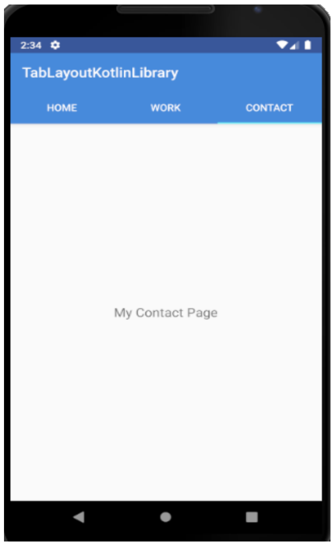
Lesson 7 – Day – 2 – Working with Tab Layout Swipe views using ViewPages.

Tab Layout Example with new Kotlin support Library. Create a TabLayout with three tabs such as HOME, WORK , and CONTACT.



Step 1 : Add the below dependency on your build.gradle

This dependency gives the material design support to your app, we used TabLayout, Viewpager, AppBarLayout and Toolbar UI in the layout from this support library

implementation **'com.google.android.material:material:1.0.0'**

Gradle system takes time to Sync. Wait until get success.

Step 2: Design your activity_main.xml Layout file

Our parent view will be ConstraintLayout. To make a toolbar where application name comes and tabs below it, we need to add AppBarLayout. Now, we will put toolbar and tablayout inside it.

To show the content of each tab, we will require fragment and to show fragment on screen we will again use the ViewPager.

Just copy the below code into your activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <com.google.android.material.appbar.AppBarLayout
        android:id="@+id/appBarLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
        app:layout_constraintBottom_toTopOf="@+id/viewPager"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/ThemeOverlay.AppCompat.Light" />

        <com.google.android.material.tabs.TabLayout
            android:id="@+id/tabs"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:tabBackground="@color/colorPrimary"
            app:tabGravity="fill"
            app:tabMode="fixed"
            app:tabTextColor="@android:color/white" />
    </com.google.android.material.appbar.AppBarLayout>

    <androidx.viewpager.widget.ViewPager
        android:id="@+id/viewPager"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/appBarLayout">

