



# Menu & Fragment

Asst. Prof. Monlica Wattana, Ph.D  
Department of Computer Science,  
Khon Kaen University



# Outline

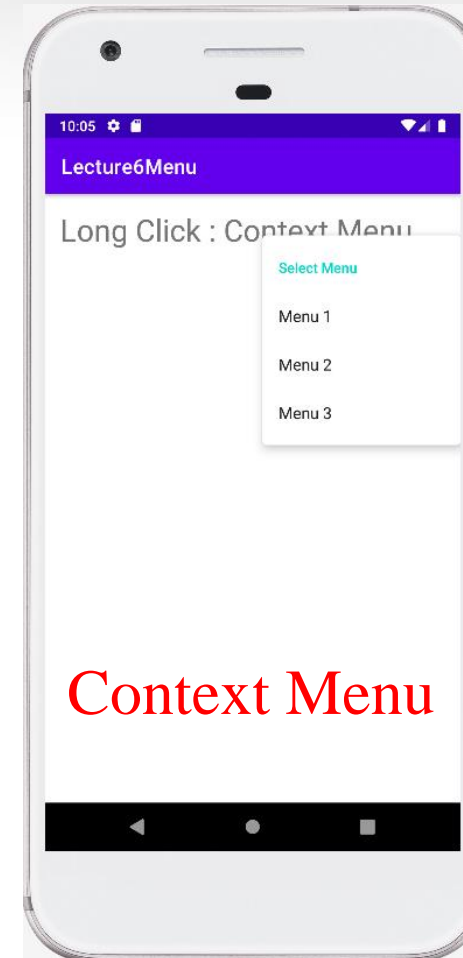
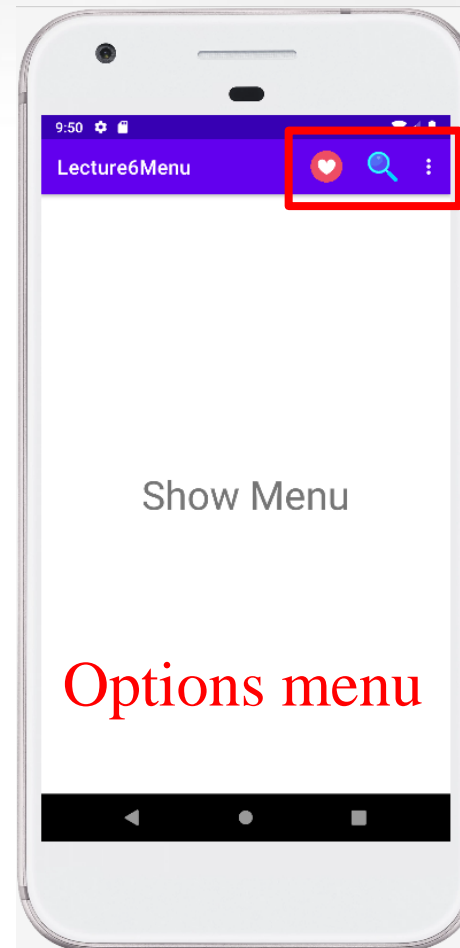
- Menu
- Fragment

# Menu

A Menu is a common user interface component in many types of applications.

## Types of Menu

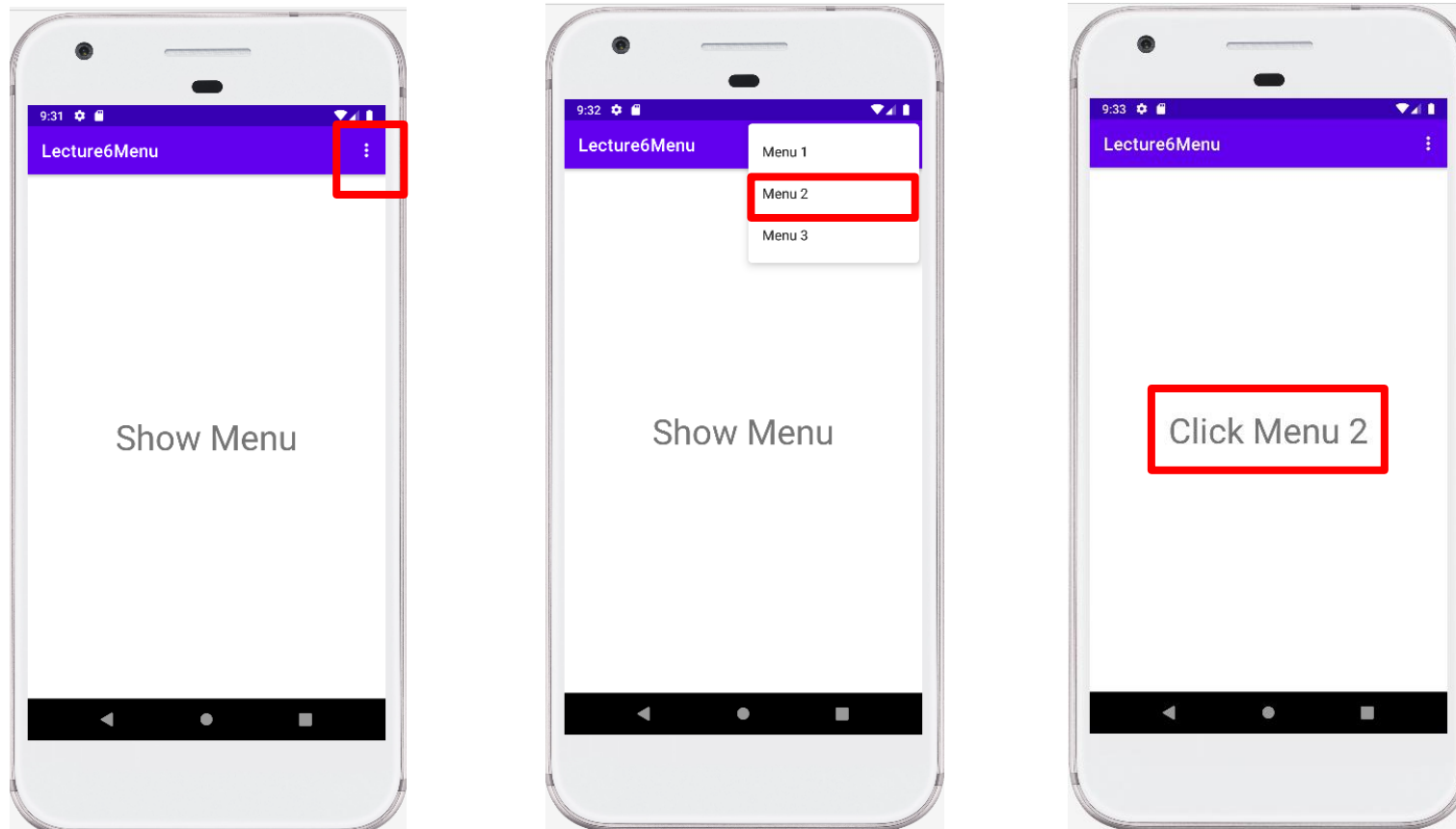
- Options menu
- Context Menu
- Popup menu





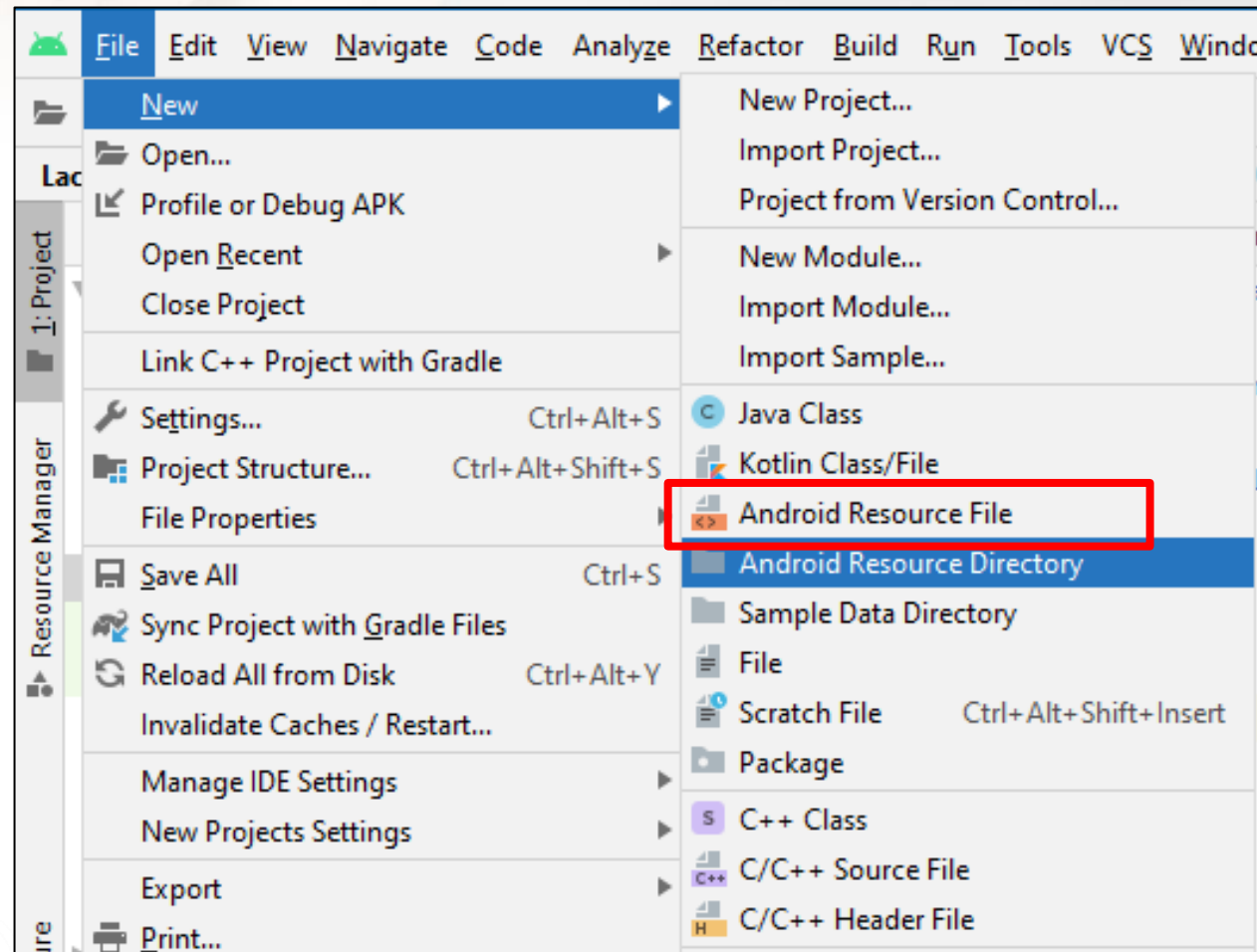
# Menu: Options menu

- **Options menu:** The options menu is the primary collection of menu items for an activity.
- E.g. : 'Search', 'Compose email', and 'Settings'.



# Menu: Options menu

- Create Options menu : New>> Android Resource File



# Menu: Options menu

## - Create Options menu (cont.)

New Resource File

File name:

Resource type:

Root element:

Source set:

Directory name:

Available qualifiers:

- Country Code
- Network Code
- Locale
- Layout Direction
- Smallest Screen Width
- Screen Width
- Screen Height
- Size
- Ratio
- Orientation

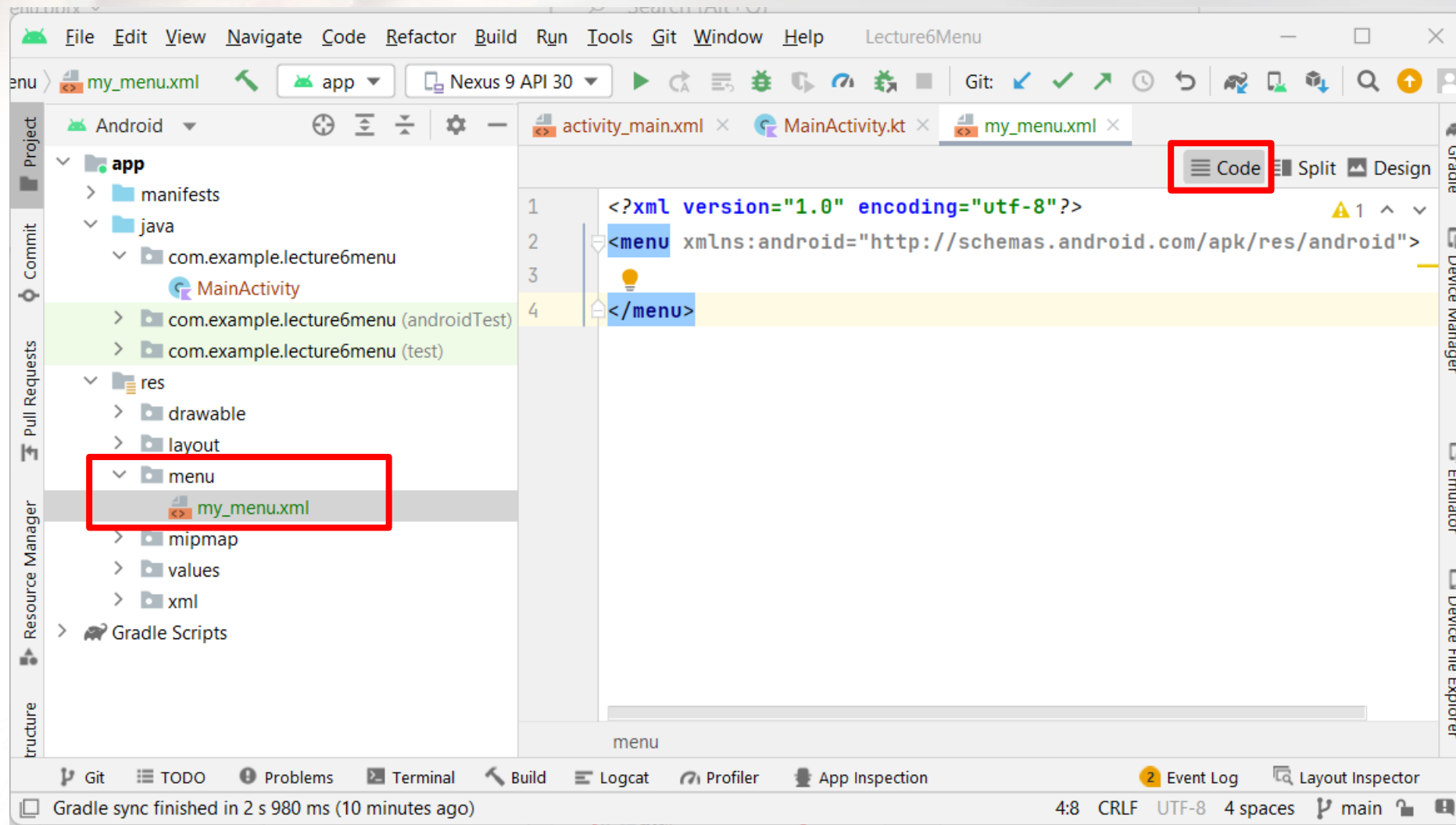
Chosen qualifiers:

Nothing to show

OK Cancel

# Menu: Options menu

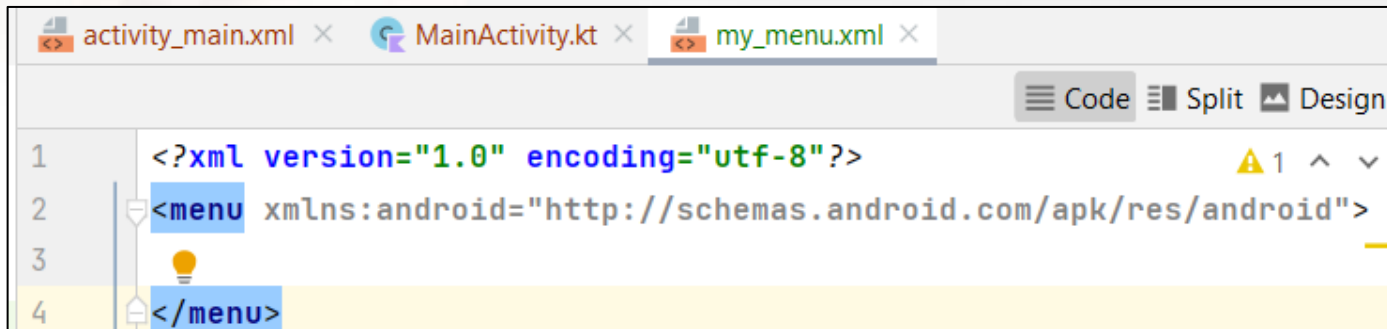
## - Create Options menu (cont.)





