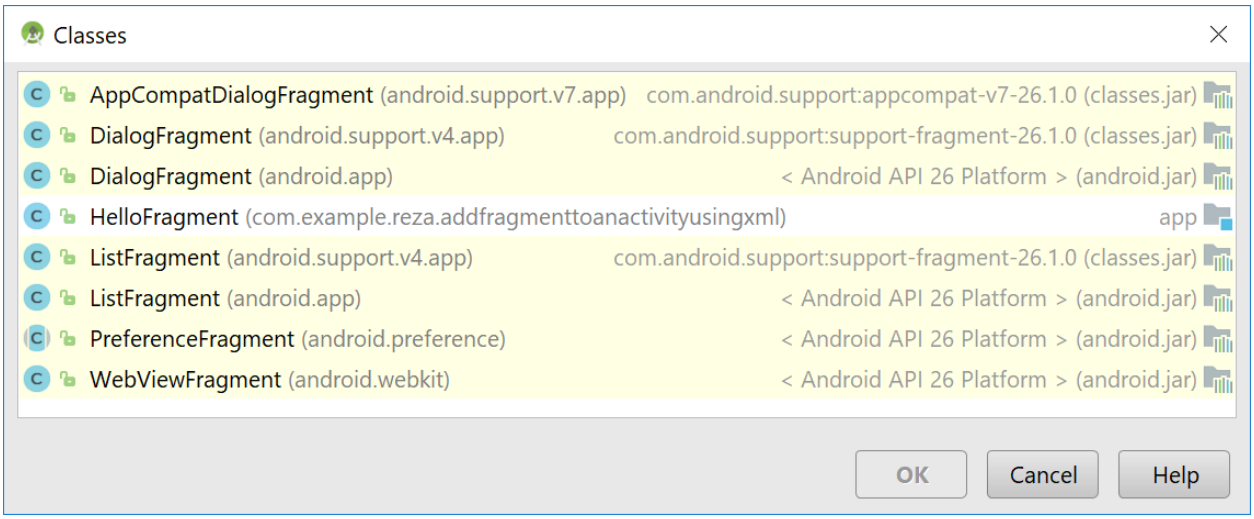

AddFragmentToAnActivityUsingXML - [C:\Android Demos\Fragments\AddFragmentToAnActivityUsingXML] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AddFragmentToAnActivityUsingXML app src main res layout activity_main.xml
activity_main.xml MainActivity.java HelloFragment.java fragment_hello.xml
Palette
Widgets
Text
Layouts
Containers
Images
Date
Transitions
Advanced
Google
Design
AppCompat
Button
ToggleButton
CheckBox
RadioButton
CheckedTextView
Spinner
ProgressBar
ProgressBar (Horizontal)
SeekBar
SeekBar (Discrete)
QuickContactBadge
RatingBar
Switch
Space
TextView
Plain Text
Password
Password (Numeric)
E-mail
Phone
Postal Address
Multiline Text
Component Tree
RelativeLayout
Nexus 4
AppTheme
Attributes
8:00
Add Fragment to An Activity Using X...
Design
Text
Terminal
6: Logcat
0: Messages
TODO
Event Log
Gradle Console
Gradle build finished in 17s 395ms (33 minutes ago)