    </androidx.viewpager.widget.ViewPager>
</androidx.constraintlayout.widget.ConstraintLayout>
```

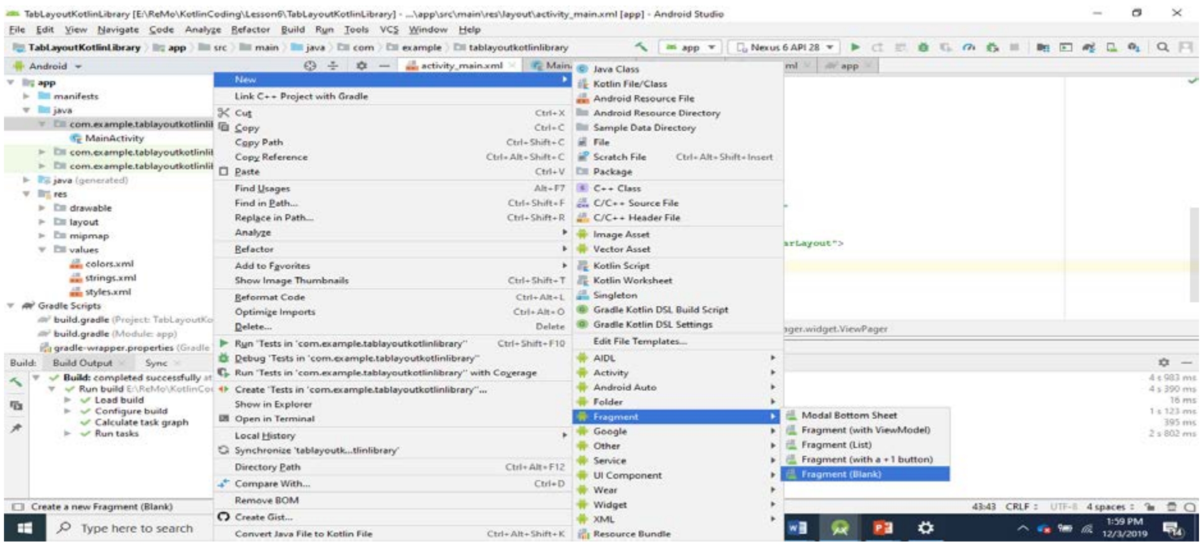
Step 3: From the above code we customize the Appbar and Toolbar themes. So go to your styles.xml and change the `<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">`

As mentioned below

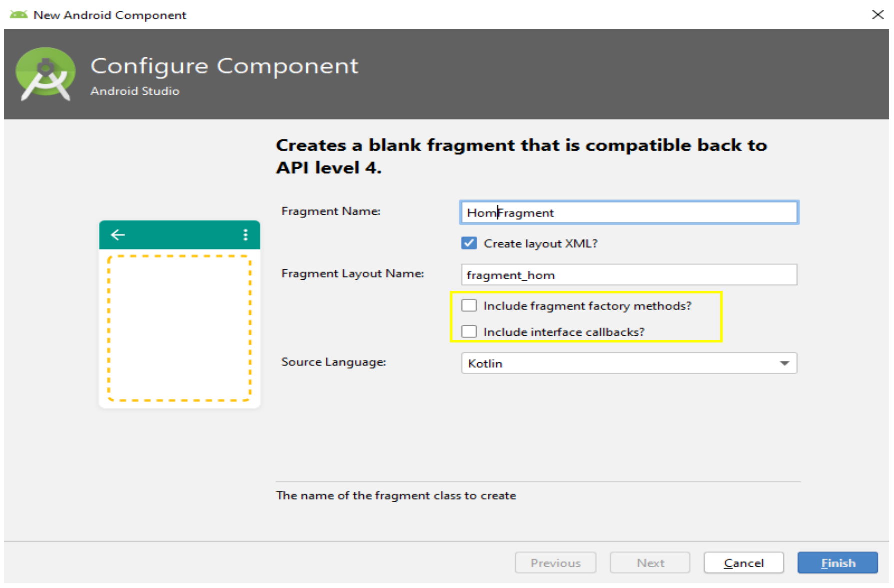
```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
```

Step 4: we need to create three fragment classes and their three respective layouts.

Fragment 1 : HomeFragment, File→New→Fragment→Fragement(Blank)



Step 5 : After doing the Step 4, you will get the below screen. Give the name for the Fragment and untick the highlighted part.



Creates a blank fragment that is compatible back to API level 4.

Fragment Name: HomFragment

[✓] Create layout XML?

Fragment Layout Name: fragment_hom

[ ] Include fragment factory methods?

[ ] Include interface callbacks?

Source Language: Kotlin

The name of the fragment class to create

Step 6: You will the auto generated code for the fragment and layout, as mentioned below.

HomeFragment.kt

```
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup

class HomeFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_home, container, false)
    }
}
```

fragment_home.xml (Just I modified and added Gravity and textSize attributes.

```
<?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">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="My Home Page"
        android:textSize="18sp"
        android:gravity="center" />

</FrameLayout>
```

Step 7 : Repeat the step 4 and step 5, for creating WorkFragment and ContactFragment

WorkFragment.kt

```
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup


class WorkFragment : Fragment() {
    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_work, container, false)
    }
}
```

fragment_work.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=".WorkFragment">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="My Work Page"
        android:textSize="18sp"
        android:gravity="center"/>

</FrameLayout>
```

ContactFragment.kt

```
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup

class ContactFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_contact, container, false)
    }


}
```

fragment_contact.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=".ContactFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="My Contact Page"
        android:textSize="18sp"
        android:gravity="center"/>

</FrameLayout>
```

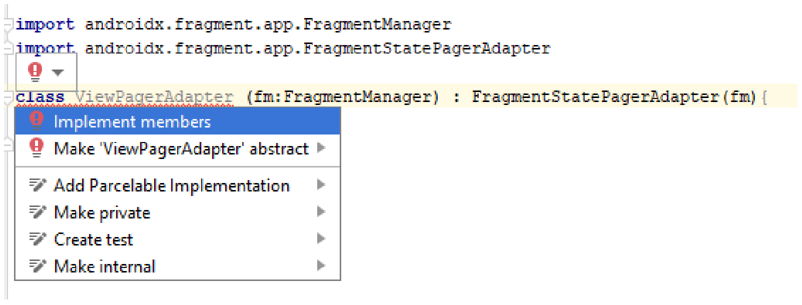
Step 8: Right Click on your Project Explorer New→ Kotlin File/Class

To connect all our fragments with the ViewPager, we need an adapter class. Here, we are creating ViewPagerAdapter class and implementing the FragmentStatePagerAdapter or FragmentPagerAdapter.

In this class, we will pass the list of fragment class instance and the title which we need to show on the tabs. FragmentStatePagerAdapter's override method will help us to get the count of the tab, the title of the tab and which fragment on which position.

Create a ViewPageAdapter class as mentioned below, use the same imports, you will a error, need to override the methods.

At last, from our MainActivity which is our host activity of all the fragment, we wire up the viewPager with the adapter class. Also, we will bind our viewPager with our TabLayout.

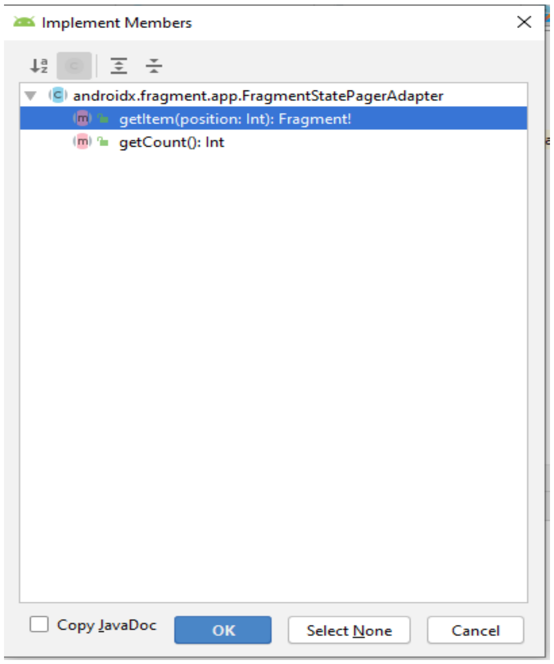


```
import androidx.fragment.app.FragmentManager
import androidx.fragment.app.FragmentStatePagerAdapter

class ViewPagerAdapter (fm:FragmentManager) : FragmentStatePagerAdapter(fm){
```

Implement members
Make 'ViewPagerAdapter' abstract
Add Parcelable Implementation
Make private
Create test
Make internal

Click all the implemented methods and select OK


Implement Members
androidx.fragment.app.FragmentStatePagerAdapter
getItem(position: Int): Fragment!
getCount(): Int
Copy JavaDoc
OK
Select None
Cancel

Complete code ViewPagerAdapter.kt

```
import androidx.fragment.app.Fragment
import androidx.fragment.app.FragmentManager
import androidx.fragment.app.FragmentStatePagerAdapter

class ViewPagerAdapter (fm:FragmentManager) : FragmentStatePagerAdapter(fm){
    private val mFragmentList = ArrayList<Fragment>()
    private val mFragmentTitleList = ArrayList<String>()
    // return the right fragment tabbed
    override fun getItem(position: Int): Fragment {
        return mFragmentList[position]
    }
    // return the count of tabs
    override fun getCount(): Int {
        return mFragmentList.size
    }
    override fun getPageTitle(position: Int): CharSequence? {
        return mFragmentTitleList[position]
    }

    fun addFragment(fragment: Fragment, title: String) {
        mFragmentList.add(fragment)
        mFragmentTitleList.add(title)
    }
}
```

Step 9 : MainActivity.kt

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        // Set the toolbar into your appbar in onCreate
        setSupportActionBar(toolbar)
        // Create an object of ViewPagerAdapter by passing supportFragmentManager
        val adapter = ViewPagerAdapter(supportFragmentManager)
        // Call the addFragment to add Fragment tabs and Tab Title
        adapter.addFragment(HomeFragment(), "HOME")
        adapter.addFragment(WorkFragment(), "WORK")
        adapter.addFragment(ContactFragment(), "CONTACT")
        // set your adpater to the ViewPager UI on the Layout
        viewPager.adapter = adapter
        // set the ViewPager to the respective tabs
        tabs.setupWithViewPager(viewPager)
    }
}
```