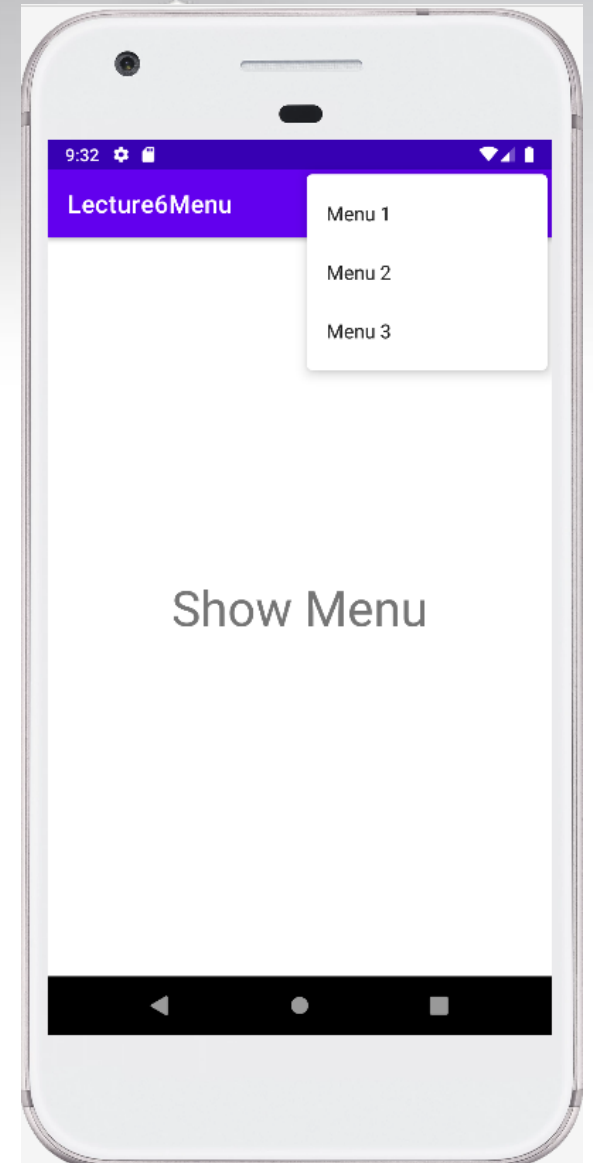
# Menu: Options menu

- res>> menu>> my\_menu.xml



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <menu xmlns:android="http://schemas.android.com/apk/res/android">
3
4 </menu>
```

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/menu1"
          android:title="Menu 1"/>
    <item android:id="@+id/menu2"
          android:title="Menu 2"/>
    <item android:id="@+id/menu3"
          android:title="Menu 3" />
</menu>
```

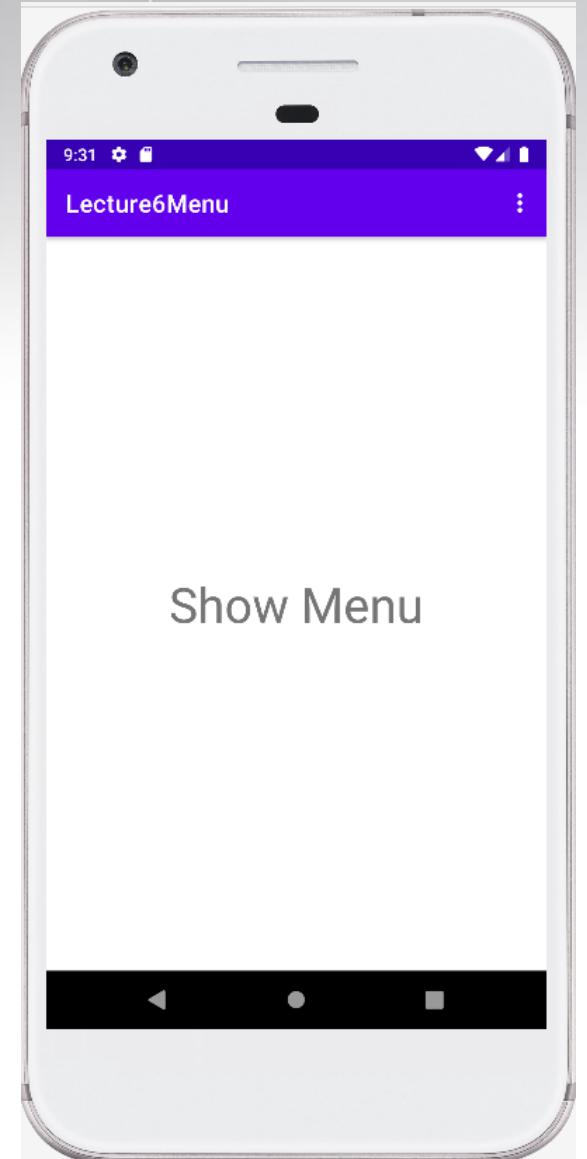




# Menu: Options menu

- activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Show Menu"
        android:layout_centerInParent="true"
        android:textSize="40sp"/>
</RelativeLayout>
```

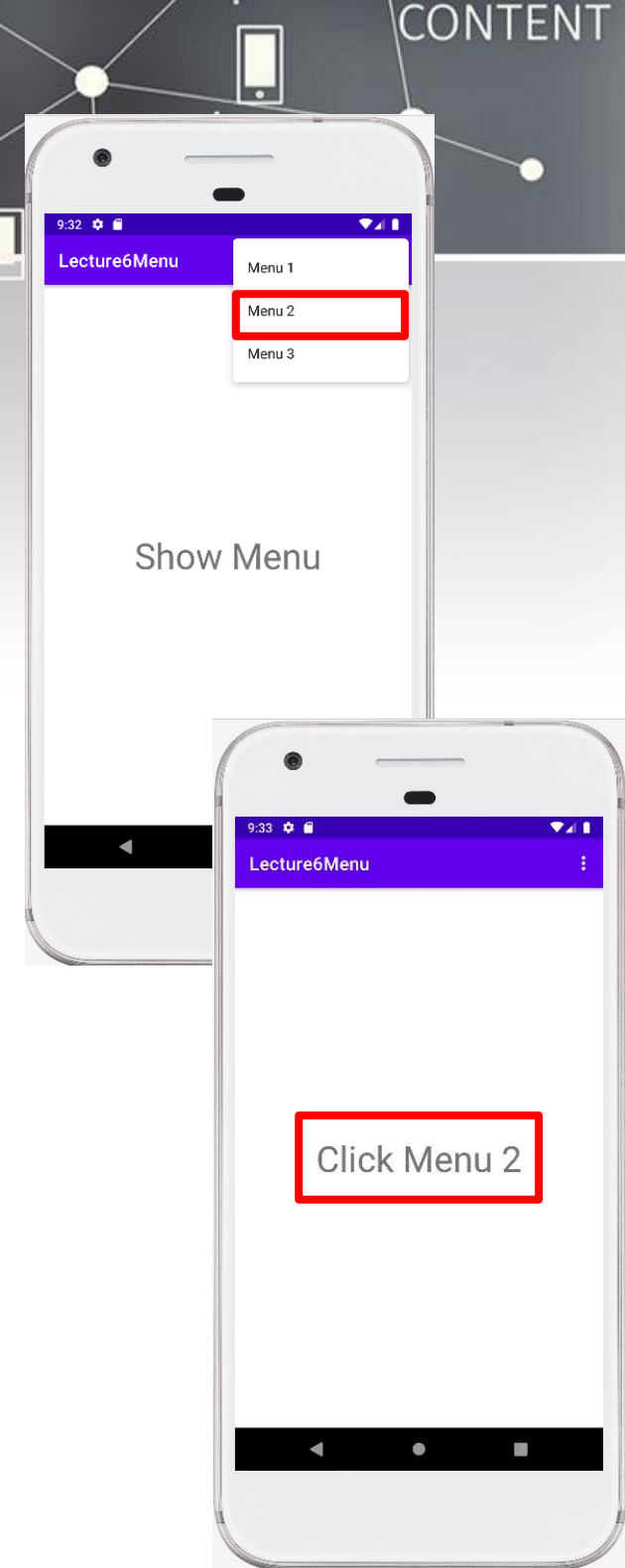


## MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
    }  
    override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
        menuInflater.inflate(R.menu.my_menu, menu)  
        return super.onCreateOptionsMenu(menu)  
    }  
    override fun onOptionsItemSelected(item: MenuItem): Boolean {  
        when (item.itemId) {  
            R.id.menu1 -> binding.textView.text = "Click Menu 1"  
            R.id.menu2 -> binding.textView.text = "Click Menu 2"  
            R.id.menu3 -> binding.textView.text = "Click Menu 3"  
            else -> binding.textView.text = "No Selected"  
        }  
        return super.onOptionsItemSelected(item)  
    }  
}
```

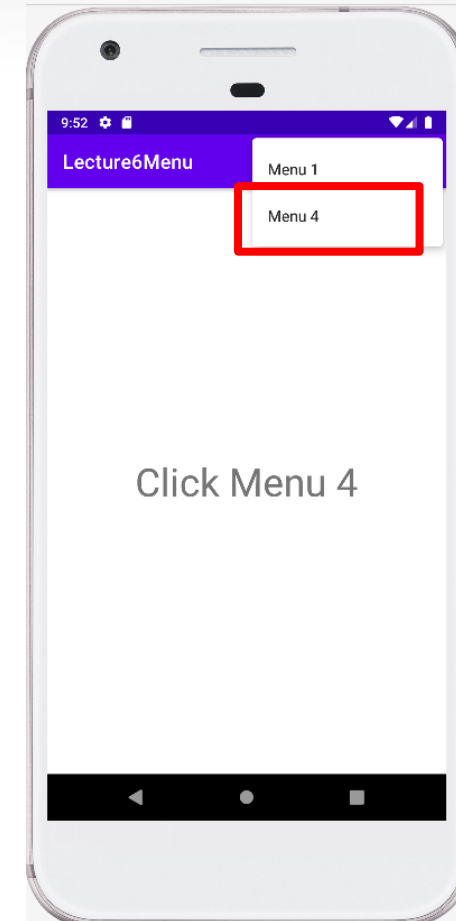
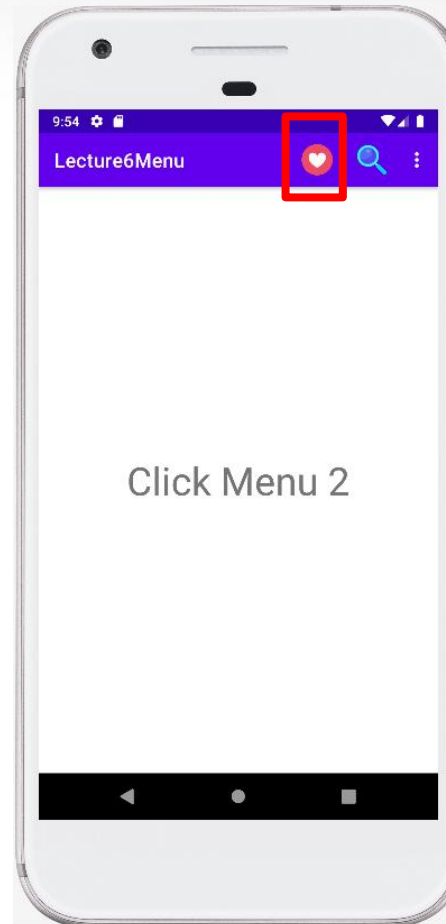
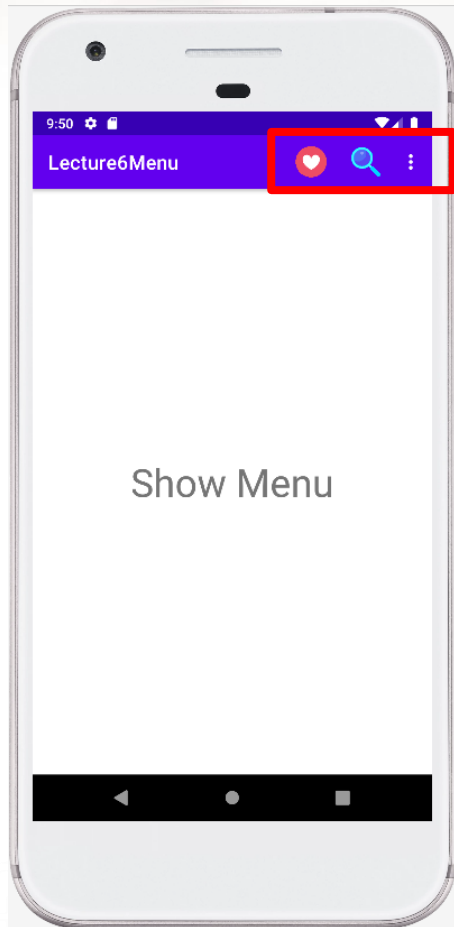
Create menu

# Menu: Options menu



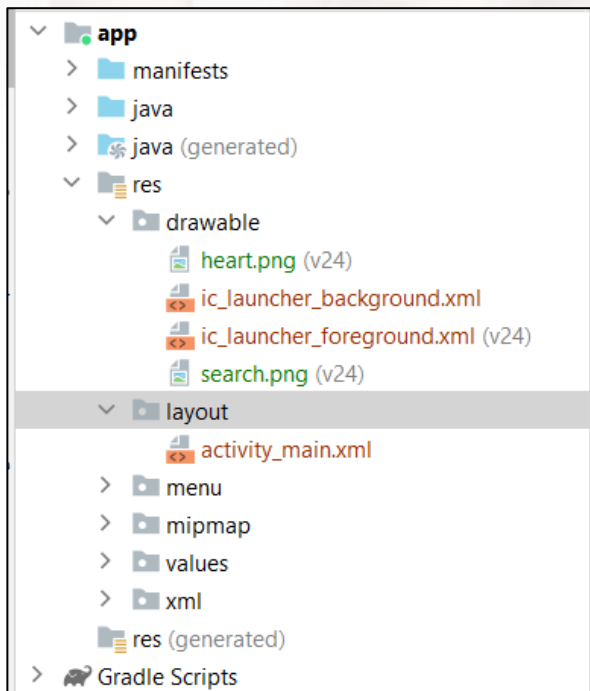
# Menu: Option Menu with Icon

## Option Menu with Icon

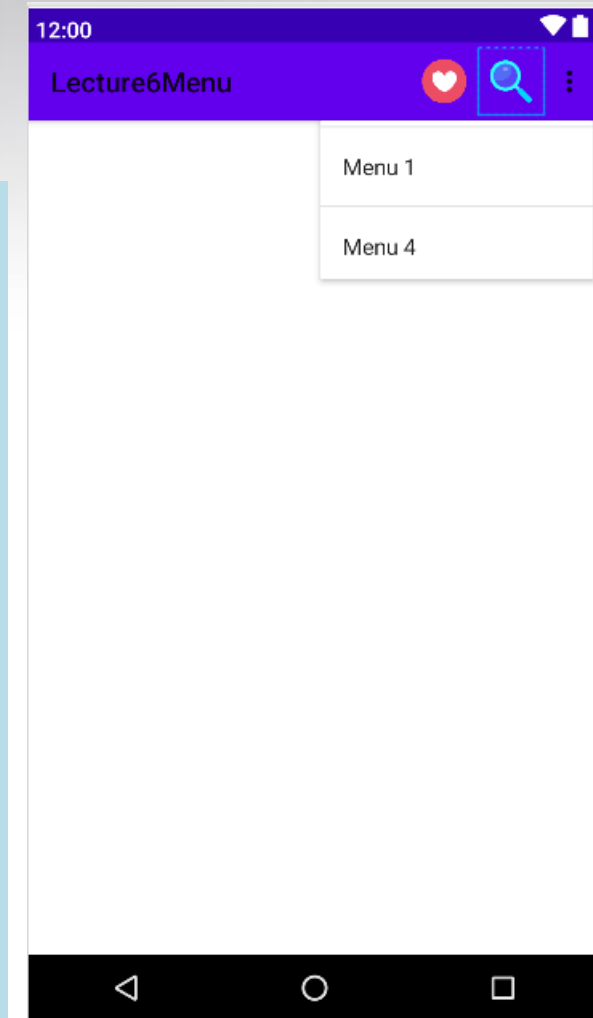


# Menu: Option Menu with Icon

- res>> menu>> my\_menu\_icon.xml



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/menu1"
          android:title="Menu 1"/>
    <item android:id="@+id/menu2"
          android:title="Menu 2"
          android:icon="@drawable/heart"
          app:showAsAction="always"/>
    <item android:id="@+id/menu3"
          android:title="Menu 3"
          android:icon="@drawable/search"
          app:showAsAction="ifRoom"/>
    <item android:id="@+id/menu4"
          android:title="Menu 4"/>
</menu>
```

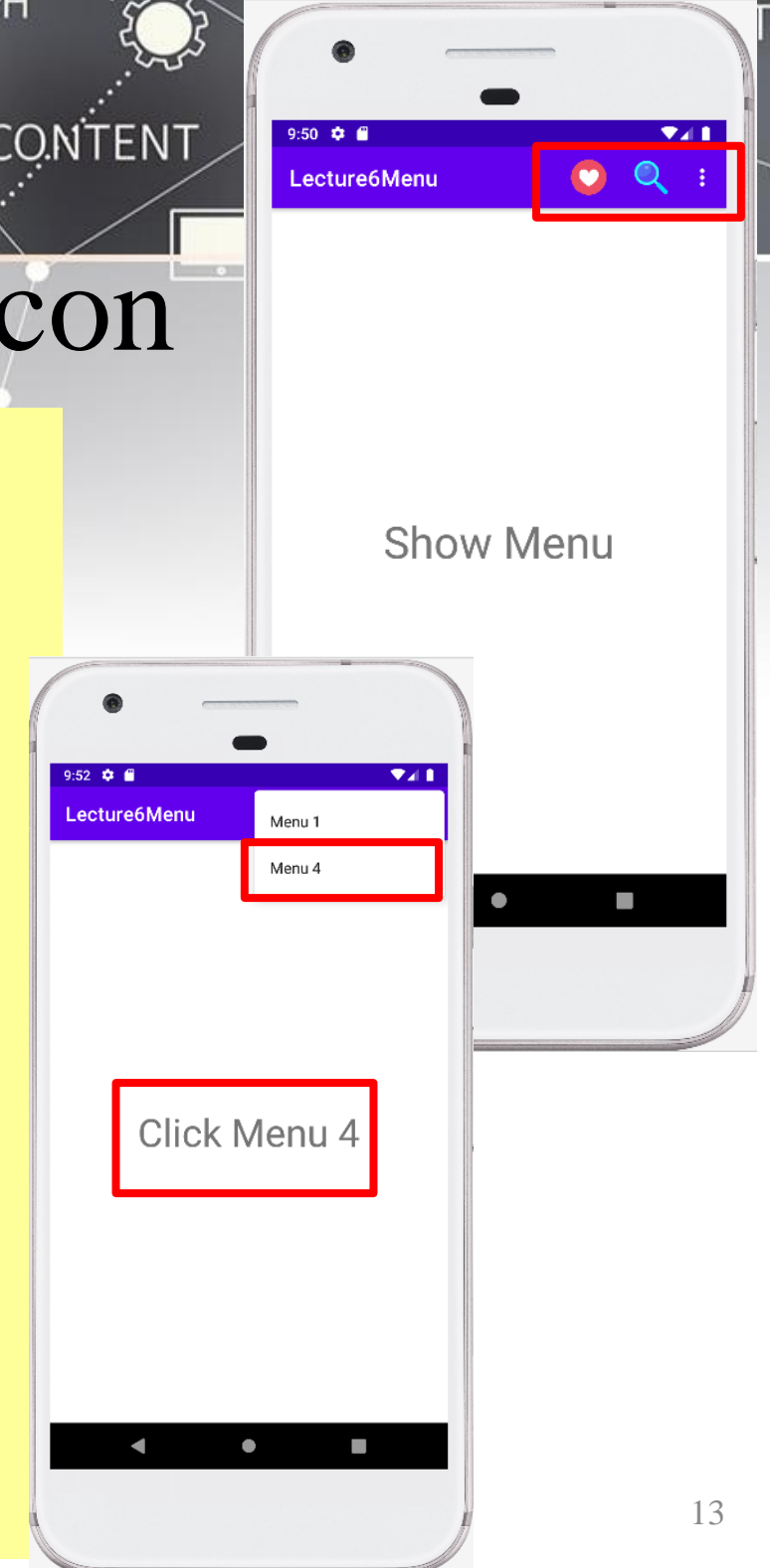




# Menu: Options menu with Icon

MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
    }  
    override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
        menuInflater.inflate(R.menu.my_menu_icon, menu)  
        return super.onCreateOptionsMenu(menu)  
    }  
    override fun onOptionsItemSelected(item: MenuItem): Boolean {  
        when (item.itemId) {  
            R.id.menu1 -> binding.textView.text = "Click Menu 1"  
            R.id.menu2 -> binding.textView.text = "Click Menu 2"  
            R.id.menu3 -> binding.textView.text = "Click Menu 3"  
            R.id.menu4 -> binding.textView.text = "Click Menu 4"  
            else -> binding.textView.text = "No Selected"  
        }  
        return super.onOptionsItemSelected(item)  
    }  
}
```





## Menu: Context Menus

- Context Menu are “**right-click**” Menu
- Android supports context menus through an action called a *long click*
- A long click is a mouse click held down slightly longer than usual on any Android view
- Context Menus are associated with views



## Using Context Menus

- Implementing a context menu:
  - Register a view for a context menu
  - Populate the context menu
  - Respond to context-menu clicks.



## Register a view for a context menu

- View registration for the context menu is done in `onCreate()` method of an Activity
  - The registration is done using `registerForContextMenu` method
- Ex.

```
registerForContextMenu(this.getTextView());
```





# Populating a Context Menu

- Once a view is registered for context menus, Android will call the `onCreateContextMenu()` method with this view as the argument
- This is where you can populate the context menu items for that context menu (same as regular menu item)

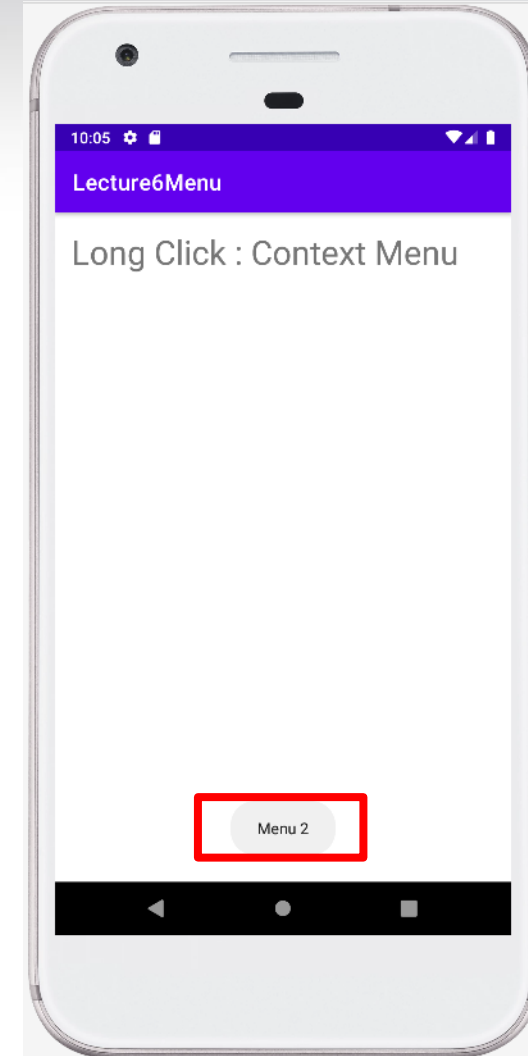
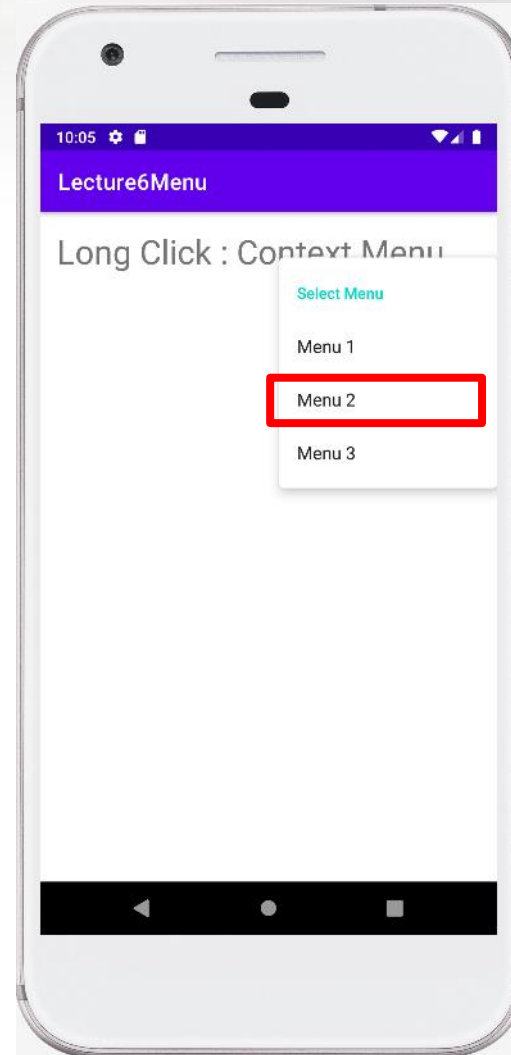
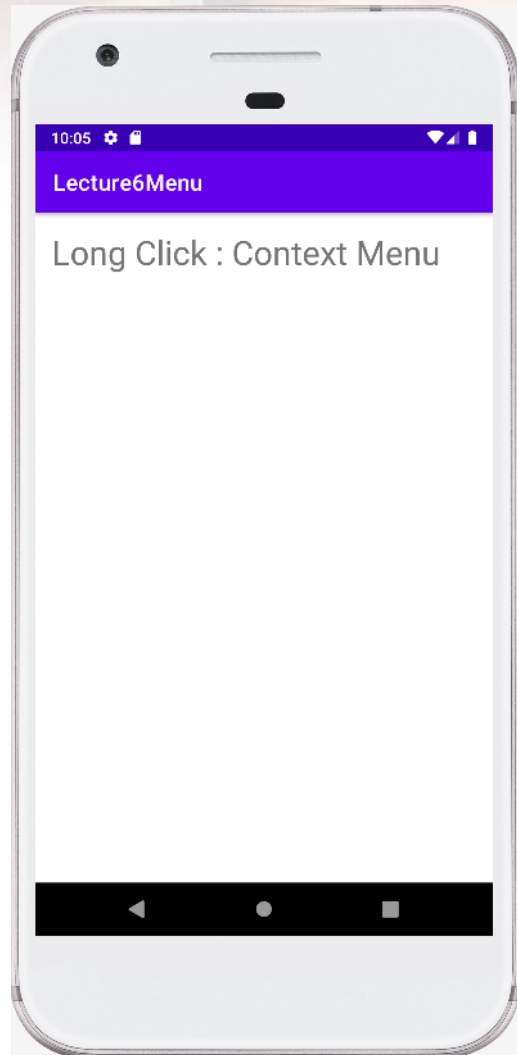
```
override fun onCreateContextMenu(menu: ContextMenu?,  
v: View?, menuInfo: ContextMenu.ContextMenuInfo?) {  
  
    super.onCreateContextMenu(menu, v, menuInfo)  
    ...  
}
```



# Responding to Context Menu Items

- Android provides a callback method called `onContextItemSelected()`

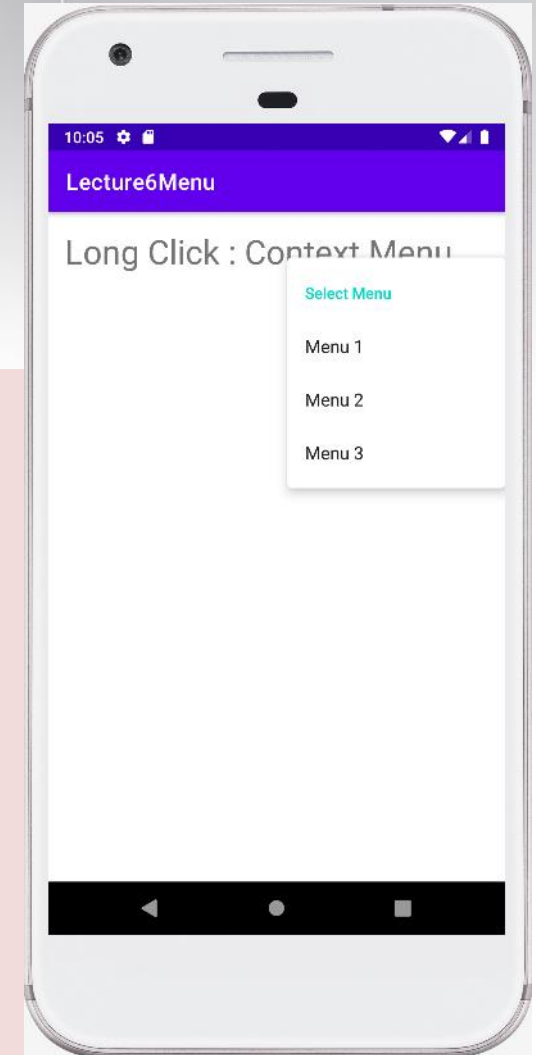
# Menu: Context Menu



# Menu: Context Menu

- res>> menu>> my\_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/menu1"
    android:title="Menu 1"/>
  <item android:id="@+id/menu2"
    android:title="Menu 2"/>
  <item android:id="@+id/menu3"
    android:title="Menu 3" />
</menu>
```

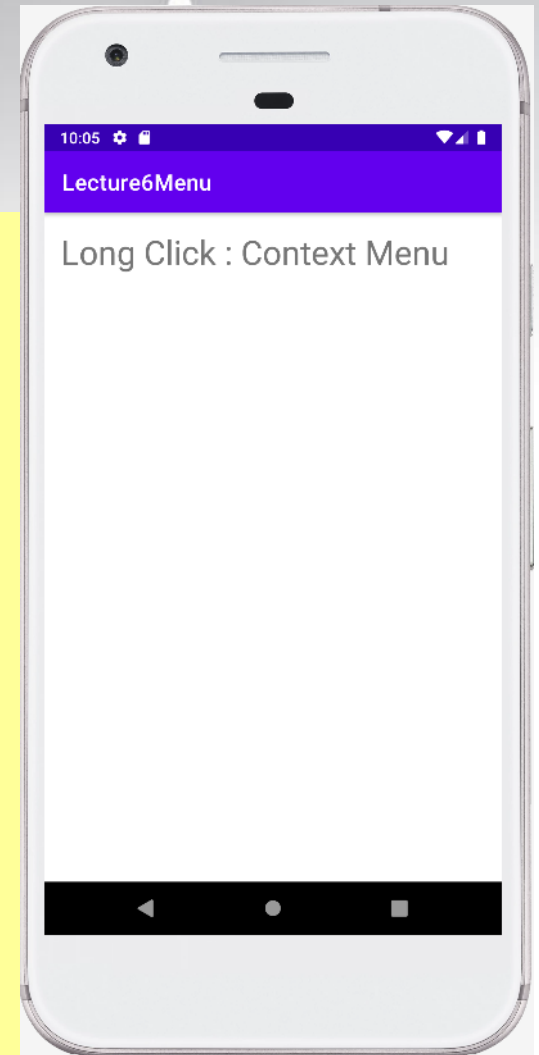




# Menu: Context Menu

- res>> layout>>activity\_main.xml

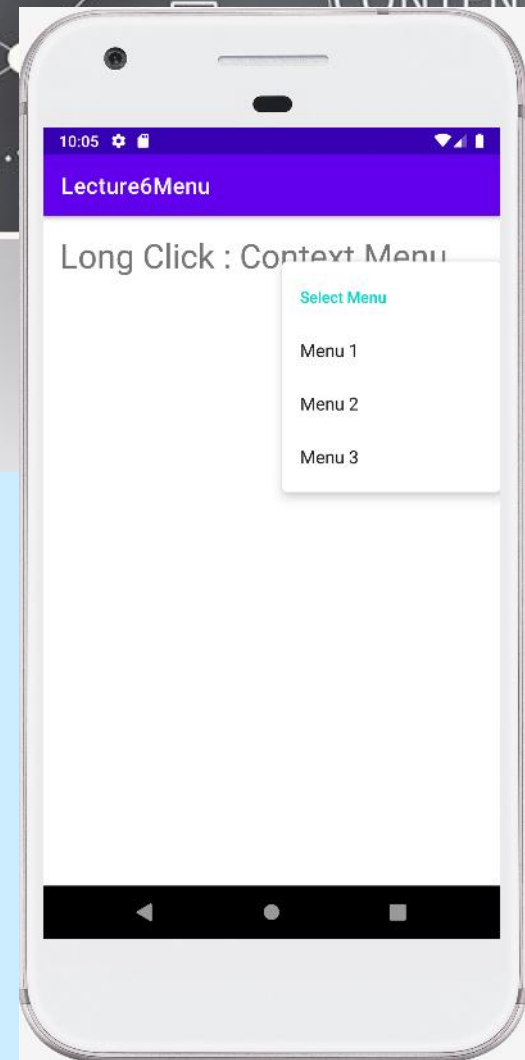
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="15dp"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Long Click : Context Menu"
        android:textSize="30sp"/>
</LinearLayout>
```



# Menu: Context Menu

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
  
        registerForContextMenu(binding.textView)  
    }  
  
    override fun onCreateContextMenu(menu: ContextMenu?, v: View?, menuInfo: ContextMenu.ContextMenuInfo?)  
    {  
        super.onCreateContextMenu(menu, v, menuInfo)  
        menuInflater.inflate(R.menu.my_menu, menu)  
        menu?.setHeaderTitle("Select Menu")  
    }  
}
```

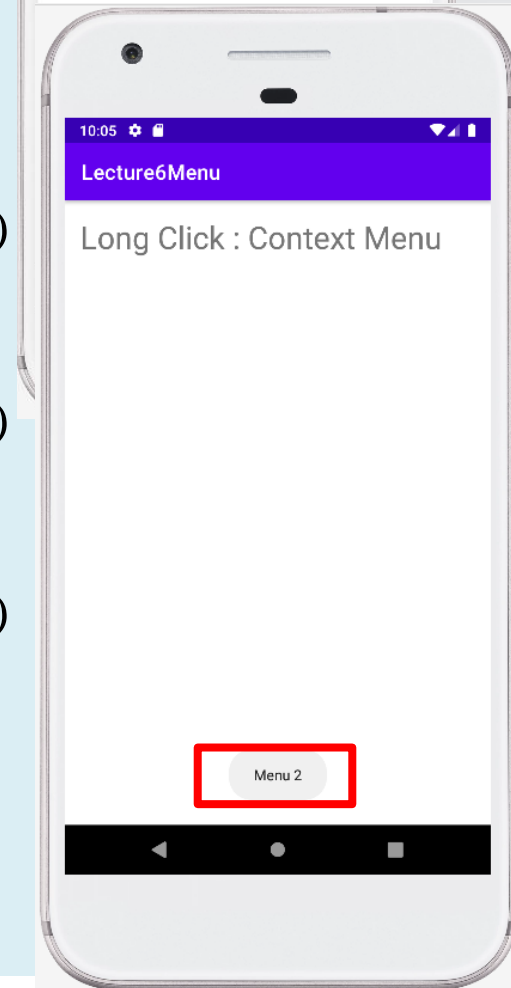
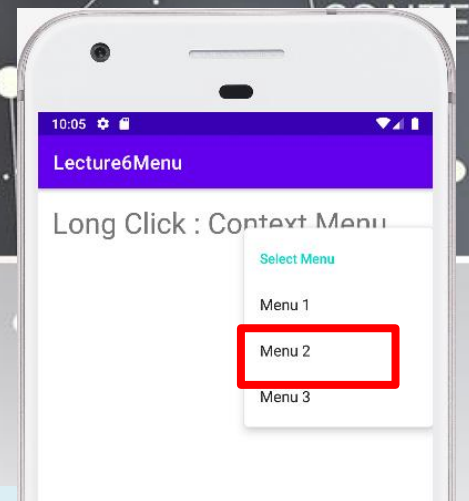
MainActivity.kt



# Menu: Context Menu

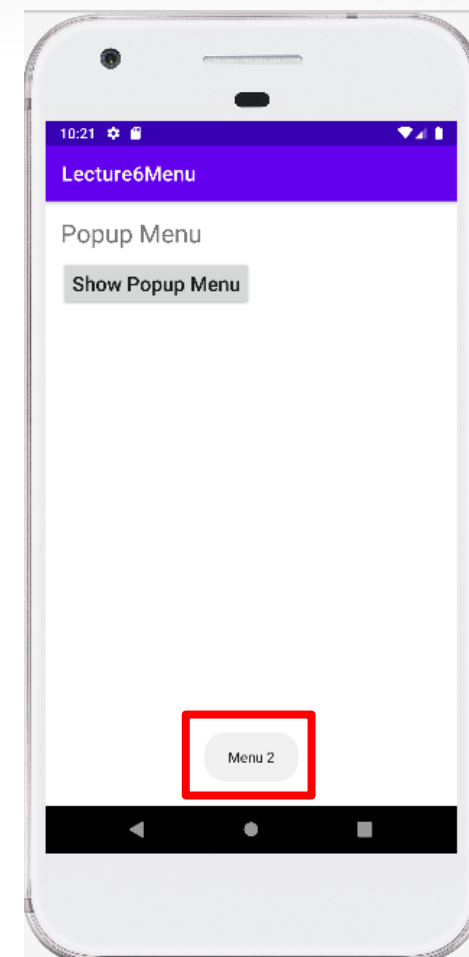
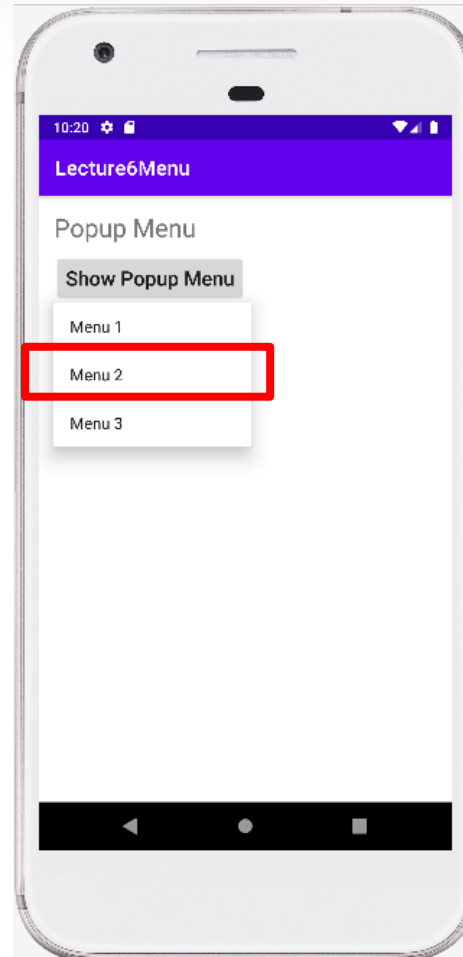
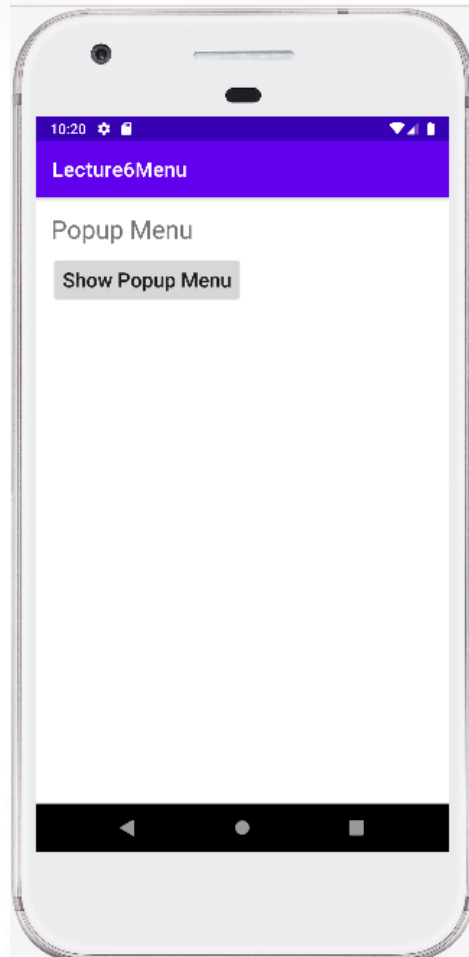
## MainActivity.kt (cont)

```
override fun onContextItemSelected(item: MenuItem): Boolean {  
    when (item.itemId) {  
        R.id.menu1 -> {  
            Toast.makeText(applicationContext, item.title, Toast.LENGTH_LONG).show()  
            return true  
        }  
        R.id.menu2 -> {  
            Toast.makeText(applicationContext, item.title, Toast.LENGTH_LONG).show()  
            return true  
        }  
        R.id.menu3 -> {  
            Toast.makeText(applicationContext, item.title, Toast.LENGTH_LONG).show()  
            return true  
        }  
        else -> return super.onOptionsItemSelected(item)  
    }  
}
```



# Menu: Popup menu

- Menu displays the menu below the anchor text if space is available otherwise above the anchor text.

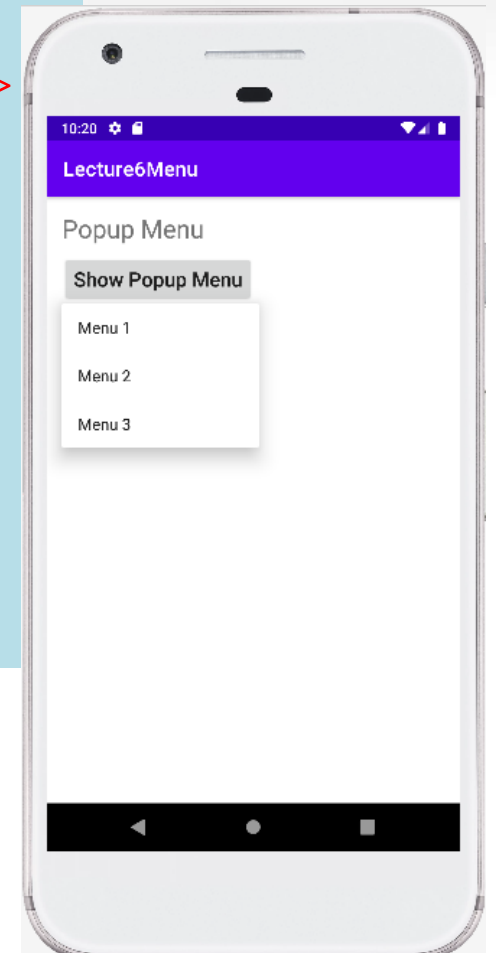




# Menu: Popup menu

- res>> menu>> my\_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/menu1"
        android:title="Menu 1"/>
    <item android:id="@+id/menu2"
        android:title="Menu 2"/>
    <item android:id="@+id/menu3"
        android:title="Menu 3" />
</menu>
```

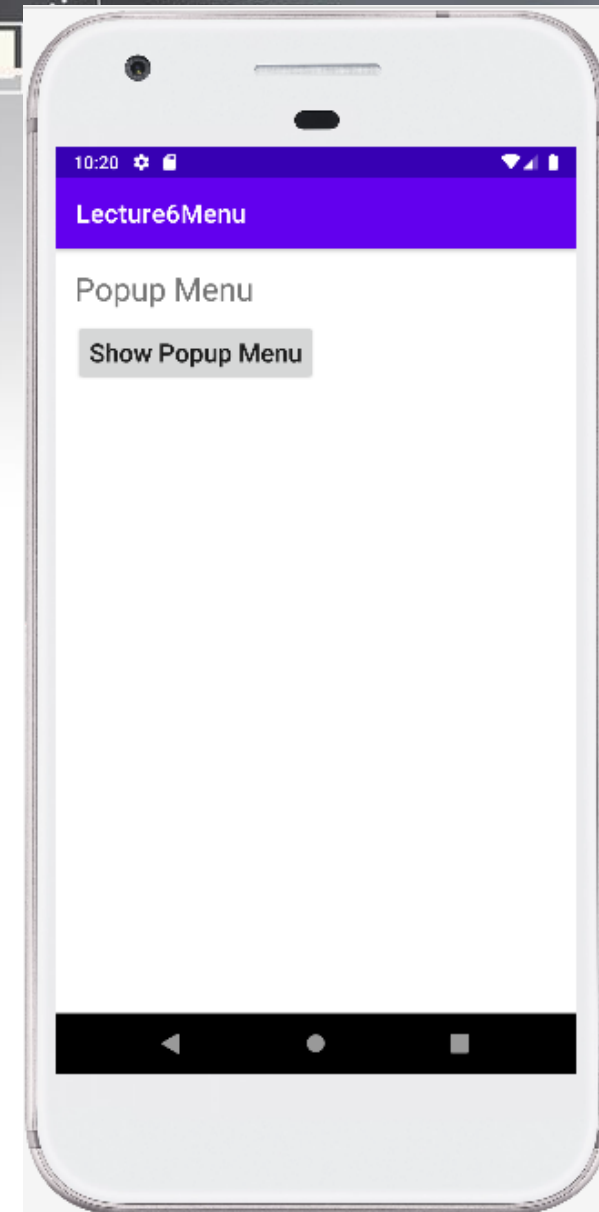


# Menu: Popup menu

res>> layout

>>activity\_main.xml

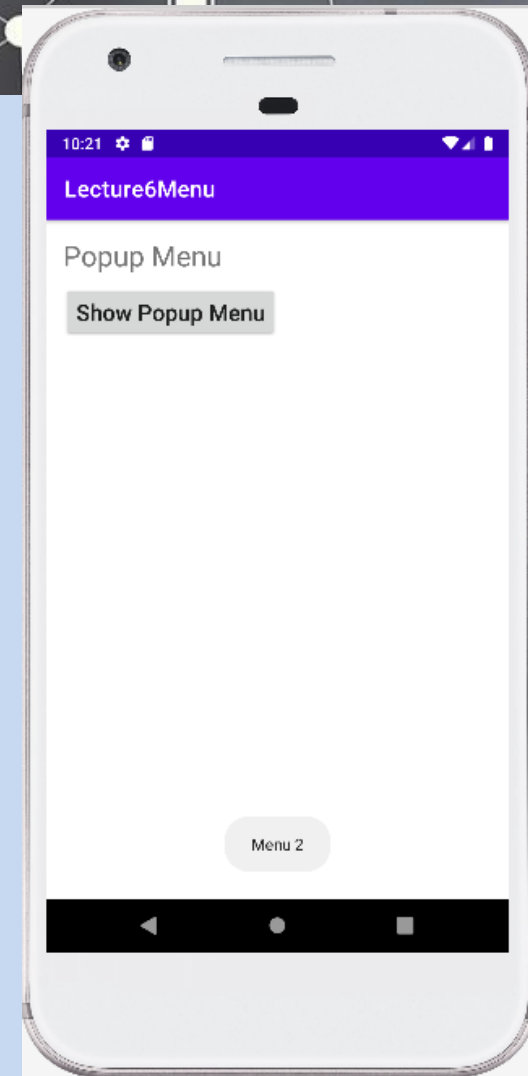
```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Popup Menu "
        android:textSize="25sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btn_popMenu"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Show Popup Menu"
        android:textSize="20sp"
        android:textAllCaps="false"
        android:layout_marginTop="10dp"
        android:onClick="showPopupMenu"/></LinearLayout>
```



# Menu: Popup menu

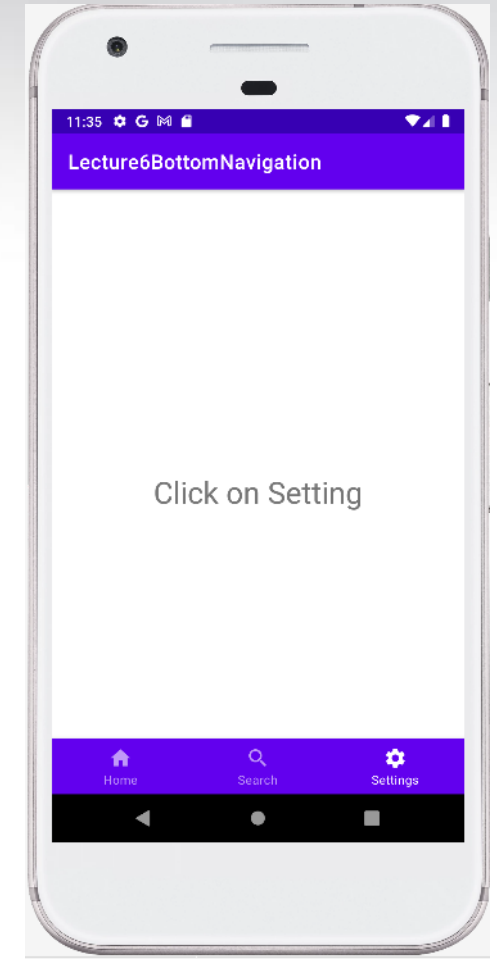
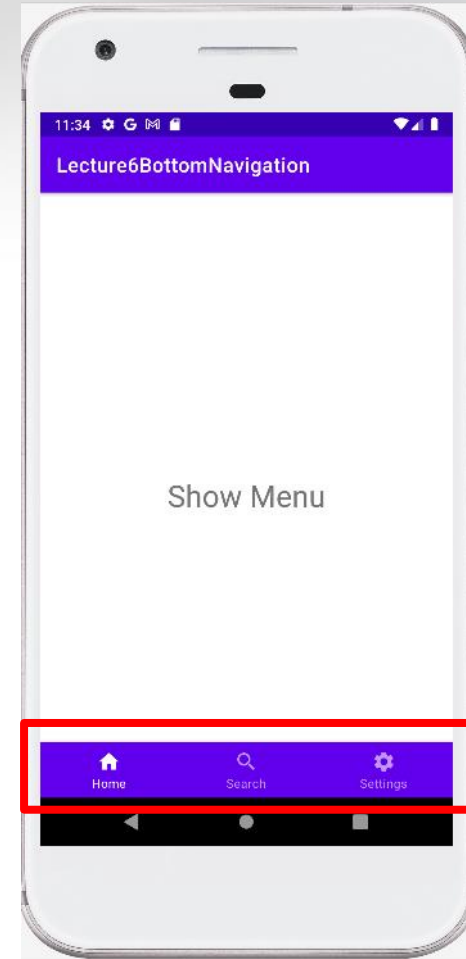
## MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
    }  
  
    fun showPopupMenu(v: View){  
        //Creating the instance of PopupMenu  
        val popup = PopupMenu(this, binding.btnPopupMenu)  
        //Inflating the Popup using xml file  
        popup.inflate(R.menu.my_menu)  
  
        popup.setOnMenuItemClickListener(PopupMenu.OnMenuItemClickListener {  
            item: MenuItem? ->  
            when (item!!.itemId) {  
                R.id.menu1 -> Toast.makeText(this, item.title, Toast.LENGTH_SHORT).show()  
                R.id.menu2 -> Toast.makeText(this, item.title, Toast.LENGTH_SHORT).show()  
                R.id.menu3 -> Toast.makeText(this, item.title, Toast.LENGTH_SHORT).show() }  
            true })  
        popup.show()  
    }  
}
```



# Bottom Navigation Bar

- BottomNavigationBar is a widget that displays a row of small widgets at the bottom of an app.
- Usually, show around three to five items.
- Each item must have a label and an icon.
- BottomNavigationBar allows to select one item at a time and quickly navigate to a given page.

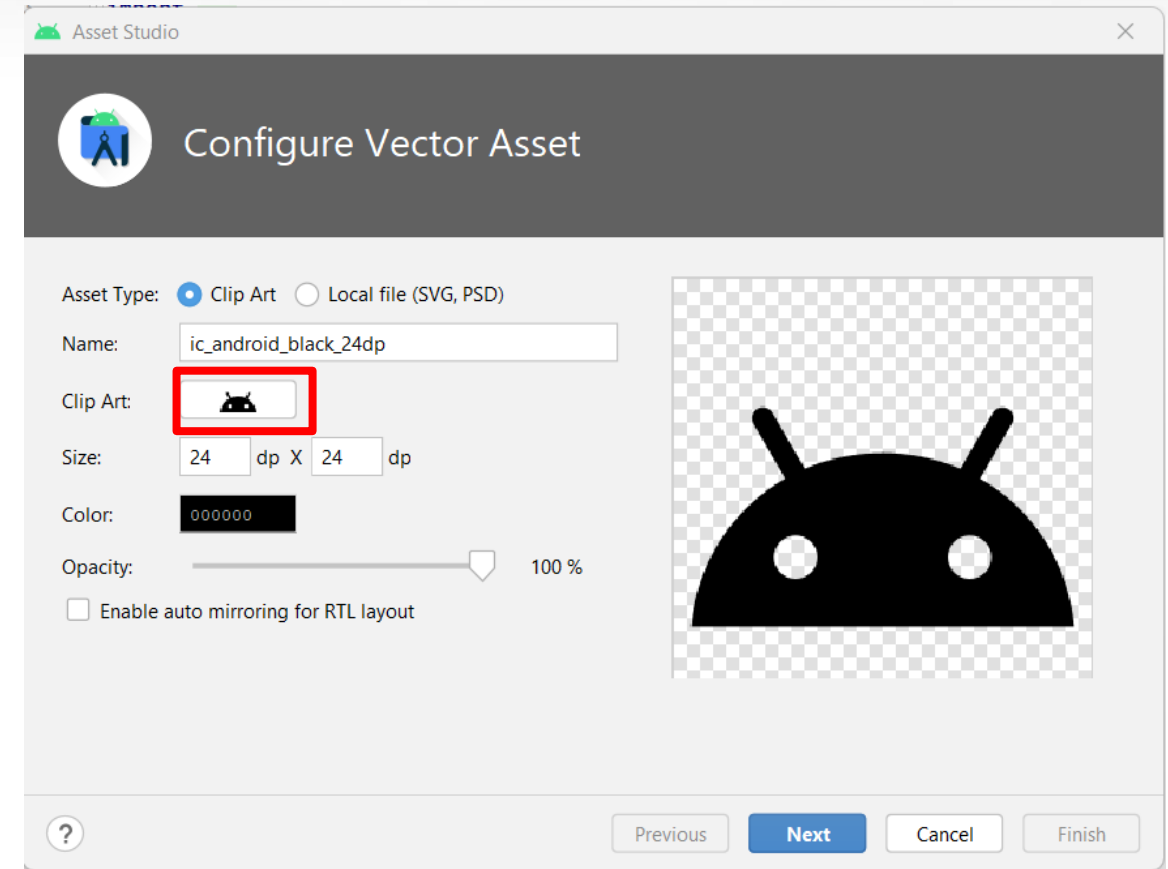
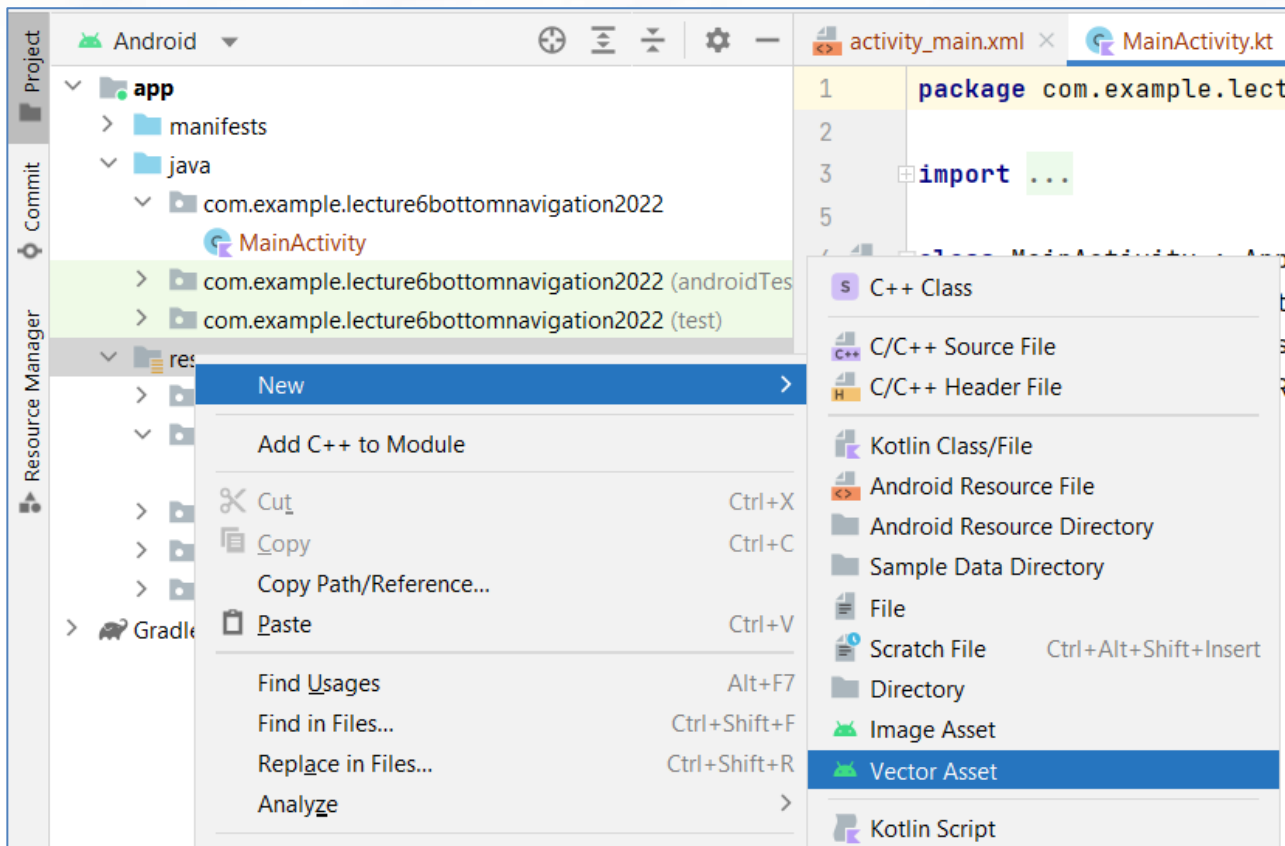




# Bottom Navigation Bar

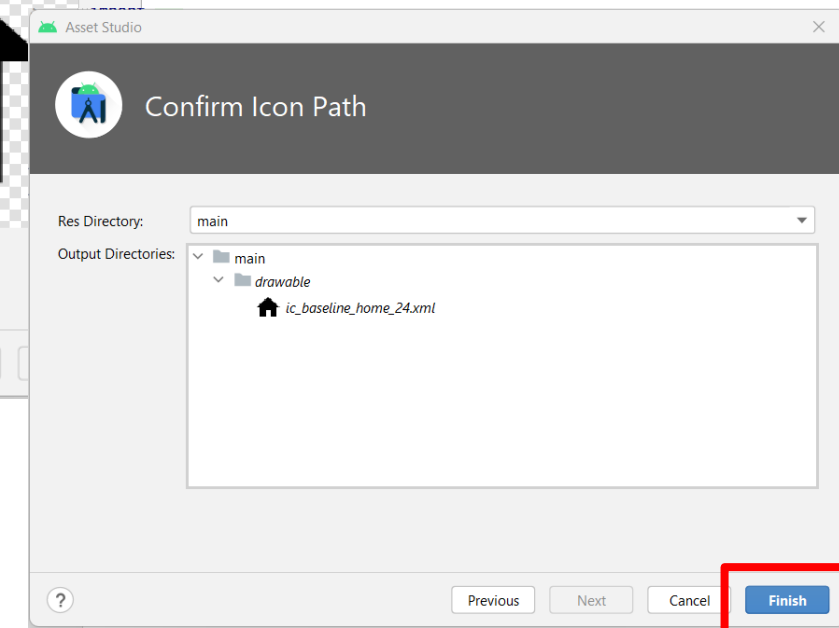
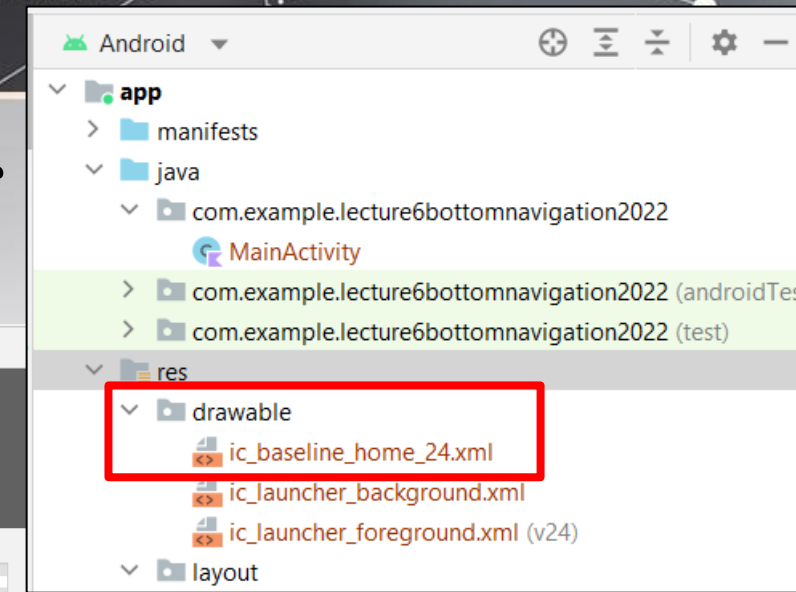
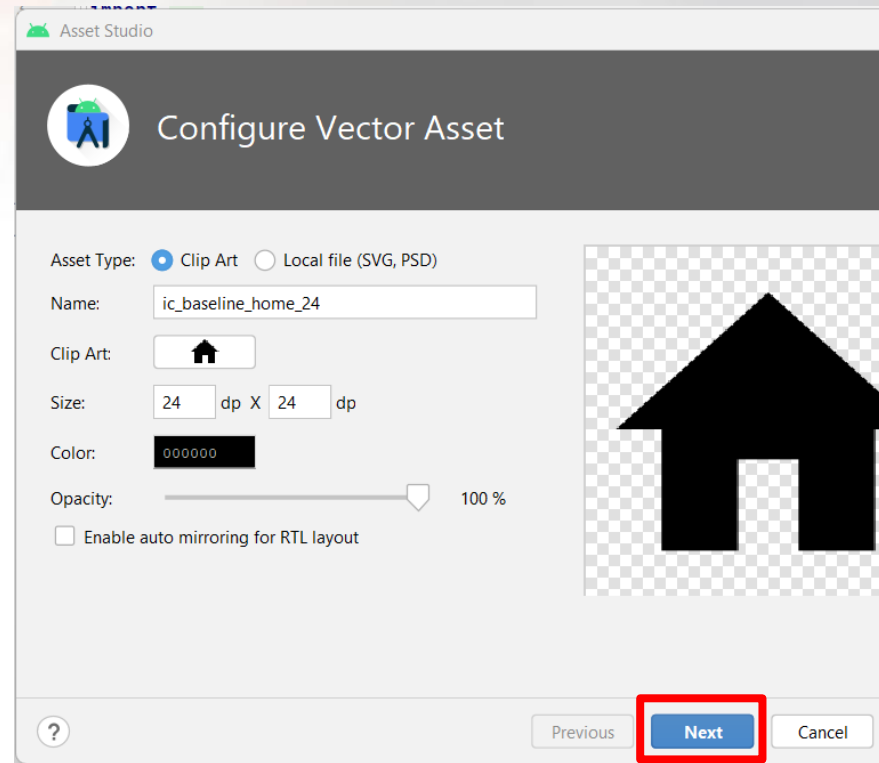
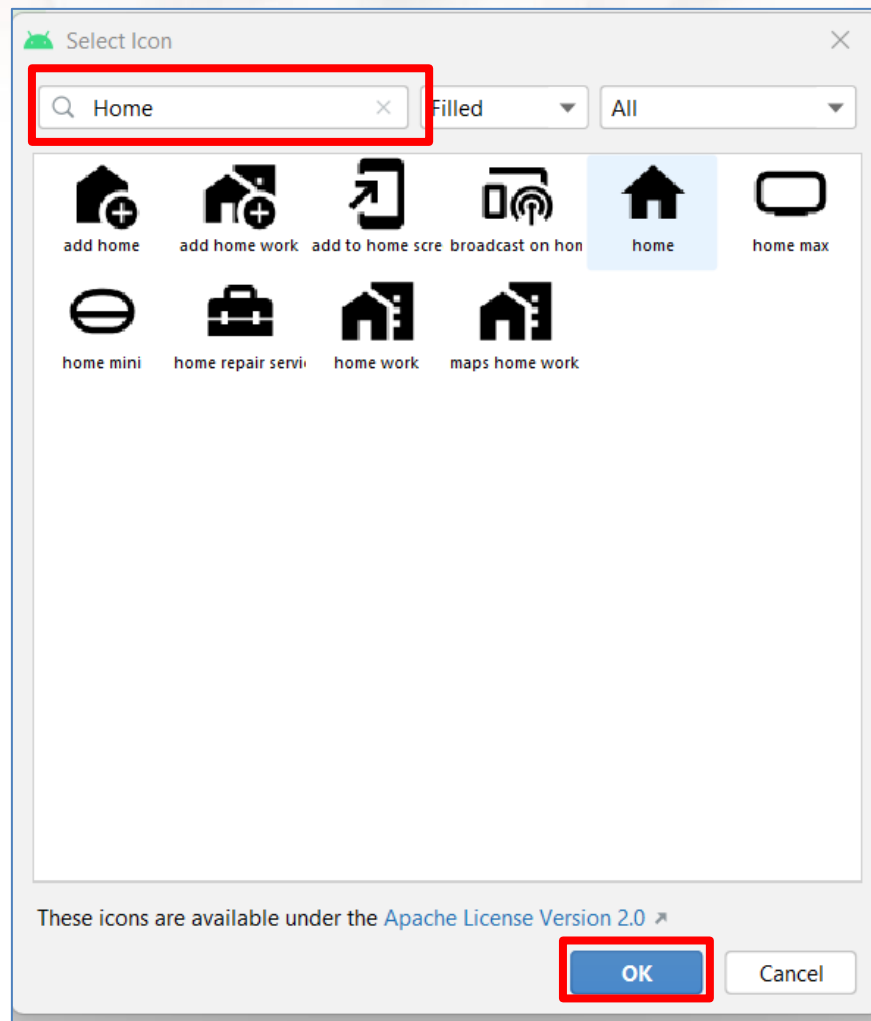
Creating a menu for the Bottom Navigation Bar

1. Add the icons



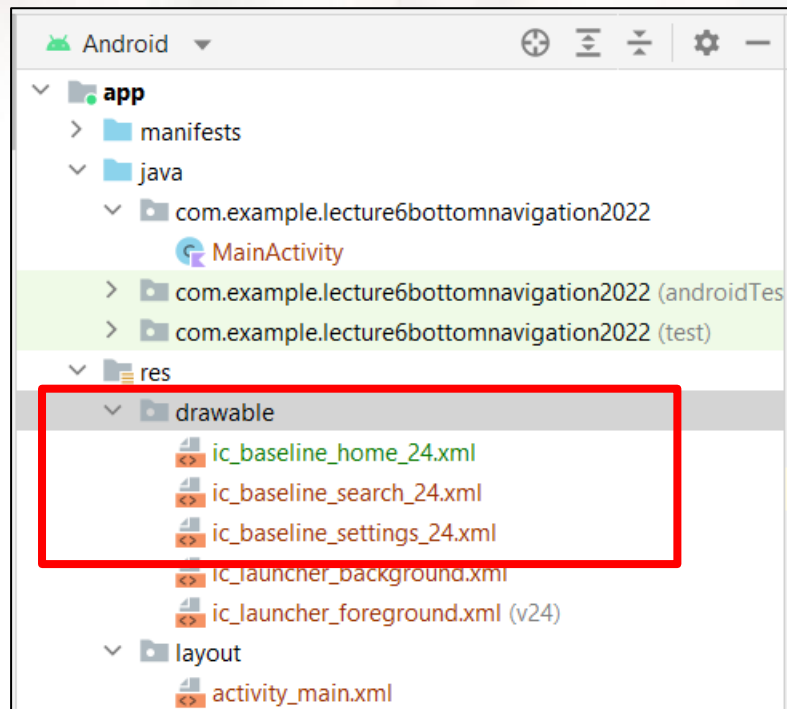
# Bottom Navigation Bar

## 1. Add the icons (cont)

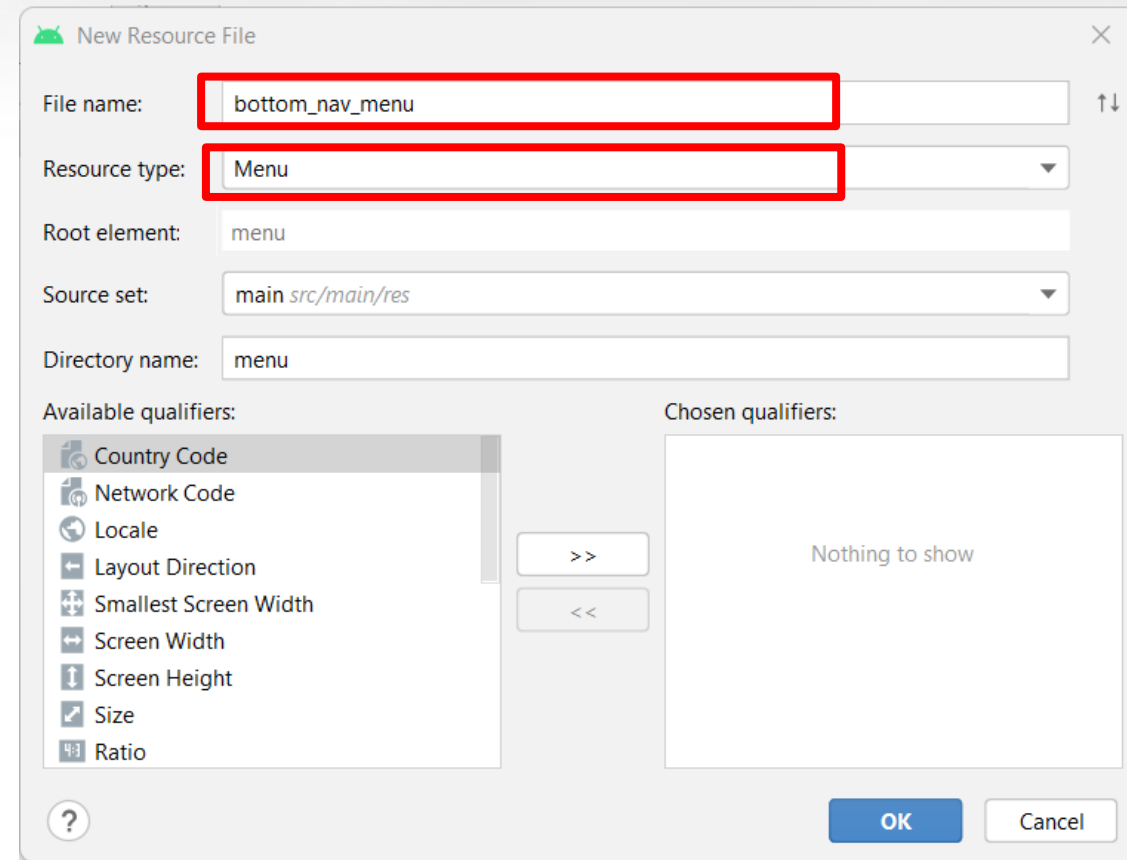


# Bottom Navigation Bar

## 1. Add the icons (cont)



## 2. Create Menu



# Bottom Navigation Bar

## 2. Create Menu (cont.)

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <menu xmlns:android="http://schemas.android.com/apk/res/android">
3   <item
4     android:id="@+id/home"
5     android:title="Home"
6     android:icon="@drawable/ic_baseline_home_24"/>
7   <item
8     android:id="@+id/search"
9     android:title="Search"
10    android:icon="@drawable/ic_baseline_search_24"/>
11  <item
12    android:id="@+id/settings"
13    android:title="Settings"
14    android:icon="@drawable/ic_baseline_settings_24"/>
15 </menu>
```



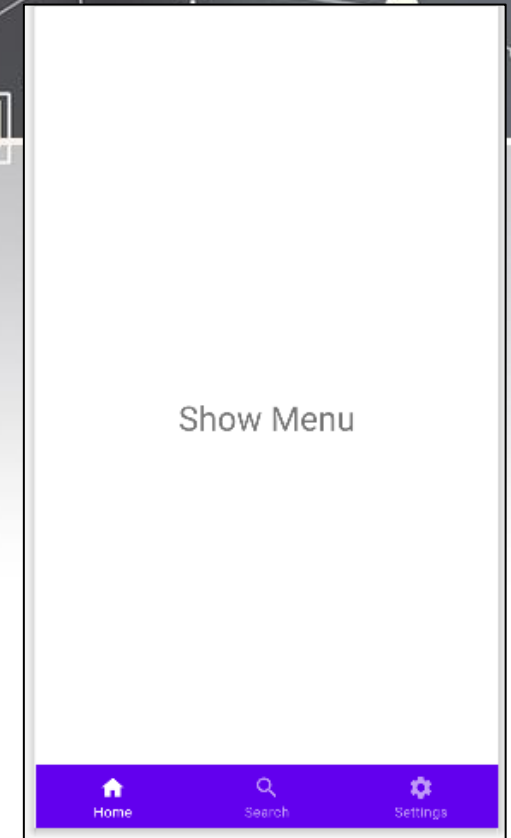
# Bottom Navigation Bar

activity\_main.xml

```
<?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">
```

```
<TextView
    android:id="@+id/txtShowMenu"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Show Menu"
    android:layout_centerInParent="true"
    android:textSize="30sp"/>
```

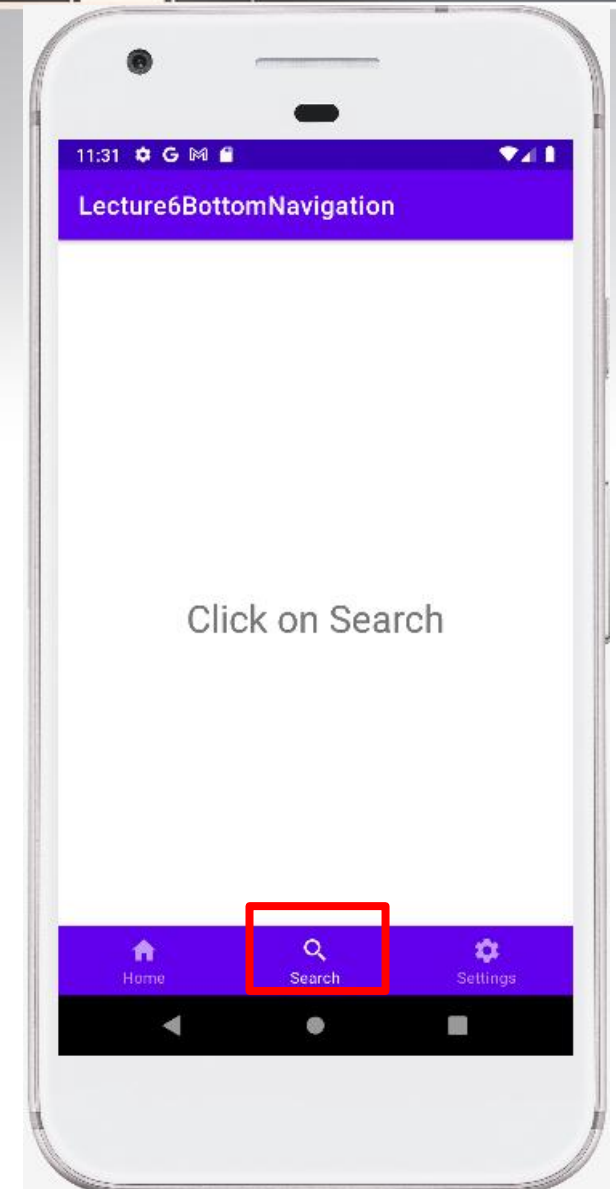
```
<com.google.android.material.bottomnavigation.BottomNavigationView
    android:id="@+id/bottom_navigation"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    style="@style/Widget.MaterialComponents.BottomNavigationView.Colored"
    app:menu="@menu/bottom_nav_menu" />
</RelativeLayout>
```



# Bottom Navigation Bar

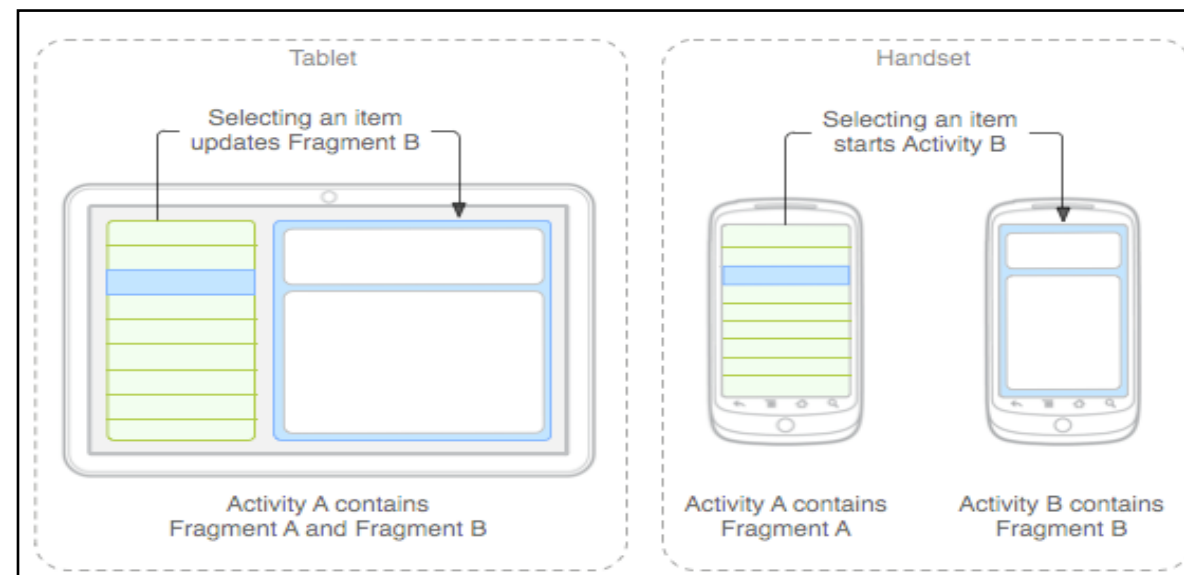
## MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
  
        // Click on Bottom Navigation Bar  
        binding.bottomNavigation.setOnItemSelectedListener {  
            when (it.itemId){  
                R.id.home    -> binding.txtShowMenu.text= "Click on Home"  
                R.id.search  -> binding.txtShowMenu.text= "Click on Search"  
                R.id.settings -> binding.txtShowMenu.text= "Click on Setting"  
            }  
            true  
        }  
    }  
}
```



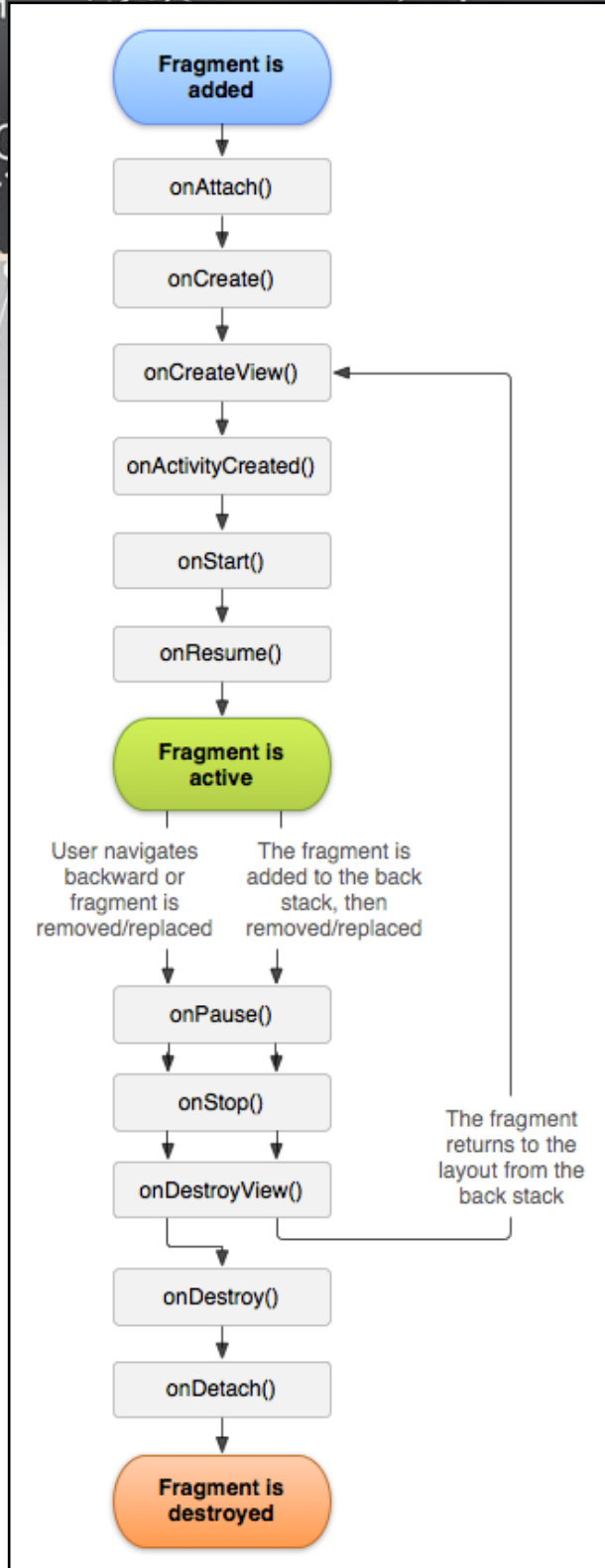
# Fragment

- An activity is a container for views
- When you have a larger screen device than a phone –like a tablet it can look too simple to use phone interface here.
- Fragments
  - Mini-activities, each with its own set of views
  - One or more fragments can be embedded in an Activity
  - You can do this dynamically as a function of the device type (tablet or not) or orientation



# Fragment Lifecycle

- Fragment in an Activity---Activity Lifecycle influences
  - Activity paused → all its fragments paused
  - Activity destroyed → all its fragments paused
  - Activity running → manipulate each fragment independently.
- Fragment transaction → add, remove, replace, etc.
  - adds it to a **back stack** that's managed by the activity—each back stack entry in the activity is a record of the fragment transaction that occurred.
  - The back stack allows the user to reverse a fragment transaction (navigate backwards), by pressing the **Back button**.







## Fragment methods (callback functions)

- **onAttach(Activity)** called once the fragment is associated with its activity.
- **onCreate(Bundle)** called to do initial creation of the fragment.
- **onCreateView(LayoutInflater, ViewGroup, Bundle)** creates and returns the view hierarchy associated with the fragment.
- **onActivityCreated(Bundle)** tells the fragment that its activity has completed its own **Activity.onCreate**.
- **onStart()** makes the fragment visible to the user (based on its containing activity being started).
- **onResume()** makes the fragment interacting with the user (based on its containing activity being resumed).



## Fragment methods (callback functions)

As a fragment is no longer being used, it goes through a reverse series of callbacks:

- **onPause()** fragment is no longer interacting with the user either because its activity is being paused or a fragment operation is modifying it in the activity.
- **onStop()** fragment is no longer visible to the user either because its activity is being stopped or a fragment operation is modifying it in the activity.
- **onDestroyView()** allows the fragment to clean up resources associated with its View.
- **onDestroy()** called to do final cleanup of the fragment's state.
- **onDetach()** called immediately prior to the fragment no longer being associated with its activity.

# Fragments and their UI – onCreateView() using XML

- Can implement onCreateView using XML

Activity parent's ViewGroup



```
class ExampleFragment : Fragment {
```

```
    override fun onCreateView( inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle? ): View? {
```

**Bundle** that provides data about the previous instance of the fragment, if the fragment is being resumed

```
// Inflate the layout for this fragment
```

```
    return inflater.inflate(R.layout.example_fragment, container, false);
```

```
}
```

```
}
```

*example\_fragment.xml* file that contains the layout  
This will be contained in resource layout folder.

The background features a pair of hands holding a smartphone. Overlaid on the image is a network diagram with various icons and labels. The labels include 'MONITORING', 'RESOURCE', 'SEARCH', 'CONTENT', and 'WEBSITE'. The diagram consists of nodes connected by lines, with some nodes highlighted in orange. A hand is pointing at one of the orange nodes.

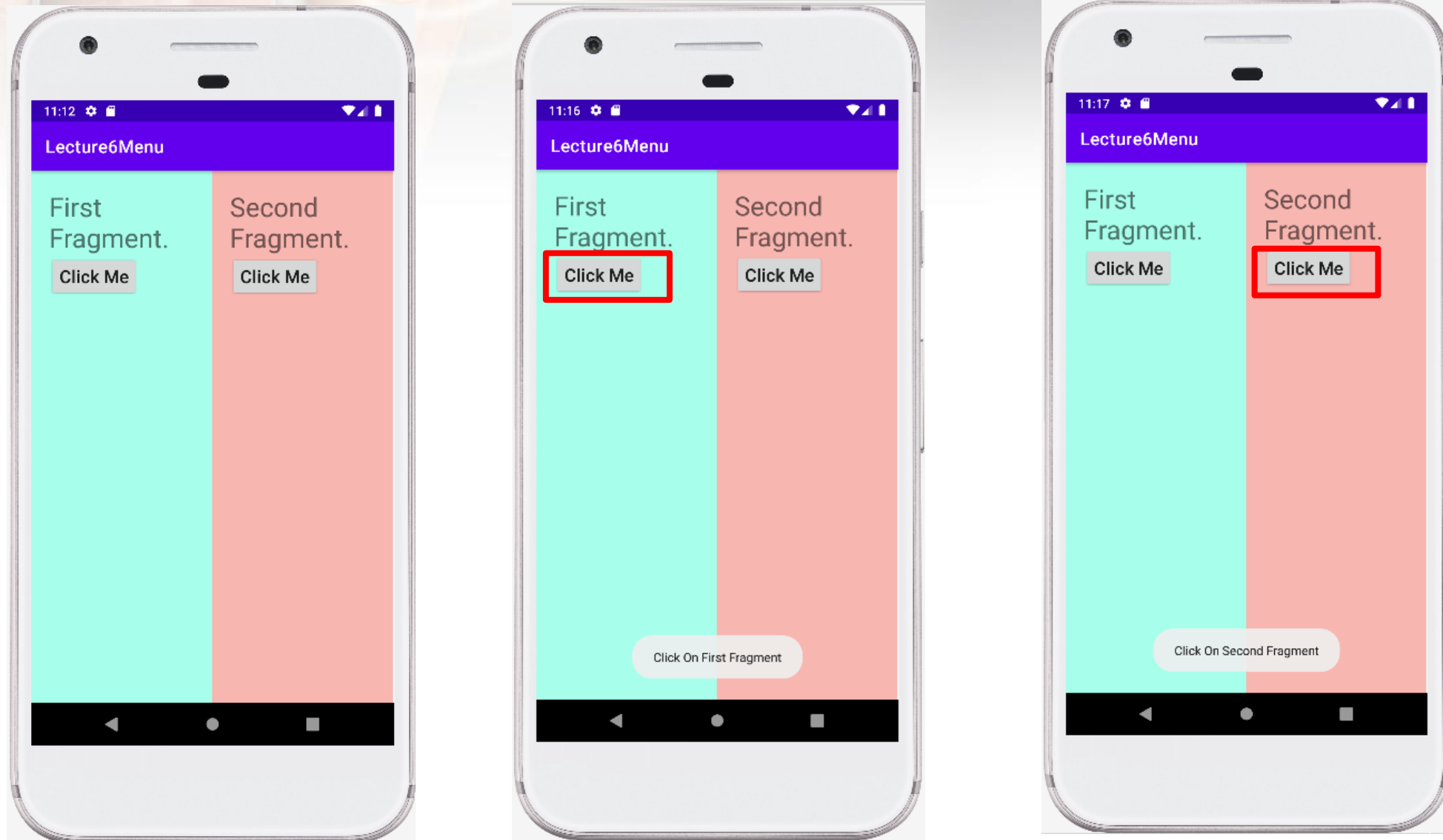
# Adding Fragment

There are 2 options for adding fragment.

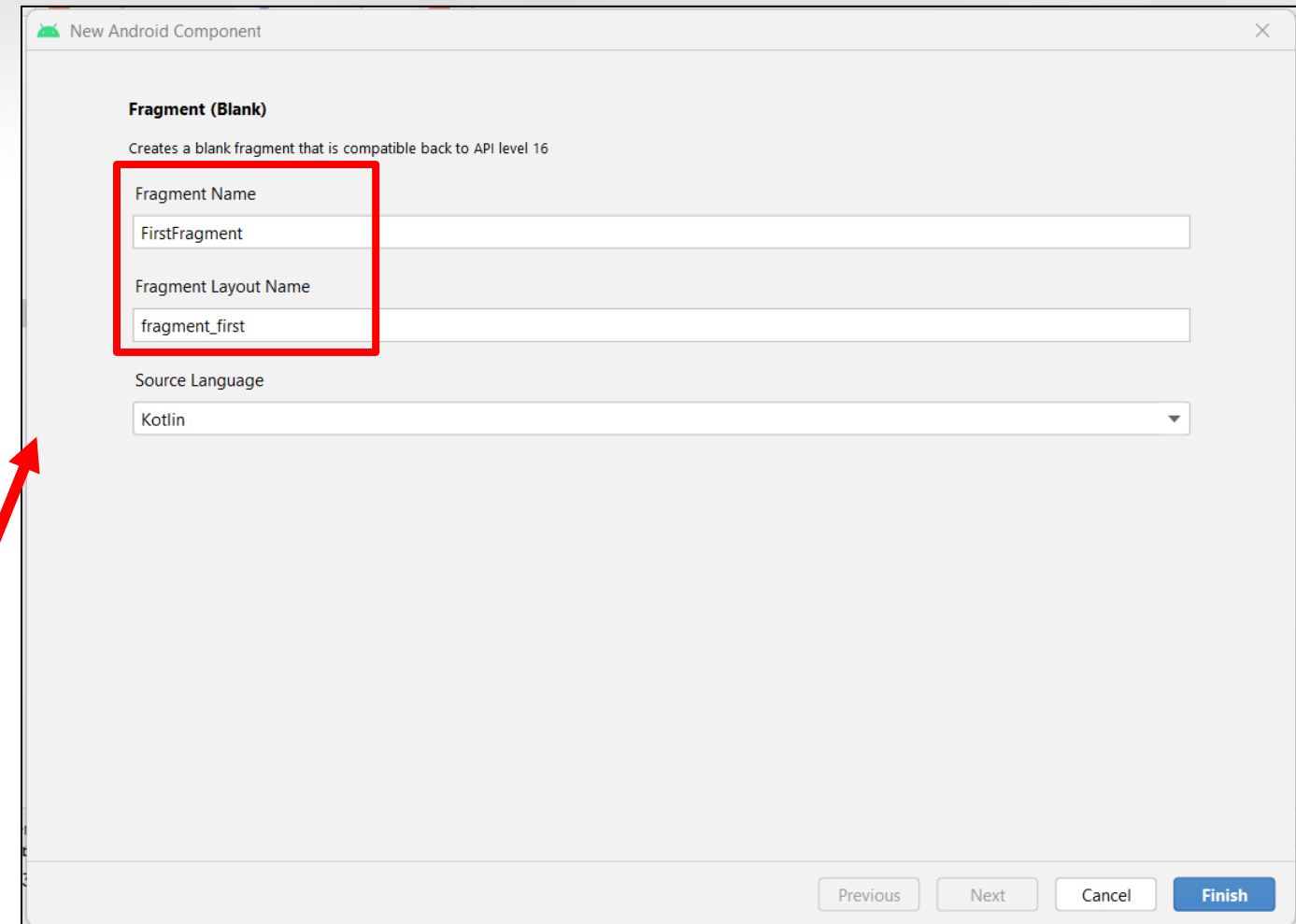
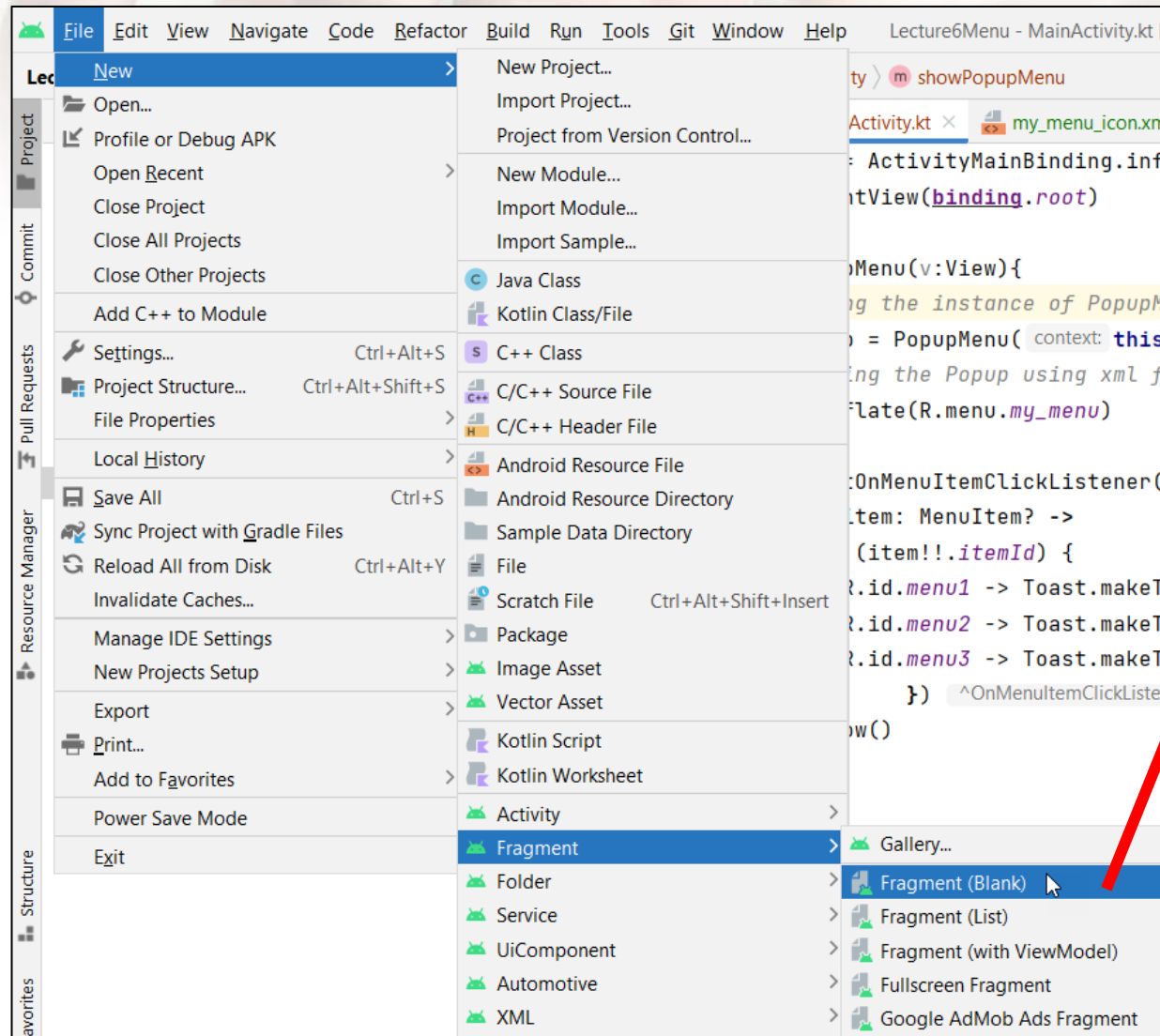
1. Adding to an Activity via layout XML.
2. Creating and adding to an Activity via CODE.



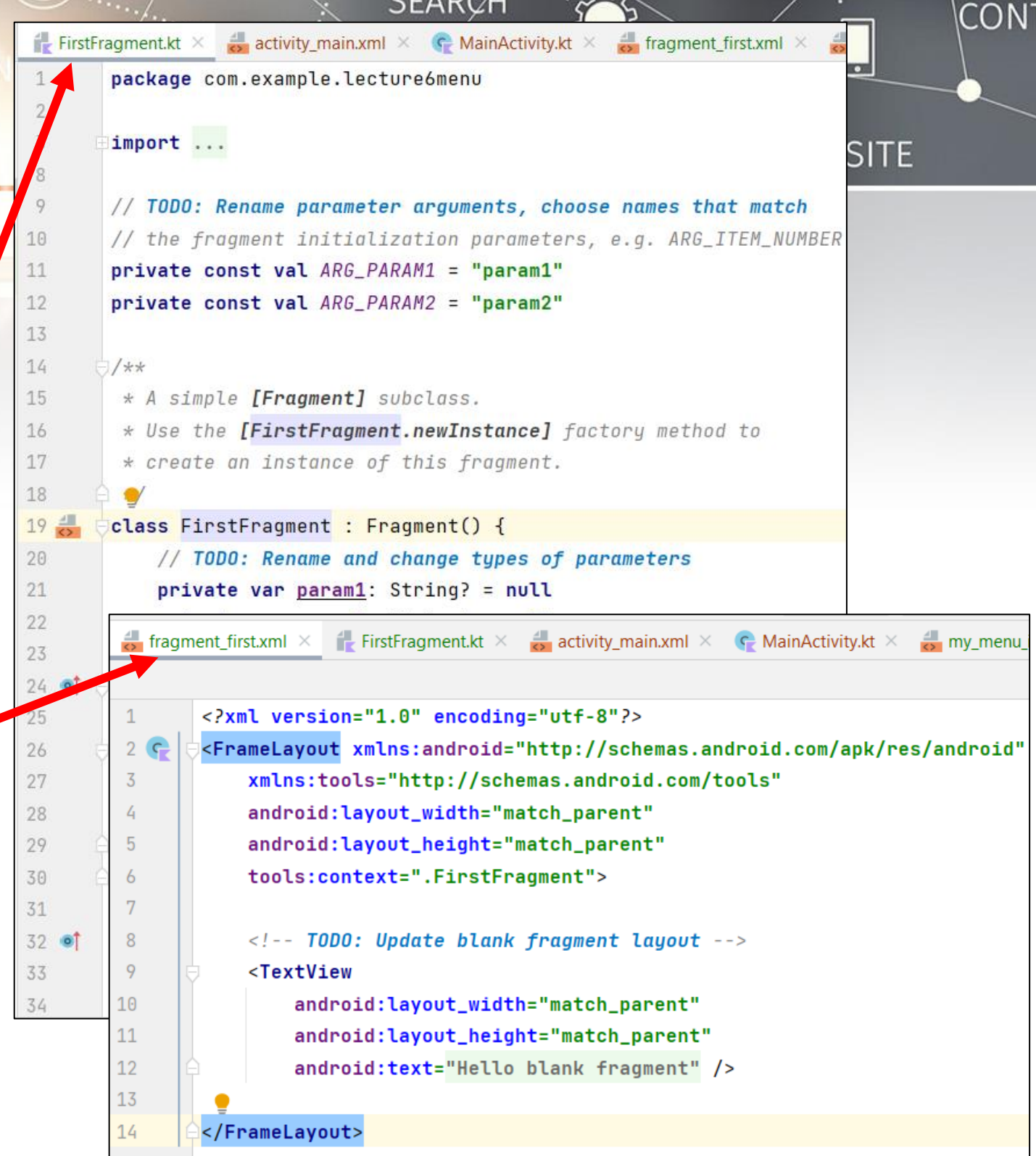
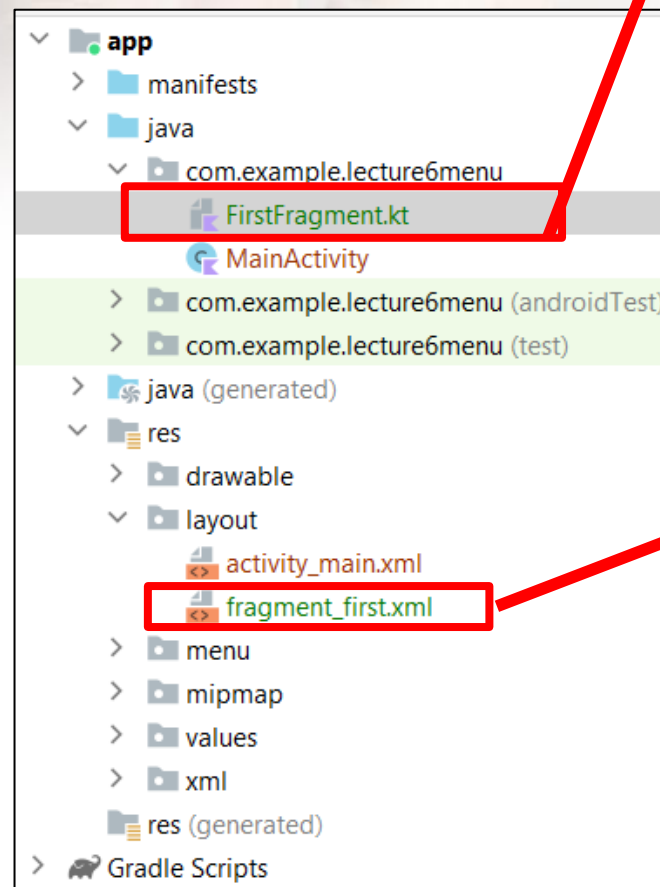
## OPTION 1: Adding to an Activity via layout XML.



# OPTION 1: Adding to an Activity via layout XML



## OPTION 1: Adding to an Activity via layout XML.





## OPTION 1 –adding to an Activity via layout XML.

New Android Component

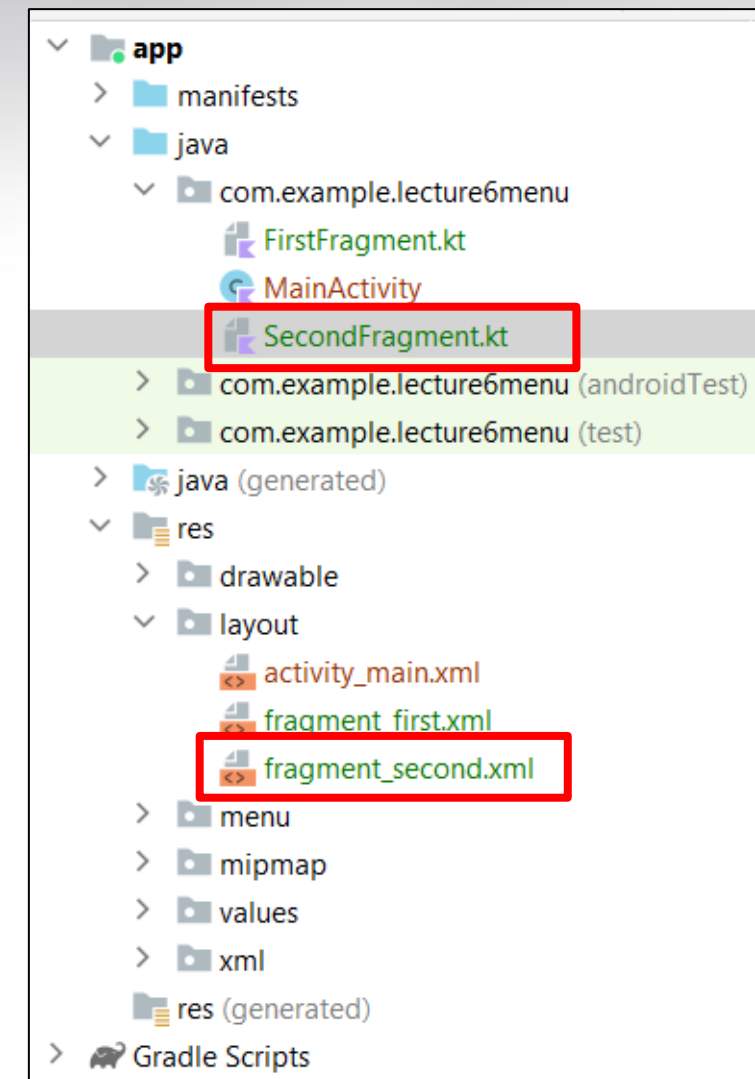
**Fragment (Blank)**  
Creates a blank fragment that is compatible back to API level 16

Fragment Name  
SecondFragment

Fragment Layout Name  
fragment\_second

Source Language  
Kotlin

Previous Next Cancel Finish





# OPTION 1: Adding to an Activity via layout XML.

## fragment\_first.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    android:background="#A7FFEB"
    android:padding="20dp"
    tools:context=".FirstFragment">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="First Fragment."
        android:textSize="30sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnClickFirstFrag"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAllCaps="false"
        android:textSize="20sp"
        android:text="Click Me" /></LinearLayout>
```

First Fragment.

Click Me

## fragment\_second.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    android:background="#F8B6B1"
    android:padding="20dp"
    tools:context=".SecondFragment">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second Fragment."
        android:textSize="30sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnClickSecondFrag"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAllCaps="false"
        android:textSize="20sp"
        android:text="Click Me" /></LinearLayout>
```

Second Fragment.

Click Me

## OPTION 1: Adding to an Activity via layout XML.

### FirstFragment.kt

```
1 package com.example.lecture6menu
2
3 import ...
4
5 // TODO: Rename parameter arguments, choose names that match
6 // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
7 private const val ARG_PARAM1 = "param1"
8 private const val ARG_PARAM2 = "param2"
9
10 /**
11  * A simple [Fragment] subclass.
12  * Use the [FirstFragment.newInstance] factory method to
13  * create an instance of this fragment.
14  */
15 class FirstFragment : Fragment() {
16     // TODO: Rename and change types of parameters
17     private var param1: String? = null
18     private var param2: String? = null
19
20     override fun onCreate(savedInstanceState: Bundle?) {
21         super.onCreate(savedInstanceState)
22         arguments?.let { it: Bundle
23             param1 = it.getString(ARG_PARAM1)
24             param2 = it.getString(ARG_PARAM2)
25         }
26     }
27
28     override fun onCreateView(
29         inflater: LayoutInflater, container: ViewGroup?,
30         savedInstanceState: Bundle?
```



```
1 package com.example.lecture6menu
2
3 import ...
4
5 class FirstFragment : Fragment() {
6     override fun onCreateView(
7         inflater: LayoutInflater, container: ViewGroup?,
8         savedInstanceState: Bundle?
9     ): View? {
10         // Inflate the layout for this fragment
11         return inflater.inflate(R.layout.fragment_first, container, attachToRoot: false)
12     }
13 }
```

## OPTION 1: Adding to an Activity via layout XML.

- FirstFragment.kt

```
class FirstFragment : Fragment() {  
    private lateinit var bindingFirst : FragmentFirstBinding  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        // Inflate the layout for this fragment  
        bindingFirst = FragmentFirstBinding.inflate(inflater)  
        bindingFirst.btnClickFirstFrag.setOnClickListener(){  
            val toast = Toast.makeText(context, "Click On First Fragment", Toast.LENGTH_SHORT)  
            toast.show()  
        }  
        return bindingFirst.root  
    }  
}
```

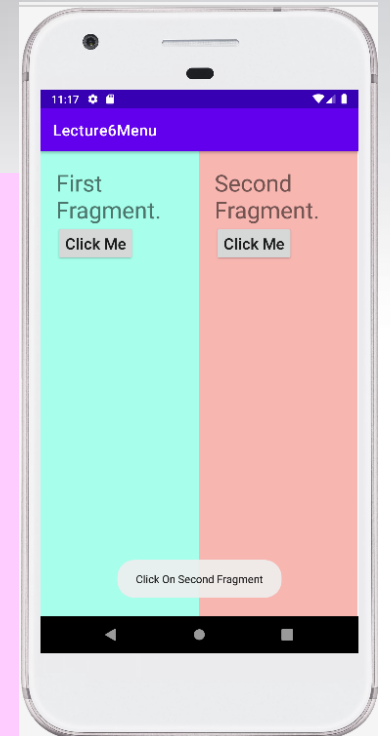




# OPTION 1: Adding to an Activity via Activity layout XML.

- SecondFragment.kt

```
class SecondFragment : Fragment() {  
    private lateinit var bindingSecond : FragmentSecondBinding  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        // Inflate the layout for this fragment  
        bindingSecond = FragmentSecondBinding.inflate(inflater)  
        bindingSecond.btnClickSecondFrag.setOnClickListener(){  
            val toast = Toast.makeText(context, "Click On Second Fragment", Toast.LENGTH_SHORT)  
            toast.show()  
        }  
        return bindingSecond.root  
    }  
}
```





## OPTION 1: Adding to an Activity via layout XML.

activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    tools:context=".MainActivity">

    <fragment
        android:id="@+id/first_fragment"
        android:name="com.myweb.lac6menufragment.FirstFragment"
        android:layout_width="200dp"
        android:layout_height="match_parent"/>

    <fragment
        android:id="@+id/second_fragment"
        android:name="com.myweb.lac6menufragment.SecondFragment"
        android:layout_width="200dp"
        android:layout_height="match_parent"/>
</LinearLayout>
```



## OPTION 1: Adding to an Activity via layout XML.

- MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
    }  
}
```





## OPTION 2: Creating and adding to an Activity via CODE.

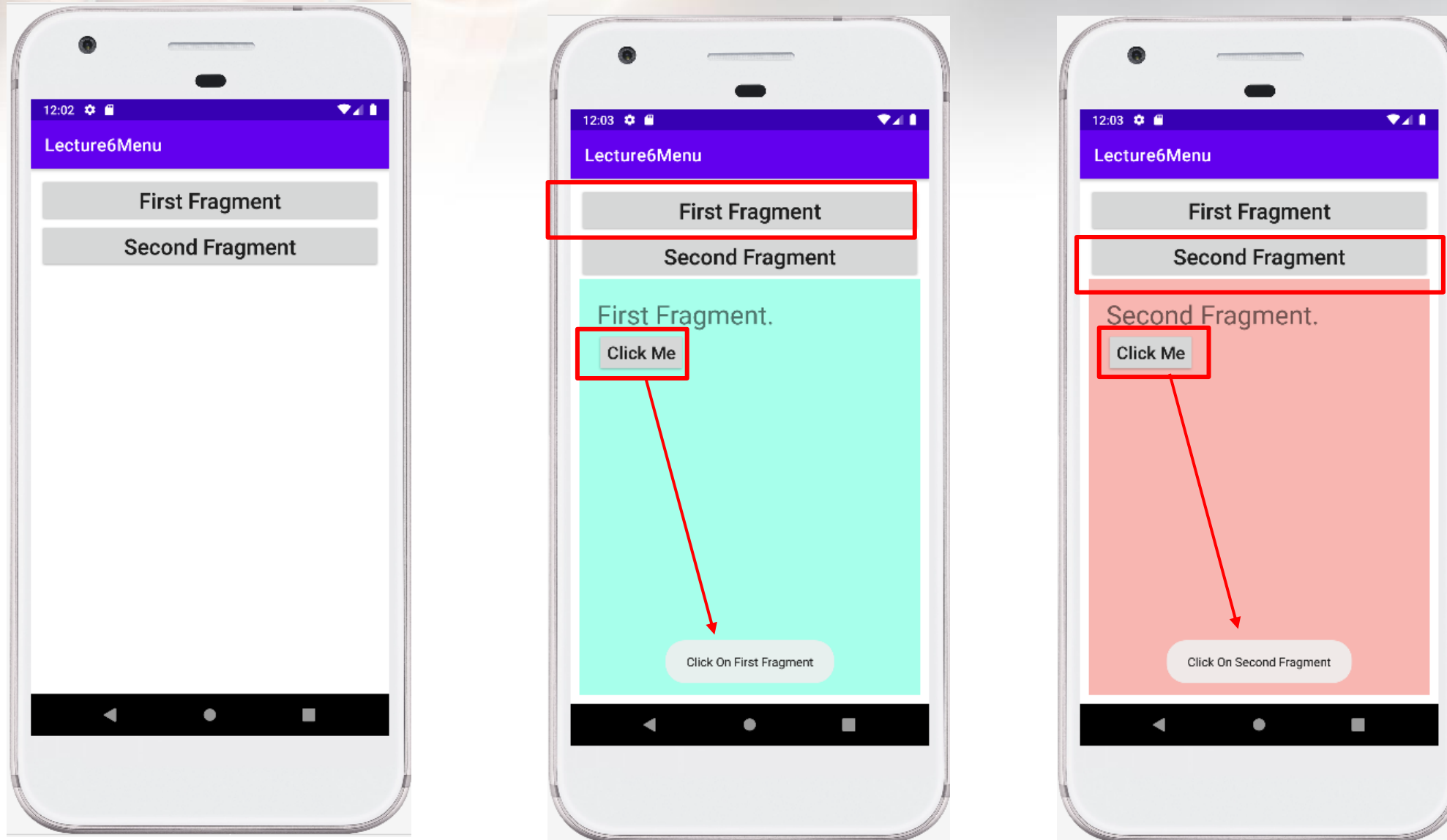
- Inside Activity Code where you want to add Fragment (dynamically anywhere or in `onCreate()` callback)
- Get FragmentTransaction associated with this Activity

```
val fragmentManager : FragmentManager = supportFragmentManager()
val fragmentTransaction : FragmentTransaction = fragmentManager.beginTransaction()
```
- Create instance of Fragment

```
val fragment = ExampleFragment()
```
- Add Fragment instance to Activity (`add` or `replace method`)

```
fragmentTransaction.add(R.id.fragment_container, fragment)
fragmentTransaction.commit()
```

## OPTION 2: Creating and adding to an Activity via CODE.



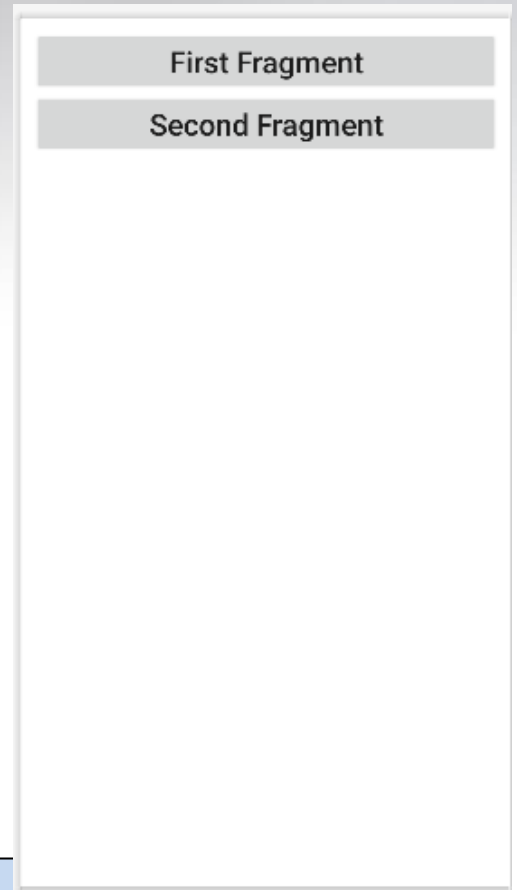


## OPTION 2: Creating and adding to an Activity via CODE.

activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout ....
    android:orientation="vertical"
    android:padding="10dp"
    tools:context=".MainActivity">
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnFirstFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="First Fragment"
        android:textSize="25sp"
        android:textAllCaps="false"
        android:onClick="clickFirstFragment"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnSecondFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Second Fragment"
        android:textSize="25sp"
        android:textAllCaps="false"
        android:onClick="clickSecondFragment"/>
```

```
<FrameLayout
    android:id="@+id/frameLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
</LinearLayout>
```



## OPTION 2: Creating and adding to an Activity via CODE.

### fragment\_first.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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"
    android:orientation="vertical"
    android:background="#A7FFEB"
    android:padding="20dp"
    tools:context=".FirstFragment">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="First Fragment."
        android:textSize="30sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnClickFirstFrag"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAllCaps="false"
        android:textSize="20sp"
        android:text="Click Me" /></LinearLayout>
```

First Fragment.

Click Me

### fragment\_second.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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"
    android:orientation="vertical"
    android:background="#F8B6B1"
    android:padding="20dp"
    tools:context=".SecondFragment">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second Fragment."
        android:textSize="30sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnClickSecondFrag"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAllCaps="false"
        android:textSize="20sp"
        android:text="Click Me" /></LinearLayout>
```

Second Fragment.

Click Me

## OPTION 2: Creating and adding to an Activity via CODE.

- FirstFragment.kt

```
class FirstFragment : Fragment() {  
    private lateinit var bindingFirst : FragmentFirstBinding  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        // Inflate the layout for this fragment  
        bindingFirst = FragmentFirstBinding.inflate(inflater)  
        bindingFirst.btnClickFirstFrag.setOnClickListener(){  
            val toast = Toast.makeText(context, "Click On First Fragment", Toast.LENGTH_SHORT)  
            toast.show()  
        }  
        return bindingFirst.root  
    }  
}
```

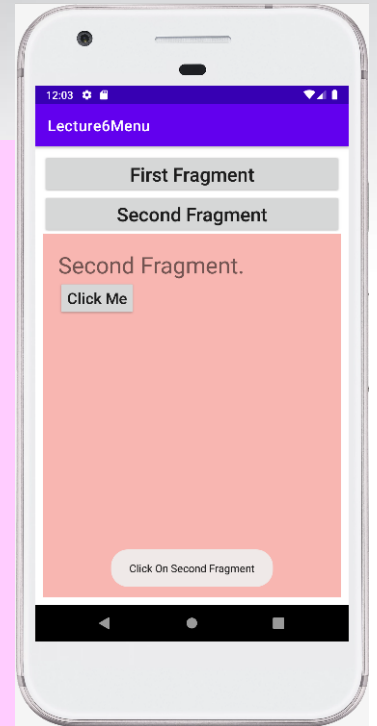




## OPTION 2: Creating and adding to an Activity via CODE.

- SecondFragment.kt

```
class SecondFragment : Fragment() {  
    private lateinit var bindingSecond : FragmentSecondBinding  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        // Inflate the layout for this fragment  
        bindingSecond = FragmentSecondBinding.inflate(inflater)  
        bindingSecond.btnClickSecondFrag.setOnClickListener(){  
            val toast = Toast.makeText(context, "Click On Second Fragment", Toast.LENGTH_SHORT)  
            toast.show()  
        }  
        return bindingSecond.root  
    }  
}
```

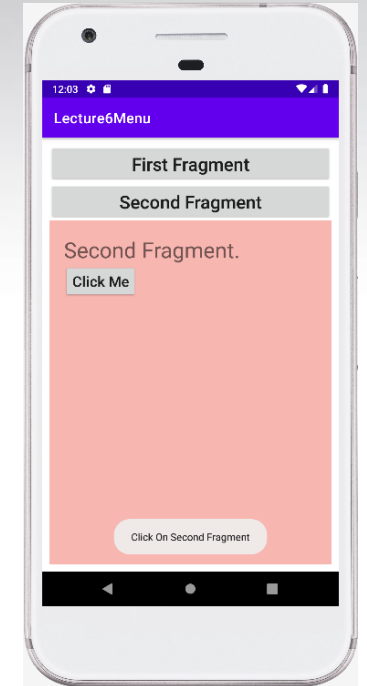
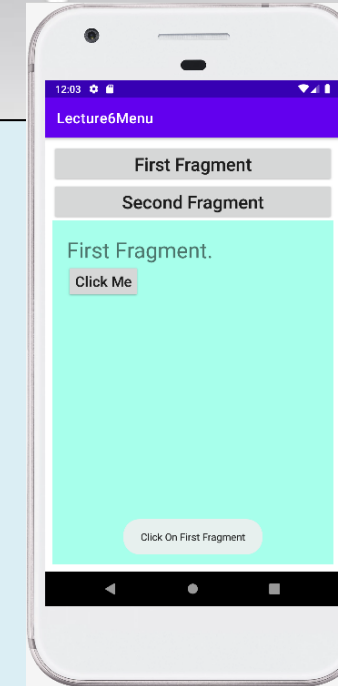




## OPTION 2: Creating and adding to an Activity via CODE.

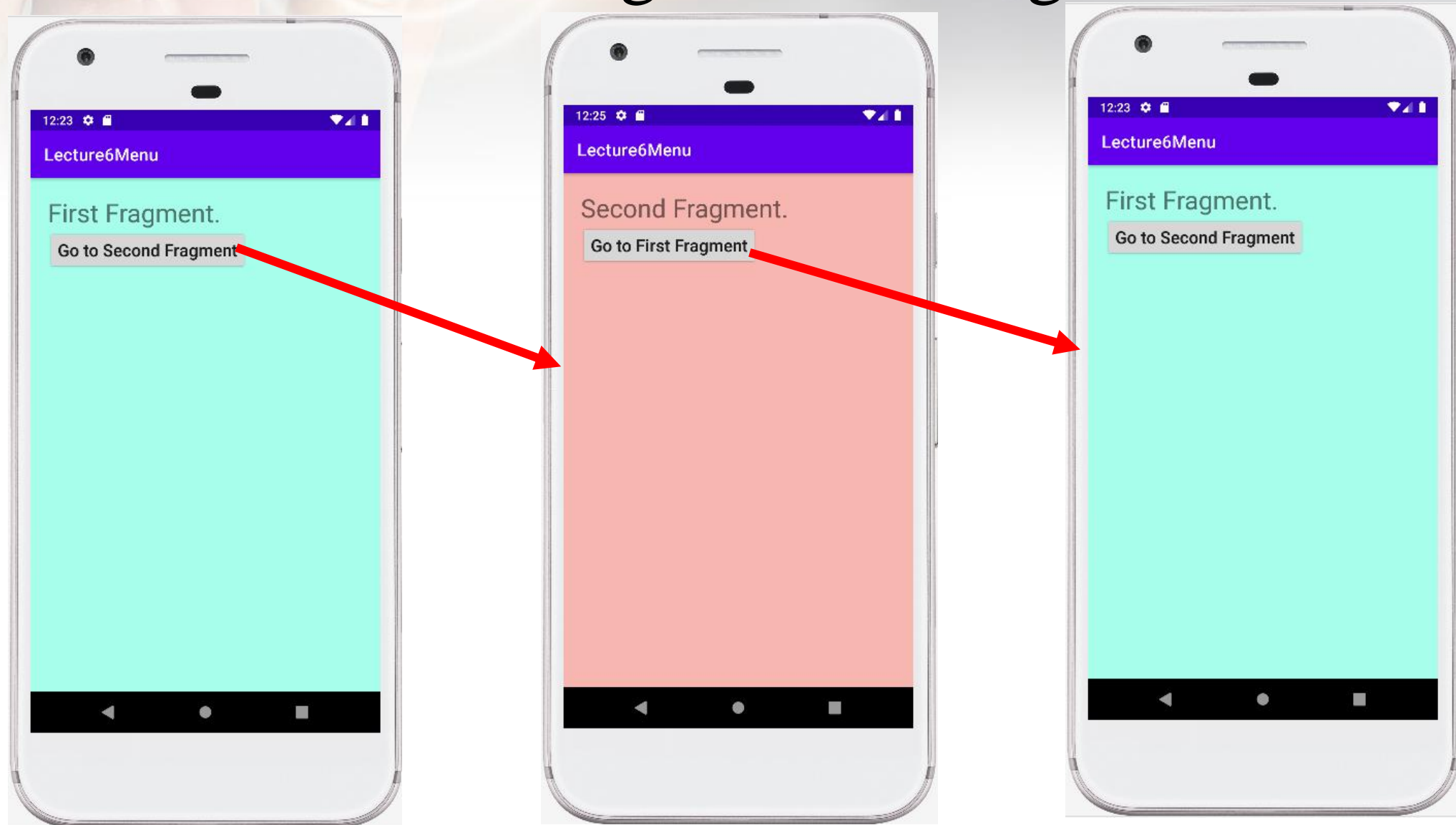
- MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding : ActivityMainBinding  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
    }  
}  
  
fun clickFirstFragment(v: View) {  
    supportFragmentManager.beginTransaction().add(  
        R.id.frameLayout,  
        FirstFragment()  
    ).commit()  
}
```



```
fun clickSecondFragment(v: View) {  
    supportFragmentManager.beginTransaction().add(  
        R.id.frameLayout,  
        SecondFragment()  
    ).commit()  
}
```

# Call Another Fragment on Fragment





# Call Another Fragment on Fragment

- activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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"
    android:orientation="vertical"
    tools:context=".MainActivity">

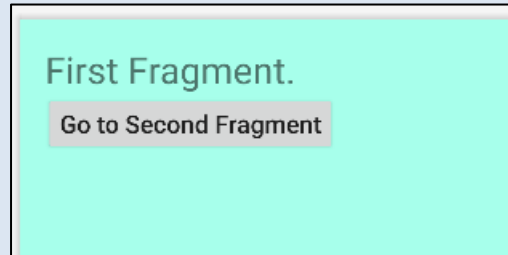
    <FrameLayout
        android:id="@+id/frameLayout"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</LinearLayout>
```

# Call Another Fragment on Fragment

## fragment\_first.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    android:background="#A7FFEB"
    android:padding="20dp"
    tools:context=".FirstFragment">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="First Fragment."
        android:textSize="30sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnClickFirstFrag"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAllCaps="false"
        android:textSize="20sp"
        android:text="Go to Second Fragment" /> </LinearLayout>
```



## fragment\_second.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    android:background="#F8B6B1"
    android:padding="20dp"
    tools:context=".SecondFragment">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second Fragment."
        android:textSize="30sp"/>
    <androidx.appcompat.widget.AppCompatButton
        android:id="@+id/btnClickSecondFrag"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAllCaps="false"
        android:textSize="20sp"
        android:text="Go to First Fragment" /> </LinearLayout>
```



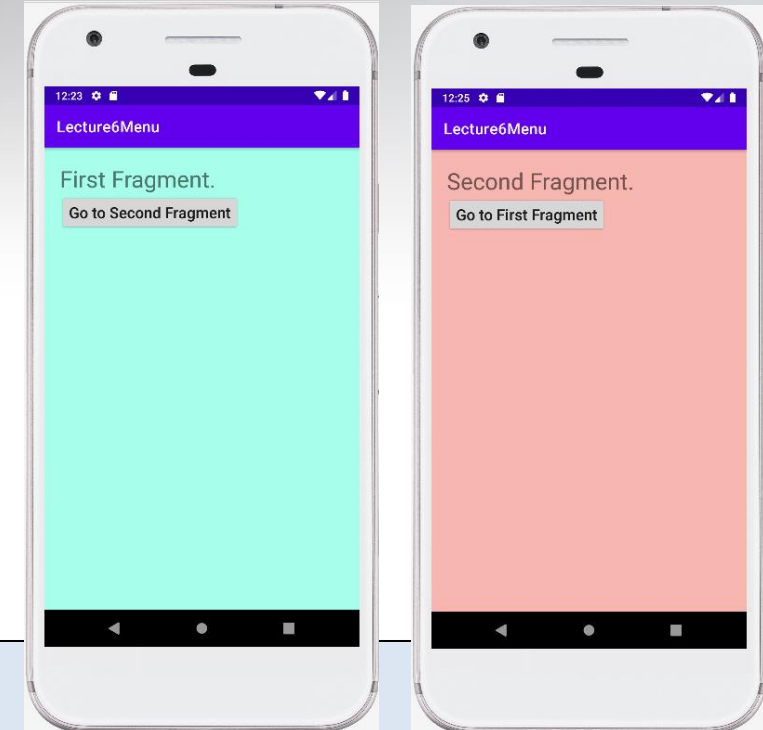


# Call Another Fragment on Fragment

## FirstFragment.kt

```
class FirstFragment : Fragment() {  
    private lateinit var bindingFirst : FragmentFirstBinding  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        // Inflate the layout for this fragment  
        bindingFirst = FragmentFirstBinding.inflate(layoutInflater)  
        bindingFirst.btnClickFirstFrag.setOnClickListener(){  
            var fragment : Fragment? = null  
            fragment = SecondFragment()  
            replaceFragment(fragment)  
        }  
        return bindingFirst.root  
    }  
}
```

```
fun replaceFragment(someFragment: Fragment){  
    var binding: ActivityMainBinding  
    binding = ActivityMainBinding.inflate(layoutInflater)  
    val transaction = requireActivity().supportFragmentManager.beginTransaction()  
    transaction.replace(binding.frameLayout.id, someFragment)  
    transaction.commit()  
}
```

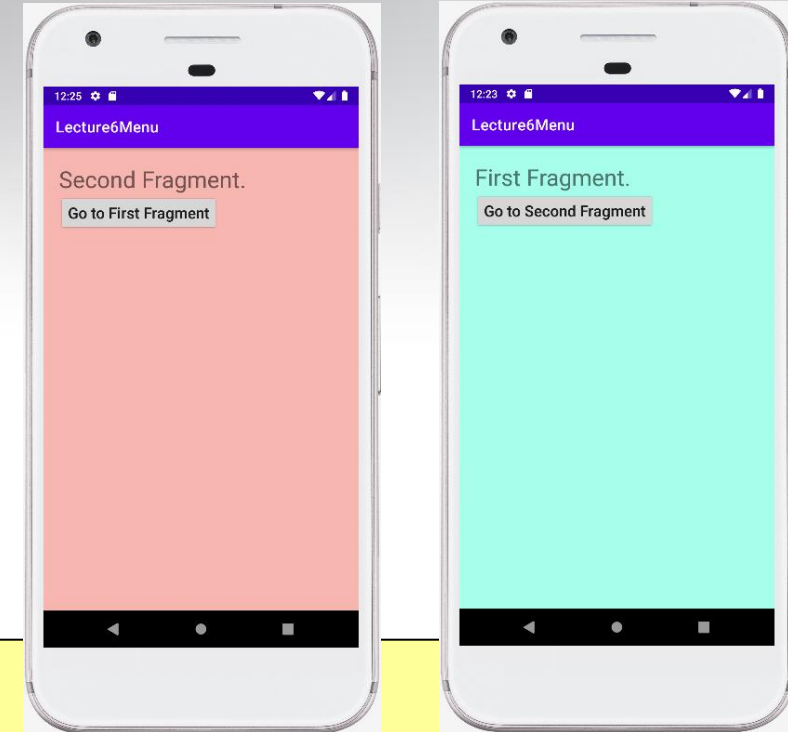


# Call Another Fragment on Fragment

## SecondFragment.kt

```
class SecondFragment : Fragment() {  
    private lateinit var bindingSecond : FragmentSecondBinding  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        // Inflate the layout for this fragment  
        bindingSecond = FragmentSecondBinding.inflate(inflater)  
        bindingSecond.btnClickSecondFrag.setOnClickListener(){  
            var fragment : Fragment? = null  
            fragment = FirstFragment()  
            replaceFragment(fragment)  
        }  
        return bindingSecond.root  
    }  
}
```

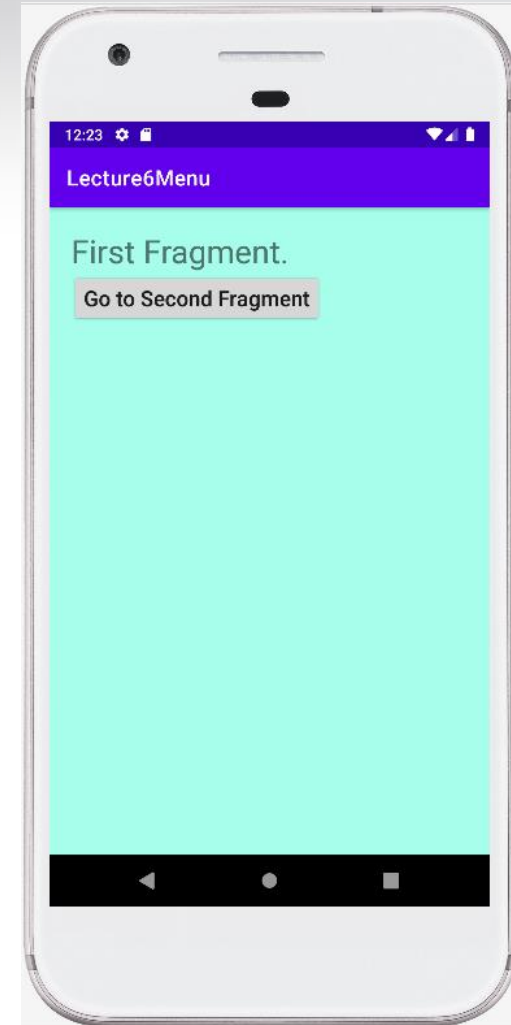
```
fun replaceFragment(someFragment: Fragment){  
    var binding: ActivityMainBinding  
    binding = ActivityMainBinding.inflate(inflater)  
    val transaction = requireActivity().supportFragmentManager.beginTransaction()  
    transaction.replace(binding.frameLayout.id, someFragment)  
    transaction.addToBackStack(null)  
    transaction.commit()  
}
```



# Call Another Fragment on Fragment

## MainActivity.kt

```
class MainActivity : AppCompatActivity() {  
  
    private lateinit var binding : ActivityMainBinding  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(layoutInflater)  
        setContentView(binding.root)  
  
        supportFragmentManager.beginTransaction().add(  
            R.id.frameLayout,  
            FirstFragment()  
        ).commit()  
    }  
}
```





# End of Chapter





# References

- <https://www.javatpoint.com/android-option-menu-example>
- <http://home.iitk.ac.in/~triveni/fragment.pptx>
- <https://marif.yolasite.com/resources/Android-Lecture6-Fragments.pptx>
- [http://www.cs.unibo.it/projects/android/2014/slides/16\\_fragments.ppt](http://www.cs.unibo.it/projects/android/2014/slides/16_fragments.ppt)
- [http://www.cse.bgu.ac.il/common/download.asp?FileName=Lecture% 204.ppt&AppID=2&MainID=552&SecID=4667&MinID=3](http://www.cse.bgu.ac.il/common/download.asp?FileName=Lecture%204.ppt&AppID=2&MainID=552&SecID=4667&MinID=3)
- [ftp://103.81.117.86/04% 20IT% 20Department/VBS/Android/android% 20material/android% 20material/UI.ppt](ftp://103.81.117.86/04%20IT%20Department/VBS/Android/android%20material/android%20material/UI.ppt)
- <https://abhiandroid.com/ui/fragment>
- <https://android--code.blogspot.com/2018/02/android-kotlin-menu-and.html>