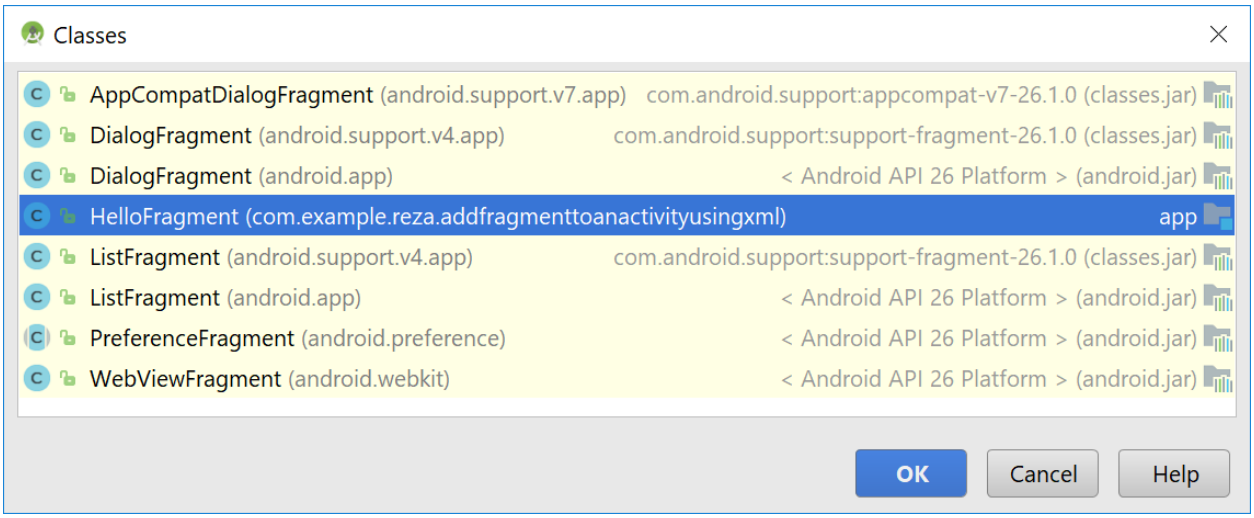

AddFragmentToAnActivityUsingXML - [C:\Android Demos\Fragments\AddFragmentToAnActivityUsingXML] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AddFragmentToAnActivityUsingXML app src main res layout activity_main.xml
activity_main.xml MainActivity.java HelloFragment.java fragment_hello.xml
Palette
Widgets
Text
Layouts
Containers
Images
Date
Transitions
Advanced
Google
Design
AppCompat
ConstraintLayout
GridLayout
FrameLayout
LinearLayout (horizontal)
LinearLayout (vertical)
RelativeLayout
TableLayout
TableRow
<fragment>
Nexus 4
AppTheme
Attributes
8:00
Add Fragment to An Activity Using X...
Design
Text
Terminal
6: Logcat
0: Messages
TODO
Event Log
Gradle Console
Gradle build finished in 17s 395ms (34 minutes ago)

Now select <fragment> and drag into the activity_main.xml file and you will get this pop up that shows you several different fragments as shown:

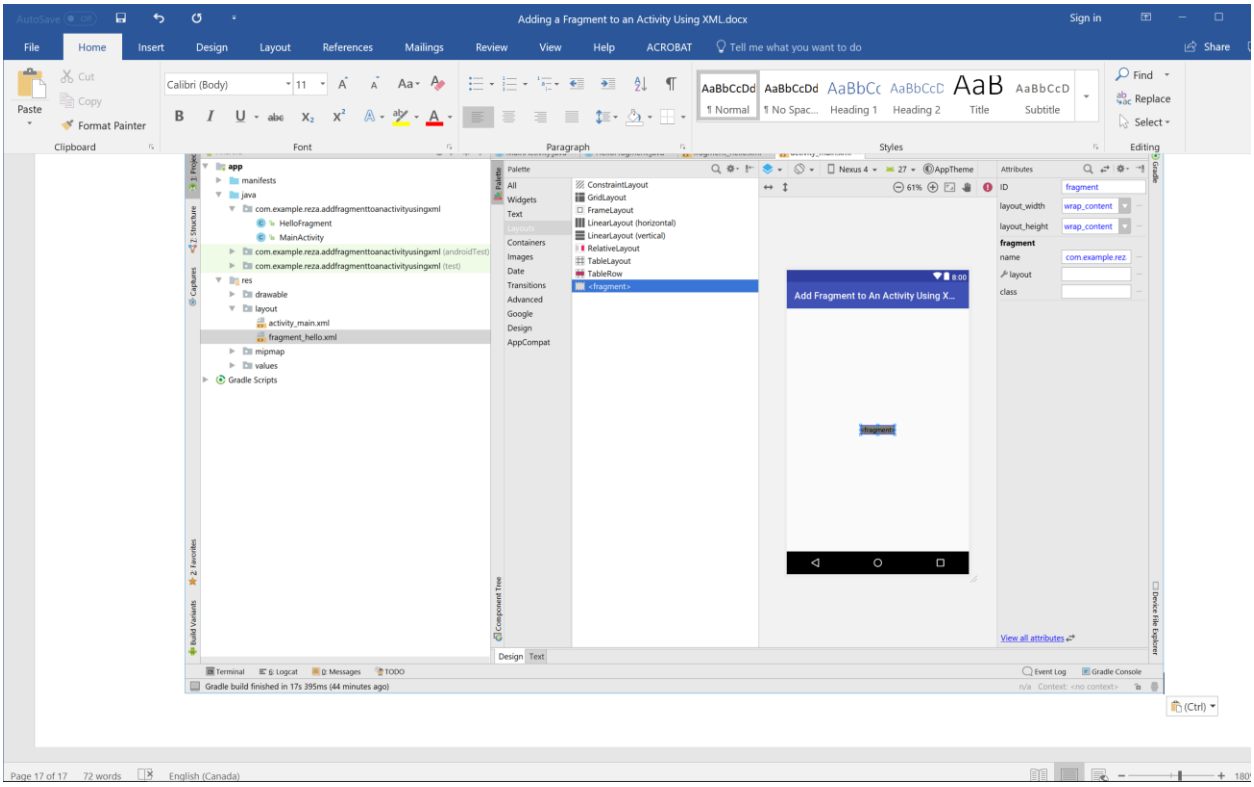


You see you can add many different predefined fragments but what you are interested is the HelloFragment fragment so select HelloFragment and click Ok as shown:

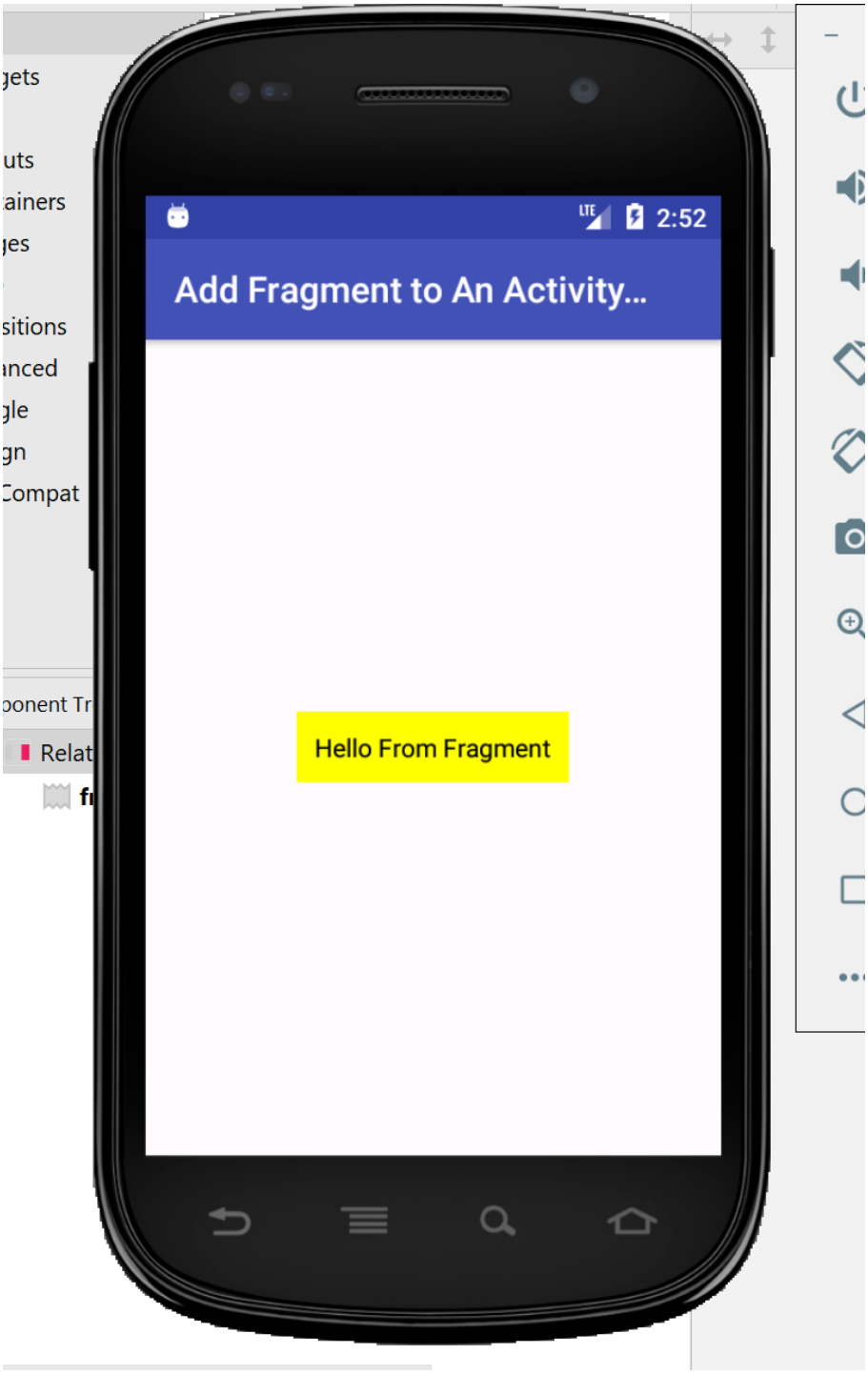



Classes
AppCompatDialogFragment (android.support.v7.app) com.android.support:appcompat-v7-26.1.0 (classes.jar)
DialogFragment (android.support.v4.app) com.android.support:support-fragment-26.1.0 (classes.jar)
DialogFragment (android.app) < Android API 26 Platform > (android.jar)
HelloFragment (com.example.reza.addfragmenttoanactivityusingxml) app
ListFragment (android.support.v4.app) com.android.support:support-fragment-26.1.0 (classes.jar)
ListFragment (android.app) < Android API 26 Platform > (android.jar)
PreferenceFragment (android.preference) < Android API 26 Platform > (android.jar)
WebViewFragment (android.webkit) < Android API 26 Platform > (android.jar)
OK
Cancel
Help

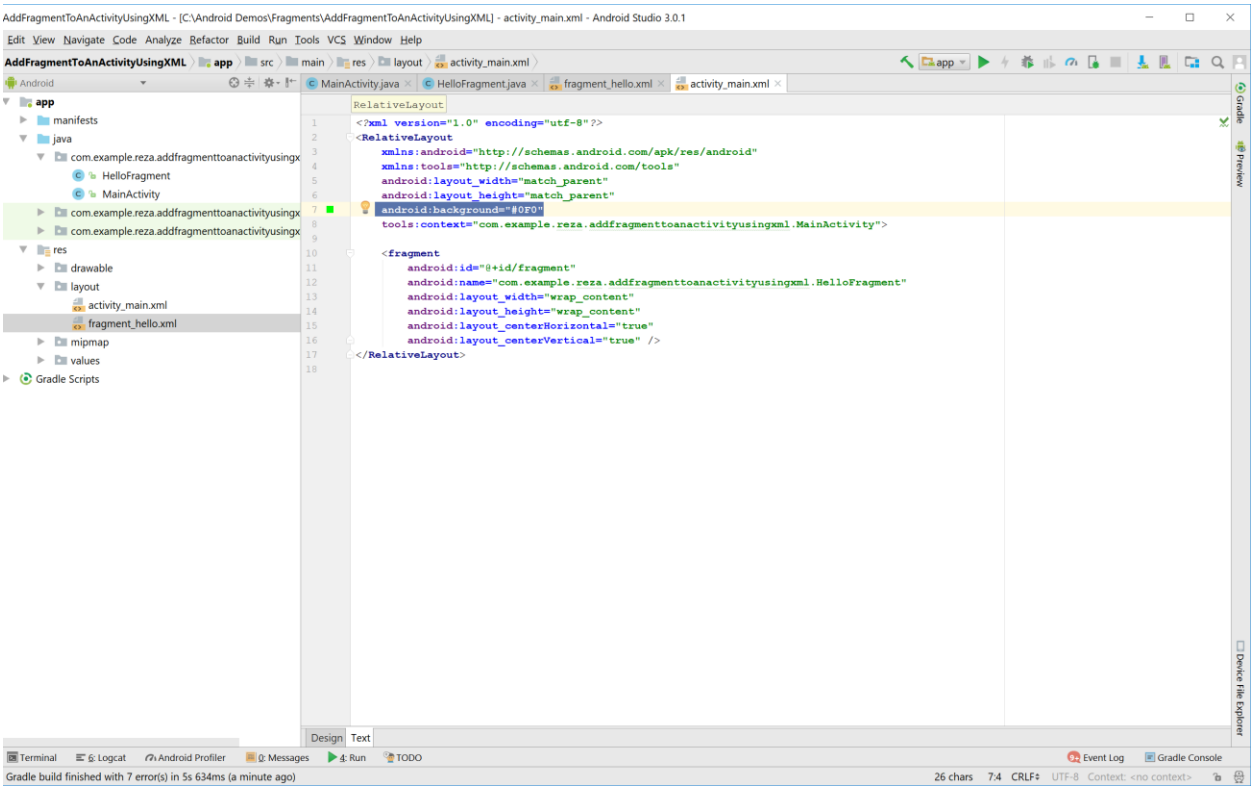
The fragment has been added to the layout file of the main activity. Again you can think of a fragment as another view that you drag to your design of your main activity:



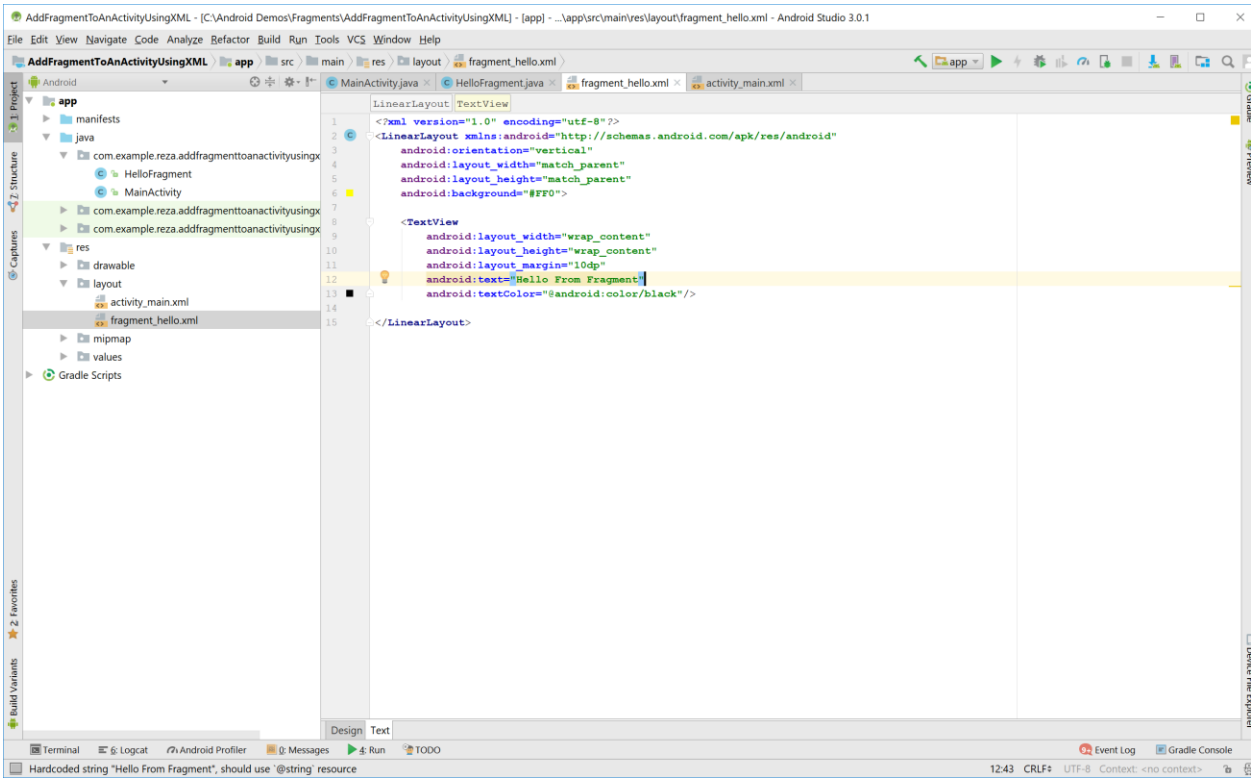
Now if you run the app, you will see your fragment in the activity as shown:



We can change the background color of layout for the main activity to a different color. So add the following to the RelativeLayout tag to assign a background color to the main activity so we are giving a green background to the main activity and our fragment has the yellow background:



Now you run the app again and you will see your fragment displayed in your main activity. Notice that our fragment with its yellow background contains only one text view which has the option wrap-content for both of its height and width so it appears the way it does (see the screen capture below for its xml code) but it could also contain any other view elements.



And here is our app running: