|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| <hitle> | All Gradle Setup | <chare> | 1 | <pext> | --------------view binding-----------------  https://docs.google.com/document/d/1EP81MKUFQsPWHvAdPzTk9hnWWzV4aKsB7O-93qzGbaU/edit  \*\*\*\*\*\*\*\*\*\*\*\*Gradle\*\*\*\*\*\*\*\*\*\*  buildFeatures {  viewBinding true  }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.myText.text = "View binding test!"  }  -----------Security Encrypted SharedPreference-----------  //Security Encrypted Shared Preference  implementation("androidx.security:security-crypto:1.0.0")  // For Identity Credential APIs  implementation("androidx.security:security-identity-credential:1.0.0-alpha03")  // For App Authentication APIs  implementation("androidx.security:security-app-authenticator:1.0.0-alpha02")  // For App Authentication API testing  androidTestImplementation("androidx.security:security-app-authenticator:1.0.0-alpha01")  -----------Retrofit--------------------------  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  -----------Picasso------------------------------  implementation 'com.squareup.picasso:picasso:2.71828'  --------------- RxJava2------------------------------------  implementation "io.reactivex.rxjava2:rxjava:2.2.7"  implementation "io.reactivex.rxjava2:rxandroid:2.1.1"  //RxJava2 with Retrofit  implementation "com.squareup.retrofit2:adapter-rxjava2:2.9.0"  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso for Image Loading  implementation 'com.squareup.picasso:picasso:2.71828'  ------------- Hilt + Rxjava3 + Retrofit + Glide--------------------------------------  If you need Dagger Hilt + Rxjava3 + Retrofit + Glide then go for below set of dependencies  Step 1 : add dependency into your build.gradle (Module: app)  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'dagger.hilt.android.plugin'  id 'kotlin-kapt'  }  android {  namespace 'com.example.daggerhitdogapp'  compileSdk 33  defaultConfig {  applicationId "com.example.daggerhitdogapp"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures{  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  //Hilt dependency  implementation 'com.google.dagger:hilt-android:2.42'  kapt 'com.google.dagger:hilt-android-compiler:2.42'  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  //Rx-Java  implementation 'io.reactivex.rxjava3:rxjava:3.1.6'  implementation 'io.reactivex.rxjava3:rxandroid:3.0.2'  implementation "com.github.akarnokd:rxjava3-retrofit-adapter:3.0.0"  // Glide  implementation 'com.github.bumptech.glide:glide:4.14.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.14.2'  // ktx activity with view model injection  implementation 'androidx.activity:activity-ktx:1.6.1'  }  Step 2: add dependency into you build.gradle(:app)  // Top-level build file where you can add configuration options common to all sub-projects/modules.  buildscript {  repositories {  google()  mavenCentral()  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.42'  }  }  plugins {  id 'com.android.application' version '7.2.1' apply false  id 'com.android.library' version '7.2.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.20' apply false  }  task clean(type: Delete) {  delete rootProject.buildDir  }  ----------------- Hilt Dependency----------------------------------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'dagger.hilt.android.plugin'  id 'kotlin-kapt'  }  //Dagger - Hilt  implementation "com.google.dagger:hilt-android:2.41"  kapt "com.google.dagger:hilt-android-compiler:2.37"  kapt "androidx.hilt:hilt-compiler:1.0.0"  implementation 'androidx.hilt:hilt-navigation-compose:1.0.0-beta01'  This should be at Build.gradle(project level) with compose app  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.39.1'  classpath "com.android.tools.build:gradle:7.0.3"  }  This should be at Build.gradle(project level) without compose app  buildscript {  repositories {  google()  mavenCentral()  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.41'  }  }  //Hilt dependency  //Dagger - Hilt  implementation "com.google.dagger:hilt-android:2.42"  kapt "com.google.dagger:hilt-android-compiler:2.42"  -------------- landscape - glide for compose ------------------------------  //landscape - glide for compose  implementation "com.github.skydoves:landscapist-glide:1.5.0"  --------------- image loading library for compose images ----------------------  //image loading library for compose images  implementation "io.coil-kt:coil-compose:2.0.0-rc01"  --------------For Runtime livedata in compose--------------  //For Runtime livedata in compose  implementation "androidx.compose.runtime:runtime-livedata:$compose\_version"  --------------Retrofit & OkHttp--------------  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  --------------dagger--------------  //dagger  implementation 'com.google.dagger:dagger:2.42'  kapt 'com.google.dagger:dagger-compiler:2.42'  --------------coroutine--------------  //coroutine  def coroutine\_version = "1.6.4"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:$coroutine\_version"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-core:$coroutine\_version"  --------------In coroutines for viewmodel scope--------------  //In coroutines for viewmodel scope  implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:2.2.0"  --------------lifecycle--------------  //lifecycle  implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.4.1'  implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.4.1'  --------------Glide--------------  // Glide  implementation 'com.github.bumptech.glide:glide:4.14.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.14.2'  --------------volley for API calls--------------  // volley for API calls  implementation 'com.android.volley:volley:1.2.1'  --------------gson for type conversion for data source--------------  // gson for type conversion for data source  implementation 'com.google.code.gson:gson:2.9.0'  --------------RxJava 2--------------  // RxJava 2  implementation "io.reactivex.rxjava2:rxjava:2.2.7"  implementation "io.reactivex.rxjava2:rxandroid:2.1.1"  //RxJava2 with Retrofit  implementation "com.squareup.retrofit2:adapter-rxjava2:2.9.0"  --------------RxJava 3--------------  // RxJava 3  implementation 'io.reactivex.rxjava3:rxjava:3.0.0'  implementation 'io.reactivex.rxjava3:rxandroid:3.0.0'  implementation "com.github.akarnokd:rxjava3-retrofit-adapter:3.0.0"  --------------Work manger--------------  // Work manger  implementation 'androidx.work:work-runtime:2.7.1'  implementation 'androidx.work:work-runtime-ktx:2.7.1'  --------------Room database--------------  // Room database  kapt "android.arch.persistence.room:compiler:1.1.1"  implementation 'android.arch.persistence.room:runtime:1.1.1'  --------------Room database--------------  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  --------------Livedata--------------  // Livedata  implementation 'androidx.compose.runtime:runtime-livedata:1.2.0-alpha06'  --------------Unit Testing - Google Truth --------------  //Unit Testing - Google Truth  def googleTruthVersion = "1.1.3"  testImplementation "com.google.truth:truth:$googleTruthVersion"  androidTestImplementation "com.google.truth:truth:$googleTruthVersion"  //Unit testing  testImplementation 'androidx.arch.core:core-testing:2.1.0'  testImplementation 'org.jetbrains.kotlinx:kotlinx-coroutines-test:1.4.1'  testImplementation 'org.mockito:mockito-inline:3.9.0'  testImplementation "org.mockito.kotlin:mockito-kotlin:4.0.0"  //for testing architecture components such as livedata, viewModel etc  testImplementation 'androidx.arch.core:core-testing:2.1.0'  //UI Testing  androidTestImplementation "androidx.test.espresso:espresso-core:3.3.0"  androidTestImplementation "androidx.test:runner:1.3.0"  androidTestImplementation "androidx.test:core:1.3.0"  androidTestImplementation "androidx.test.ext:junit:1.1.2"  androidTestImplementation "androidx.test:rules:1.3.0"  --------------Picasso--------------  //Picasso  implementation 'com.squareup.picasso:picasso:2.71828'  --------------Secured Shared Pref--------------  //Secured Shared Pref  implementation 'androidx.security:security-crypto:1.0.0'  implementation 'androidx.security:security-app-authenticator:1.0.0-alpha02'  --------------for nav graph--------------  // for nav graph  def nav\_version = "2.5.3"  implementation("androidx.navigation:navigation-fragment-ktx:$nav\_version")  implementation("androidx.navigation:navigation-ui-ktx:$nav\_version") | </end> |
| <hitle> | FirstAndroidApp | <chare> | 1 | <pext> | 01-12/ FirstAndroidApp  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout  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"  android:orientation="vertical">  <ImageView  android:layout\_width="200dp"  android:layout\_height="200dp"  android:layout\_gravity="center"  android:src="@drawable/ic\_launcher\_background"/>  <androidx.appcompat.widget.AppCompatEditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Enter your Email"  android:drawableStart="@drawable/baseline\_mail\_24"  android:drawablePadding="20dp"  android:layout\_margin="20dp"/>  <androidx.appcompat.widget.AppCompatEditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:drawableStart="@drawable/baseline\_key\_24"  android:drawablePadding="20dp"  android:hint="Enter your Password"  android:layout\_margin="20dp"/>  <Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Login"  android:textSize="30sp"  android:layout\_margin="30dp"/>    <Switch  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:text="Not yet Registered?"/>  </LinearLayout>  -------------------Change App Icon----------------  File/New/Image Asset  Edit or Create new image asset and set in AndroidManifest.  <application  …  android:icon="@mipmap/app\_icon"  android:roundIcon="@mipmap/app\_icon\_round"  …  -------------------Convert Java file to Kotlin file-------------------  Right click on the source java file and “Convert Java file to Kotlin file.” | </end> |
| <hitle> | Kotlinbasics | <chare> | 1 | <pext> | 01-13/ Kotlinbasics  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout  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"  android:orientation="vertical">  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30sp"  android:layout\_gravity="center"  android:layout\_margin="20dp"  android:id="@+id/txtResult"  />  <androidx.appcompat.widget.AppCompatEditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="20dp"  android:hint="Enter number1"  android:id="@+id/editNum1"  />  <androidx.appcompat.widget.AppCompatEditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="20dp"  android:hint="Enter number2"  android:id="@+id/editNum2"  />  <Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="20dp"  android:text="Add"  android:id="@+id/btnAdd"  />  <Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="20dp"  android:text="SUB"  android:id="@+id/btnSub"  />  <Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="20dp"  android:text="MUL"  android:id="@+id/btnMul"  />  <Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="20dp"  android:text="DIV"  android:id="@+id/btnDiv"  />  </LinearLayout>  class MainActivity : AppCompatActivity() {  lateinit var result: TextView  lateinit var input1: EditText  lateinit var input2: EditText  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  initViews()  }  private fun initViews() {  result = findViewById<TextView>(R.id.txtResult)  input1 = findViewById<EditText>(R.id.editNum1)  input2 = findViewById<EditText>(R.id.editNum2)  findViewById<Button>(R.id.btnAdd).setOnClickListener {  result.text = Calculator.add(input1.text.toString().toInt(), input2.text.toString().toInt()).toString()  }  findViewById<Button>(R.id.btnSub).setOnClickListener {  result.text = Calculator.sub(input1.text.toString().toInt(), input2.text.toString().toInt()).toString()  }  findViewById<Button>(R.id.btnMul).setOnClickListener {  result.text = Calculator.mul(input1.text.toString().toInt(), input2.text.toString().toInt()).toString()  }  findViewById<Button>(R.id.btnDiv).setOnClickListener {  result.text = Calculator.div(input1.text.toString().toInt(), input2.text.toString().toInt()).toString()  }  }  }  object Calculator {  fun add(a: Int, b :Int): Int {  return a + b  }  fun sub(a: Int, b :Int): Int {  return a - b  }  fun mul(a: Int, b :Int): Int {  return a \* b  }  fun div(a:Int, b :Int): Int {  return a / b  }  }  --------------extension function demo------------  class Circle(val radius: Double) {  fun area(): Double {  return Math.PI \* radius \* radius  }  }  fun Circle.perimeter(): Double {  return 2 \* Math.PI \* this.radius  }  fun String.isPalindrome() {}  fun main() {  val circle = Circle(12.0)  println(circle.area())  println(circle.perimeter())  }  ----------------enum class demo-----------------  enum class Days(val isWeekEnd: Boolean = true) {  SUNDAY(true),  MONDAY,  TUESDAY,  WEDNESDAY,  FRIDAY,  SATURDAY(true)  }  fun main() {  println("Only one day i will be not available in class that is ${Days.WEDNESDAY}")  for (day in Days.values()) {  println("${day.ordinal} = ${day.name}")  }  for(day in Days.values()) {  println("${day.ordinal} = ${day.name} is a weekend? = ${day.isWeekEnd}")  }  }  enum class AppExceptions(val isInvoked: Boolean = false) {  BAD\_REQUEST,  NO\_INTERNET,  SERVER\_NOT\_WORKING,  SOMETHING\_WENT\_WRONG(true)  }  -------------Inner/Outer class demo-------------  class OuterClassDemo {  val aboutYourPlace = "Which country you are living in?"  class NestedClass {  val city = "London"  val country = "United Kingdom"  }  }  fun main() {  val outerClassObj = OuterClassDemo()  val nestedClassObj = OuterClassDemo.NestedClass()  println(outerClassObj.aboutYourPlace)  println("I live in ${nestedClassObj.city} that is in ${nestedClassObj.country}")  }  class Car {  val ownerOfCars = "Thomas"  inner class TeslaCar() {  fun workingOfCar() {  println("This is best electric car")  println("This car is purchase by $ownerOfCars")  }  }  inner class AudiCar() {  fun workingOfCar() {  println("This is best petrol car")  println("This car is purchase by $ownerOfCars")  }  }  }  fun main() {  val teslaCarObj = Car().TeslaCar()  println(teslaCarObj.workingOfCar())  val audiCarObj = Car().AudiCar()  println(audiCarObj.workingOfCar())  }  ----------------sealed class demo-----------------  sealed class SealedClassDemo {  class A: SealedClassDemo() {  fun displaySomething() {  println("Subclass A of Sealed class Demo")  }  }  class B: SealedClassDemo() {  fun displaySomething() {  println("Subclass B of Sealed class Demo")  }  }  class C: SealedClassDemo() {  fun displaySomething() {  println("Subclass C of Sealed class Demo")  }  }  }  fun main() {  val objA = SealedClassDemo.A()  objA.displaySomething()  val objB = SealedClassDemo.B()  objB.displaySomething()  val objC = SealedClassDemo.C()  objC.displaySomething()  }  class SealedClassException {  class InternetError: SealedClassDemo() {  fun displaySomething(errorCode: Int, errorMessage: String) {  println("Got a exception where ErrorCode is $errorCode and ErrorMessage is $errorMessage")  }  }  class SimpleException: SealedClassDemo() {  fun displaySomething() {  println("Subclass SimpleException of Sealed class Demo")  }  }  class BadRequest: SealedClassDemo() {  fun displaySomething(errorCode: Int, errorMessage: String, whatIsWrong: String) {  println("Got a exception where ErrorCode is $errorCode and ErrorMessage is $errorMessage and change this from request $whatIsWrong")  }  }  }  fun main() {  val objA = SealedClassException.InternetError()  objA.displaySomething(400, "check your internet")  val objB = SealedClassException.SimpleException()  objB.displaySomething()  val objC = SealedClassException.BadRequest()  objC.displaySomething(500, "Bad attempt", "Try to add something from header")  }  sealed class SealedClassStatus {  class Loading: SealedClassStatus() {  fun displaySomething() {  println("Loader is loading")  }  }  class Success: SealedClassStatus() {  fun displaySomething() {  println("You got a success result")  }  }  class Failure: SealedClassStatus() {  fun displaySomething(errorCode: Int, errorMessage: String, whatIsWrong: String) {  println("Got a exception where ErrorCode is $errorCode and ErrorMessage is $errorMessage and change this from request $whatIsWrong")  }  }  }  --------------Sealed/Inner/Nested class Exercise---------  sealed class SealedClass {  class A: SealedClass() {  fun print() {  println("This is A")  }  }  class B: SealedClass() {  fun print() {  println("This is B")  }  }  fun sealedPrint() {  println("This is SealedClass")  }  }  class NestedInnerClass {  val outerMember: String = "This is NestedInner class member"  class NestedClass {  val nestedMember: String = "This is Nested class member"  fun print() {  /\* println(outMember) //We can't call outer class member in nested class \*/  println(nestedMember)  }  }  inner class InnerClass {  val innerMember: String = "This is Inner class member"  fun print() {  println(outerMember)  println(innerMember)  }  }  }  fun String.extensionTest() {  println("We can use extension to extend String function")  }  fun main() {  /\* val sealed = SealedClass() // We can instantiate SealedClass \*/  val sealedA = SealedClass.A()  val sealedB = SealedClass.B()  sealedA.print()  sealedB.print()  sealedA.sealedPrint() // We can call sealed class member.  val nestObj = NestedInnerClass.NestedClass() // Please check  val innerObj = NestedInnerClass().InnerClass() //Please check  val myString: String = "Hey!"  myString.extensionTest()  }  ---------------Expression/Condition Demo------------------  fun evaluateMax(a: Int, b: Int) {  println(if (a > b) a else b)  }  fun evaluateMin(a: Int, b: Int) {  println(if (a < b) a else b)  }  fun myFavPlaceInLondon(isWeekEnd: Boolean, isSunday: Boolean) {  if (isWeekEnd) {  println("London bridge")  } else if (isSunday) {  println("relax at home")  } else {  println("work")  }  }  fun main() {  evaluateMax(10, 5)  evaluateMin(100, 90)    println(Math.max(10, 5))  println(Math.min(100, 90))    when (readLine().toString()) {  "Apple" -> println("Rich with Vitamins")  "Banana" -> println("Rich with Iron")  "Orange" -> println("Rich with Vitamin C")  else -> println("Anyways good for your health")  }  }  ------------Repeat/For/While Loop Demo-----------------  fun main() {  repeat(5){  println("Android class will go from Mon to Friday")  }    repeat(5) { position ->  println("Job done for task $position")  }    val vowels = arrayOf('a', 'i', 'e', 'o', 'u')    for (vowel in vowels) {  println(vowel)  }    for (index in vowels.indices) {  println(vowels[index])  }    for ((index, vowel) in vowels.withIndex()) {  println("Vowel at index $index is $vowel")  }    val listOfFruits = listOf("Apple", "Mango", "Banana", "Orange")    for (fruit in listOfFruits) {  println(fruit)  }    val cityAndCountry = mapOf("London" to "United Kingdom", "Delhi" to "India", "Washinton" to "USA")    for (item in cityAndCountry) {  println("${item.key} is capital city of ${item.value}}")  }    for ((capital, country) in cityAndCountry) {  println("$capital is capital city of $country")  }    for (i in 1..5) {  println(i)  }    for (i in 1..20 step 4) {  println(i)  }    for (i in 20 downTo 1) {  println(i)  }    var counter = 0  while (counter < 5) {  println("item in range $counter")  if (counter == 3) {  break  }  counter++  }    while (false) {  println("This will never execute")  }    counter = 0  do {  println("item in range $counter")  counter++  } while (counter < 5)    outer@ for (i in 1..10) {  inner@ for (j in 1..9) {  println("i is $i and j is $j")  if(i == 3) {  break@outer  }  }  }    for (i in 1..10) {  if(i == 5) {  continue  }  println(i)  }    first@ for (i in 1..4) {  second@ for (j in 1..3) {  if (i==2) {  continue@first  }  println("i is $i and j is $j")  }  }  }  ------------------Print Pyramid using Loop-------------------  fun main() {  println("Half Pyramid")  for (i in 1..5) {  repeat(i) {  print("\*")  }  println()  }  println("Inverted Half Pyramid")  for (i in 5 downTo 1) {  repeat(i) {  print("\*")  }  println()  }  println("Hollow Inverted Half Pyramid")  for (i in 5 downTo 1) {  repeat(i) {  if (i == 1 || i == 5) {  print("\*")  } else {  if (it == 0 || it == i - 1) {  print("\*")  } else {  print(" ")  }  }  }  println()  }  println("Pyramid")  val bottom = 13  val ph: Int = bottom / 2  for (i in 0..ph) {  repeat(bottom) {  if (it > ph - i && it < ph + i) {  if (i % 2 == 1) {  if (it % 2 == 0) {  print("\*")  } else {  print(" ")  }  } else {  if (it % 2 == 0) {  print(" ")  } else {  print("\*")  }  }  } else {  print(" ")  }  }  println()  }  println("Down Pyramid")  for (i in ph downTo 0) {  repeat(bottom) {  if (it > ph - i && it < ph + i) {  if (i % 2 == 1) {  if (it % 2 == 0) {  print("\*")  } else {  print(" ")  }  } else {  if (it % 2 == 0) {  print(" ")  } else {  print("\*")  }  }  } else {  print(" ")  }  }  println()  }  println("Hollow Pyramid")  for (i in 1..ph) {  repeat(ph - i) {  print(" ")  }  repeat(i) {  if (i == 1) {  print("\*")  } else if (i == ph) {  print("\* ")  } else {  if (it == 0 || it == i - 1) {  print("\* ")  } else {  print(" ")  }  }  }  repeat(ph - i) {  print(" ")  }  println()  }  }  -------------LateInit/Lazy Demo--------------  data class Employee(  val name: String,  val age: Int,  val salary: Double  )  class LateInitAndLazy {  private lateinit var employee: Employee  private val employeeUsingLazy by lazy { Employee("Jones", 33, 12.0) }  fun setEmp() {  employee = Employee("Thomas", 33, 11.11)  }  fun getEmp() {  if(this::employee.isInitialized) {  println(employee.name) //lateinit property employee has not been initialized  } else {  println("Employee is not available")  }  }    fun getEmpLazyObj() {  println(employeeUsingLazy.name)  }  }  fun main() {  val lateInitAndLazy = LateInitAndLazy()  lateInitAndLazy.getEmp()  lateInitAndLazy.getEmpLazyObj()  }  -------------Lambda Demo----------------  fun main() {  val city = { println("London") }  city.invoke()  val countries = {  println("India")  println("United Kingdom")  println("USA")  }  countries.invoke()  val sum = { a: Int, b: Int -> a + b }  println( sum(1,4) )  }  -------------HighOrder function Demo----------------  val lambda = { println("Hi there") }  fun highOrderFunc(lmdb: () -> Unit) {  lmdb()  }  val sum = { a: Int, b: Int -> a + b }  fun getSum(sumLambda: (Int, Int) -> Int) {  println(sumLambda(1, 9))  }  fun getBestFruits(fruit: String): Unit {  println(fruit)  }  fun showBestFruits(fruit: String, bestFruit: (String) -> Unit) {  bestFruit(fruit)  }  fun sub(a: Int, b: Int): Int {  return a - b  }  fun doCalculate(subFunc: (Int, Int) -> Int) {  println(subFunc(10, 3))  }  fun mul(a: Int, b: Int): Int {  return a \* b  }  fun doMultiply(): ((Int, Int) -> Int) {  return ::mul  }  fun main() {  highOrderFunc(lambda)  getSum(sum)  showBestFruits("Apple", ::getBestFruits)  doCalculate(::sub)  val function = doMultiply()  println(function(10, 10))  }  ------------IsAnagram, searchPalindromicString------------  fun checkAnagram(s1: String, s2: String): Boolean {  if(s1.length != s2.length) return false  val char1 = s1.toCharArray().groupBy { it }  val char2 = s2.toCharArray().groupBy { it }  if(char1 == char2) return true  return false  }  fun checkAnagram1(s1: String, s2: String): Boolean {  if(s1.length != s2.length) return false  if( s1.toCharArray().sorted().joinToString("") == s2.toCharArray().sorted().joinToString("") ) return true  return false  }  fun isPalindromic(str: String): String {  for(i in str.length downTo 2) {  val windowed = str.windowed(i)  windowed.forEach {  if(it == it.reversed()) {  return it  }  }  }  return "There's no palindromic string!"  }  fun main() {  println(checkAnagram1("abcc", "bcca"))  println(isPalindromic("adfaewfwe"))  }  ------------String Discussion---------------------  fun main() {  val stringTest = "Thomas"  println(stringTest[0])  for (i in stringTest.indices) {  println(stringTest[i])  }  println(stringTest.length)  println(stringTest.get(3))  println(stringTest.subSequence(1,4))  println(stringTest == "Test")  println("What is your name \n Thomas")  val rawString = """Sunrise happens from east  | sunset happens from west  """.trimMargin()  println(rawString)  var myPhase = "\"A Quick brown fox jump over the\'"  println(myPhase)  var myPhase1 = "\"A Quick brown fox jump over the \""  println(myPhase1)  var a = "sunset"  var b = "sunset"  var c = "sun"  var d = c  println(a === b) //true both are the same string pool  println(a == b) //true both values are the same  println(b === c) //false both are different string pool  println(c === d) //true both are the same reference  }  -------constructor Demo----------------  //primary constructor  class A constructor(val a: Int, val b: String) {}  class B(val a: Int, val b: String) {}  //init block  class School {  init {  println(" This is init block 1")  }  init {  println(" This is init block 2")  }  init {  println(" This is init block 3")  }  }  //primary constructor and init block  class Animal(val name: String, val place: String) {  init {  println("Name is $name and it is at $place")  }  }  class City(name: String, special: String) {  val \_name: String  val \_special: String  init {  \_name = name  \_special = special  }  fun printData() {  println("city is $\_name and famous is $\_special")  }  }  //secondary constructor  class Sum {  constructor(a: Int, b: Int) {  val c = a+b  println("result is $c")  }  }  //multiple secondary constructor  class FlexibleAddition {  constructor(a: Int, b: Int) {  val c = a+b  println("sum is $c")  }  constructor(a: Int, b: Int, c: Int) {  val d = a+b+c  println("sum is $d")  }  constructor(a: Int, b: Int, c: Int, d: Int) {  val e = a+b+c+d  println("sum is $e")  }  }  //secondary constructor chaining  class Addition {  constructor(a: Int, b: Int): this(a, b, 7) {  val c = a+b  println("SecondaryConstructor1 sum is $c")  }  constructor(a: Int, b: Int, c: Int): this(a, b, c, 10) {  val d = a+b+c  println("SecondaryConstructor2 sum is $d")  }  constructor(a: Int, b: Int, c: Int, d: Int) {  val e = a+b+c+d  println("SecondaryConstructor3 sum is $e")  }  }  //secondary constructor chaining into inheritance  open class Parent {  constructor(name: String, hobby: String, age: Int, salary: Double) {  println("Parent has Name is $name hobby is $hobby age is $age salary is $salary")  }  }  class Child: Parent {  constructor(name: String, hobby: String, age: Int): super(name, hobby, age, 45.0) {  println("Child has Name is $name hobby is $hobby age is $age")  }  }  fun main() {  val objA = A(100, "Rate")  val objB = B(50, "Rate")  val school = School()  val animal = Animal("Lion", "India")  City("London", "London Eye").printData()  Sum(5, 10)  FlexibleAddition(2, 3)  FlexibleAddition(2, 3, 4)  FlexibleAddition(2, 3, 4, 5)  Addition(3, 5) //constructor chaining  Child("Thomas", "Soccer", 33)  }  ------------------class Inheritance Demo---------------  open class Employee {  var empId = 0  var firstName = ""  var lastName = ""  var email = ""  fun getFullName() {  println("$firstName $lastName")  }  }  class FullTimeEmployee: Employee() {  var annualSalary = 0  }  class PartTimeEmployee: Employee() {  var hourSalary = 0  }  open class Parent {  fun parentMethod() {  println("parent method")  }  }  class Child: Parent() {  fun childMethod() {  println("child method")  }  }  fun testAbove1() {  var parent = ClassInheritanceDemo.Parent()  parent.parentMethod()  var child = ClassInheritanceDemo.Child()  child.parentMethod()  child.childMethod()  }  ---------------abstract class Demo---------------  abstract class Shape {  var width = 0.0  var height = 0.0  var radius = 0.0  var pie = 3.14f  abstract fun getArea()  }  class Retangle(width: Double, height: Double): Shape() {  override fun getArea() {}  }  class Circle(radius: Double): Shape() {  override fun getArea() {}  }  ---------------Interface Demo----------------  interface IBankAccount {  fun deposit(amount: Int): Boolean  fun withdraw(amount: Int): Boolean  fun getBalance()  }  class SavingAccount : IBankAccount {  var \_balance: Int = 0  var \_perDayLimit: Int = 0  override fun deposit(amount: Int): Boolean {  \_balance += amount  println("Successfully deposited: $amount")  return true  }  override fun withdraw(amount: Int): Boolean {  if(\_balance < amount) {  println("Insufficient balance")  return false  }  // limit 5000  else if (\_perDayLimit + amount > 5000) {  println("Withdraw attempt failed")  return false  } else {  \_balance -= amount  \_perDayLimit += amount  println("Successfully withdraw: $amount")  return true;  }  }  override fun getBalance() {  println("Saving account balance: $\_balance")  }  }  class CurrentAccount : IBankAccount{  var \_balance: Int = 0  override fun deposit(amount: Int): Boolean {  \_balance += amount  println("Successfully deposited: $amount")  return true  }  override fun withdraw(amount: Int): Boolean {  if(\_balance < amount) {  println("Insufficient balance")  return false  }else {  \_balance -= amount  println("Successfully withdraw: $amount")  return true;  }  }  override fun getBalance() {  println("Current account balance: $\_balance")  }  }  fun main(){  val savingAccount = SavingAccount()  val currentAccount = CurrentAccount()  savingAccount.deposit(2000)  savingAccount.withdraw(1000)  savingAccount.withdraw(6000)  savingAccount.deposit(10000)  savingAccount.withdraw(6000)  savingAccount.getBalance()  println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")  currentAccount.deposit(2000)  currentAccount.withdraw(1000)  currentAccount.withdraw(6000)  currentAccount.deposit(10000)  currentAccount.withdraw(6000)  currentAccount.getBalance()  }  -----------Compile/RunTime pholymorphism-------------  //Compile Time polymorphism  fun doubleOf(a: Int): Int { return 0 }  //fun doubleOf(a: Int): Float { return 0f } //this is big error  fun doubleOf(a: Float): Int { return 1 }  fun doubleOf(a: Int, b: Float) {}  //RunTime polymorphism  open class ParentMachine {  open fun feature1() { println("feature1 of parent") }  open fun feature2() {}  }  class ChildMachine: ParentMachine() {  override fun feature1() { println("feature1 of child") }  }  fun main() {  val child = ChildMachine()  child.feature1()  val parent: ParentMachine = child  parent.feature1() //Parent indicator but child func called  }  -------------------ArrayDemo--------------  fun main() {  val intArray = arrayOf(1,23,4,5,6)  val intArrayType = intArrayOf(3,4,5,6,3,67)  val charArray = charArrayOf('a','b','c')  val longArray = longArrayOf()  val byteArray = byteArrayOf()    val x = intArray.get(0)  val y = intArray[0]    intArrayType.set(3,1000)  intArrayType[3] = 1000  }  ----------------GroupAnagrams---------------------  fun groupAnagrams(list: List<String>) : List<List<String>> {  val source = list.toMutableList()  val result = mutableListOf<MutableList<String>>()  list.forEach {  var added = false  result.forEach { cur ->  if(cur.size > 0) {  if( cur[0] == "" && it == "" || cur[0].toCharArray().sorted().joinToString("") == it.toCharArray().sorted().joinToString("")) {  cur.add(it)  added = true  }  }  }  if(!added) {  val cur = mutableListOf<String>()  cur.add(it)  result.add(cur)  }  }  return result  }  fun groupAnagrams1(list: List<String>) : List<List<String>> {  val source = list.toMutableList()  val result = mutableListOf<List<String>>()  while(source.size > 0) {  val cur = mutableListOf<String>()  cur.add(source.first())  source.removeAt(0)  var i = 0  while(i< source.size) {  if(cur[0] == "" && source[i] == "") {  cur.add(source[i])  source.removeAt(i)  i--  }  else {  if(source[i].toCharArray().sorted().joinToString("") == cur[0].toCharArray().sorted().joinToString("")) {  cur.add(source[i])  source.removeAt(i)  i--  }  }  i++  }  result.add(cur)  }  return result  }  fun main() {  println(groupAnagrams(listOf("eat","tea","tan","ate","nat","bat")))  println(groupAnagrams(listOf("","","")))  }  --------------List/Collection Operation Demo-----------  fun main() {  val fruits = listOf("mango", "orange", "apple")  val mutableFruits = mutableListOf("Apple", "Orange", "Mango")  //fruits.add("banana") //error  mutableFruits.add("Banana")  //fruits[0] = "Apple" //error  mutableFruits[0] = "apple"  val planets = setOf("Earth", "Mars", "Jupiter")  val mutablePlanets = mutableSetOf("Earth", "Mars", "Jupiter")  //planets.add // error  mutablePlanets.add("Anything")  //planets[0] = "earth" // error  //mutablePlanets[0] = "earth" //error  val cityCountry = mapOf("London" to "United Kingdom", "Delhi" to "India")  val mutableCityCountry = mutableMapOf("London" to "United Kingdom", "Delhi" to "India")  //cityCountry.put("Paris", "France") //error  //cityCountry["London"] = "UK" //error  mutableCityCountry.put("Paris", "France")  mutableCityCountry["Paris"] = "France"  for (item in fruits) {  println(item)  }  for (key in cityCountry.keys) {  println(cityCountry[key])  }  val cities = listOf("NewYork", "London", "Berlin", "Venice")  val sliced = cities.slice(0..2)  println(sliced.toString()) //[NewYork, London, Berlin]  val citiesWithNull = listOf("NewYork", "London", null, "Berlin", "Venice", null)  val nonNullCities = citiesWithNull.filterNotNull()  println(nonNullCities.toString()) //[NewYork, London, Berlin, Venice]  val nums = listOf(0, 1, 300, 88, -100, 39)  val positiveNums = nums.filter { it > 0 }  println(positiveNums.orEmpty()) //[1, 300, 88, 39]  val students = listOf(11, 111, 333, 222, 888, 999)  val dropFailedStudents = students.drop(3)  println(dropFailedStudents.toString()) //[222, 888, 999]  val numlist = mutableListOf(1,3,5,6,4,8)  val groupOfMod3 = numlist.groupBy { it % 3 }  println(groupOfMod3.toString()) //{1=[1, 4], 0=[3, 6], 2=[5, 8]}  val numbers = listOf(1,2,3,4,5,6)  val squareOfNumbers = numbers.map { it\*it }  println(squareOfNumbers.toString()) //[1, 4, 9, 16, 25, 36]  val list = listOf(1,2,3,4,5,6)  val foldList = list.fold(0) { acc, i -> acc + (i\*i) }  println(foldList) // sum of [1, 4, 9, 16, 25, 36] = 91  val foldList1 = list.fold(100) { acc, i -> acc + (i\*i) }  println(foldList1) // sum of [1, 4, 9, 16, 25, 36], add 100 = 191  val foldList2 = list.foldIndexed(0) { index, total, item -> if (index % 2 == 0) (total + item) else total }  println(foldList2) // 1+3+5 = 9  val reduceList = numbers.reduce { acc, i -> acc + (i\*i) }  println(reduceList) //91  val emptyList = listOf<Int>()  //below code is run time exception - //Empty collection can't be reduced.  //val reduceList1 = emptyList.reduce { acc, i -> acc + (i\*i) }  val foldList3 = emptyList.fold(1) { acc, i -> acc + (i\*i) }  println(foldList3) // 1  val listOfNum = listOf(1, 2,3,4,5)  val chunked = listOfNum.chunked(3)  println(chunked) //[[1, 2, 3], [4, 5]]  val windowed = listOfNum.windowed(2)  println(windowed) //[[1, 2], [2, 3], [3, 4], [4, 5]]f  }  ---------------LongestCommonPrefix-------------  fun longestCommonPrefix(strs: List<String>): String {  val minLen = strs.map { it.length }.min()  for (i in minLen downTo 1) {  val res = strs[0].substring(0, i)  val diffs = strs.map { if(it.substring(0, i) == res) 0 else 1 }.sum()  if(diffs == 0) return res  }  return ""  }  fun longestCommonPrefix1(strs: Array<String>): String {  if (strs.isEmpty()) return ""  if (strs.size == 1) return strs[0]  strs.sort()  for (i in strs[0].indices) {  if (strs[0][i] != strs[strs.size - 1][i]) return strs[0].substring(0, i)  }  return strs[0]  }  fun main() {  println(longestCommonPrefix(listOf("flower", "flow", "flight")))  }  ----------------maxTrapWaterBetweenBuildings----------------  fun maxTrapBetweenBuildings(height: IntArray): Int {  var result = 0  if (height.isEmpty()) return result  val len = height.size  val arrLeft = IntArray(len)  val arrRight = IntArray(len)  var leftMax = 0  var rightMax = 0  var left = 0  var right = len - 1  while (left <= len - 1 || right >= 0) {  if (height[left] > leftMax) {  leftMax = height[left]  }  arrLeft[left] = leftMax  if (height[right] > rightMax) {  rightMax = height[right]  }  arrRight[right] = rightMax  left += 1  right -= 1  }  for (i in 0 until len) {  result += arrLeft[i].coerceAtMost(arrRight[i]) - height[i]  }  return result  }  fun main() {  println(maxTrapBetweenBuildings(intArrayOf(0,1,0,2,1,0,1,3,2,1,2,1)))  }  ----------------ThreeSum-----------------  fun threeSum(nums: IntArray): List<List<Int>> {  if(nums.size<3) return listOf()  val res = mutableListOf<List<Int>>()  for(i in 0..nums.size-3) {  for(j in i+1 .. nums.size-2) {  for(k in j+1 .. nums.size-1) {  if(i != j && j != k && i != k) {  println("$i, $j, $k")  if(nums[i] + nums[j] + nums[k] == 0) {  val newlist = listOf(nums[i], nums[j], nums[k]).sorted()  res.remove(newlist)  res.add(newlist)  }  }  }  }  }  return res  }  fun main() {  println(threeSum(listOf(-1,0,1,2,-1,-4)))  }  -------------BinarySearch-----------------  fun binarySearch(nums: List<Int>, target: Int): Boolean {  if(nums.isEmpty()) return false  val sorted = nums.sorted()  println(sorted)  var left = 0  var right = nums.size-1  var middle = left + (right - left) / 2  while(sorted[middle] != target) {  if(right - left == 1) {  return sorted[left] == target || sorted[right] == target  }  else {  if(sorted[middle] < target) {  left = middle  middle = left + (right - left) / 2  }  else {  right = middle  middle = left + (right - left) / 2  }  }  println("$left, $middle, $right")  }  return true;  }  ----------BinarySearch IsPerfectSquare-----------------  fun isPerfectSquare(num: Int): Boolean {  val list = (1..num).toList()  var left = 0  var right = list.size - 1  while(left <= right) {  var mid = left + (right - left) / 2  if(list[mid] \* list[mid] == num) {  return true  }  if(list[mid] \* list[mid] > num) {  right = mid - 1  }  else {  left = mid + 1  }  }  return false  }  ----------BinarySearch Sqrt-----------------  fun mySqrt(x: Int): Int {  if(x <= 0) return 0  val list = (1..x).toList()  var left = 1  var right = list.size  while(left <= right) {  var mid = left + (right - left) / 2  if(mid \* mid == x) {  return mid  }  if(mid \* mid > x) {  right = mid - 1  }  else {  left = mid + 1  }  println("$left, $mid, $right")  }  return left-1  }  fun peakIndexInMountainArray(arr: IntArray): Int {  if(arr.size < 3) return -1  var left = 1  var right = arr.size - 2  while(left <= right) {  var mid = left + (right - left) / 2  if(arr[mid-1] < arr[mid] && arr[mid] > arr[mid+1]) {  return mid  }  else if(arr[mid-1] > arr[mid]) {  right = mid - 1  }  else {  left = mid + 1  }  }  return -1  }  ----------HasMap/Set Demo------------  fun main() {  val luckyNumber = hashSetOf(1, 3, 4, 4, 5, 67)  println(luckyNumber) // [1, 3, 67, 4, 5]  val myCities = hashSetOf("London", "NewYork", "Paris", "London")  val badCities = hashSetOf("Miami", "NewYork")  myCities.addAll(badCities)  println(myCities) //[London, NewYork, Paris, Miami]  myCities.remove("NewYork")  println(myCities) //[London, Paris, Miami]  println(myCities.elementAt(2))  val cityRanking = HashMap<String, Int>()  cityRanking["London"] = 1  cityRanking["NewYork"] = 2  cityRanking["Paris"] = 3  cityRanking["Miami"] = 4  println(cityRanking.toString())  for (key in cityRanking.keys) {  println("Rank $key is ${cityRanking[key]}")  }  }  ----------Hash intersect or check duplicates two array-----------  fun intersect(nums1: IntArray, nums2: IntArray): IntArray {  val allIntersections = mutableListOf<Int>()  if (nums1.size <= nums2.size) {  val biggerArray = nums2.toMutableList()  nums1.forEach{  if (biggerArray.remove(it)) allIntersections.add(it)  }  } else {  val biggerArray = nums1.toMutableList()  nums2.forEach{  if (biggerArray.remove(it)) allIntersections.add(it)  }  }  return allIntersections.toIntArray()  }  fun intersect1(nums1: IntArray, nums2: IntArray): IntArray {  val res = mutableListOf<Int>()  val longArr = if(nums1.size > nums2.size) nums1.toMutableList() else nums2.toMutableList()  val shortArr = if(nums1.size > nums2.size) nums2 else nums1  shortArr.forEach {  if(longArr.remove(it)) res.add(it)  }  return res.toIntArray()  }  fun main() {  println(intersect(intArrayOf(1,2,2,1), intArrayOf(2,2)).toList().toString())  println(intersect(intArrayOf(4,9,5), intArrayOf(9,4,9,8,4)).toList().toString())  }  ----------------ContainsDuplicate Hash-------------------------------  fun containsDuplicate(nums: IntArray): Boolean {  val norepeat = HashSet<Int>()  nums.forEach {  if(!norepeat.add(it)) {  return true  }  }  return false  }  fun main() {  println(containsDuplicate(intArrayOf(1,2,3,1)))  println(containsDuplicate(intArrayOf(1,2,3,4)))  }  --------------FirstUniqChar Hash-------------------  fun firstUniqChar(s: String): Int {  if(s.length <= 0 ) return -1  if(s.length <= 1) return 0  val chars = s.toCharArray().toMutableList()  var repeatChars = HashSet<Char>()  while(chars.size > 0) {  val res = chars[0]  chars.removeAt(0)  if(chars.size <= 0) {  if(repeatChars.add(res)) {  return s.indexOf(res)  }  else {  return -1  }  }  if(chars.remove(res)) {  repeatChars.add(res)  }  else {  println(repeatChars)  if(repeatChars.add(res)) {  return s.indexOf(res)  }  }  }  return -1  }  fun main() {  // println(firstUniqChar("leetcode"))  // println(firstUniqChar("loveleetcode"))  // println(firstUniqChar("aabb"))  println(firstUniqChar("aadadaad"))  ------------WordBreak---------------  fun wordBreak2(s: String, wordDict: List<String>): Boolean {  if(s.isEmpty()) return false  val len = s.length  val wordSet = HashSet(wordDict)  val data = BooleanArray(len + 1)  data[0] = true  for(hi in 1..len){  for(lo in 0..hi){  if(data[lo] && wordSet.contains(s.substring(lo, hi))){  data[hi] = true  break  }  }  }  return data[len]  }  fun main() {  println(wordBreak2("ableetcode", listOf("leet", "code")))  println(wordBreak2("cars", listOf("car", "ca", "rs")))  }  -----------------WordSubSets---------------  fun wordSubsets(words1: Array<String>, words2: Array<String>): List<String> {  val res = mutableListOf<String>()  words1.forEach {  res.add(it)  for(i in 0..words2.size - 1) {  if(it.indexOf(words2[i]) < 0) {  res.removeLast()  break  }  }  }  return res  }  fun main() {  println(wordSubsets(arrayOf("amazon","apple","facebook","google","leetcode"), arrayOf("e","o")))  }  -----------------Rotate Array/List-------------  fun rotate(nums: IntArray, k: Int): Unit {  if (nums.isEmpty() || k == 0) return  var tempArray = IntArray(nums.size)  for (i in nums.indices) {  tempArray[(i + k) % nums.size] = nums[i]  }  for (i in nums.indices) {  nums[i] = tempArray[i]  }  return  }  fun main() {  rotate(intArrayOf(1,2,3,4,5,6,7), 3)  }  -------------MergeSort------------  fun mergeSort(list: List<Int>): List<Int> {  if(list.size <= 1) return list  fun merge(left: MutableList<Int>, right:MutableList<Int>): MutableList<Int> {  val result = mutableListOf<Int>()  while(left.isNotEmpty() && right.isNotEmpty()) {  if(left.first() < right.first()){  result.add(left.removeAt(0))  } else {  result.add(right.removeAt(0))  }  }  result.addAll(left)  result.addAll(right)  return result  }  val mid = list.size/2  val left = mergeSort(list.subList(0, mid)).toMutableList()  val right = mergeSort(list.subList(mid, list.size)).toMutableList()  return merge(left, right)  }  fun main() {  println(mergeSort(listOf(4,2,3,5,2,1,4,6,7)))  }  -------a,ab,abc LongestWord-----------  fun main() {  println( commonPrefx("appl", "banana"))  println( longestWord(arrayOf("w","wo","wor","worl","world")) )  println( longestWord(arrayOf("a","banana","app","appl","ap","apply","apple")) )  println( longestWord(arrayOf("m","mo","moc","moch","mocha","l","la","lat","latt","latte","c","ca","cat")) )  println( longestWord(arrayOf("ogz","eyj","e","ey","hmn","v","hm","ogznkb","ogzn","hmnm","eyjuo","vuq","ogznk","og","eyjuoi","d")))  println( longestWord(arrayOf("rac","rs","ra","on","r","otif","o","onpdu","rsf","rs","ot","oti","racy","onpd")))  }  fun maxString(s1: String, s2: String) = if (s1.length >= s2.length) s1 else s2  fun longestWord(words: Array<String>): String {  val sorted = words.sorted()  println(sorted)  var res = ""  var follow = ""  for (i in 0..sorted.size-1) {  if(follow.length > 0) {  if(follow.length + 1 == sorted[i].length  && sorted[i].substring(0, follow.length) == follow) {  follow = sorted[i]  if (i>=sorted.size-1) {  println(follow)  res = maxString(res, follow)  }  } else {  println(follow)  res = maxString(res, follow)  follow = commonPrefx(follow, sorted[i])  if(follow.length + 1 == sorted[i].length  && sorted[i].substring(0, follow.length) == follow) {  follow = sorted[i]  }  println("common-$follow")  }  } else {  if(sorted[i].length <= 1) {  follow = sorted[i]  }  }  }  return res  }  fun commonPrefx(s1: String, s2: String): String {  var index = 0  while (index < s1.length && index < s2.length) {  if (s1[index] != s2[index]) {  if(index>0) return s1.substring(0, index)  else return ""  }  index++  }  return if(index>0) s1.substring(0, index)  else ""  }  fun longestWord1(words: Array<String>): String {  val sorted = words.sorted()  println(sorted)  val len = words.size  val stepList = IntArray(len)  var res = ""  var step = 0  for(i in 0..len - 1) {  stepList[i] = -1  if(sorted[i].length == 1) {  step++  stepList[i] = step  } else if(i > 0) {  if(sorted[i] == sorted[i -1]) {  stepList[i] = stepList[i - 1]  } else if(sorted[i - 1].length + 1 == sorted[i].length  && sorted[i].substring(0, sorted[i - 1].length) == sorted[i - 1]) {  stepList[i] = step  } else {  stepList[i] = 0  }  }  if(i > 0) {  println(stepList.toList().toString())  if(stepList[i - 1] > 0 && stepList[i - 1] != stepList[i]) {  res = maxString(res, sorted[i - 1])  }  else if(stepList[i] > 0 && i == len - 1) {  res = maxString(res, sorted[i])  }  }  }  return res  }  -----------lo/hi IsPalindrome, getDuplicates, wordbreak------------  fun isPalindrome(str: String): Boolean {  if(str.length <= 0) return false  var lo = 0  var hi = str.length-1  while(lo <= hi) {  if(str.get(lo) != str.get(hi)) {  return false  }  lo++  hi--  }  return true  }  fun getDuplicates(str: String): MutableList<Char> {  val original = str.lowercase().toMutableList()  val removing = str.lowercase().toMutableList()  val result = mutableListOf<Char>()  while(removing.size > 0) {  val curchar = removing.removeAt(0)  original.remove(curchar)  if(original.remove(curchar)) {  result.add(curchar)  }  }  return result  }  fun wordBreak(s: String, wordDict: List<String>): Boolean {  if(s.isEmpty()) return false  val len = s.length  val wordSet = HashSet(wordDict)  val data = BooleanArray(len + 1)  data[0] = true  for(hi in 1..len){  for(lo in 0..hi){  if(data[lo] && wordSet.contains(s.substring(lo, hi))){  data[hi] = true  break  }  }  }  return data[len]  }  fun main() {  println(isPalindrome("abcba"))  println(isPalindrome("abc"))  println(getDuplicates("aabbccdd"))  println(getDuplicates("abbccDd"))  }  -------------LinkedList<T>-----------  data class Node<T>(var value: T, var next: Node<T>? = null) {  override fun toString(): String {  return if (next != null) {  "$value -> ${next.toString()}"  } else {  "$value"  }  }  }  class LinkedList<T> {  private var head: Node<T>? = null  private var tail: Node<T>? = null  private var size = 0  fun isEmpty() = size == 0  override fun toString(): String {  if(isEmpty()) {  return "Empty List"  } else {  return head.toString()  }  }  fun push(value: T) {  head = Node(value = value, next = head)  if(tail == null) {  tail = head  }  size++  }  fun append(value: T) {  if(isEmpty()) {  push(value)  return  }  tail?.next = Node(value = value)  tail = tail?.next  size ++  }  fun nodeAt(index: Int): Node<T>? {  var currentNode = head  var currentIndex = 0  while(currentNode != null && currentIndex < index) {  currentNode = currentNode.next  currentIndex++  }  return currentNode  }  fun insert(value: T, afterNode: Node<T>): Node<T> {  if(tail == afterNode) {  append(value)  return tail!!  }  val newNode = Node(value = value, next = afterNode.next)  afterNode.next = newNode  size++  return newNode  }  fun removeAt(index: Int): Node<T>? {  var currentNode = head  var currentIndex = 0  while(currentNode != null && currentIndex < index - 1) {  currentNode = currentNode.next  currentIndex++  }  if(currentNode == null) return null  if(currentNode.next == null) return null  val res = currentNode.next  currentNode.next = res?.next  size--  return res  }  }  fun main() {  val node1 = Node(value = 1)  val node2 = Node(value = 2)  val node3 = Node(value = 3)  node1.next = node2  node2.next = node3  println(node1)  val list = LinkedList<Int>()  list.append(100)  list.append(200)  list.append(300)  println(list)  }  //Teacher solution  fun longestWord(words: Array<String>): String {    val map = words.toSortedSet(Comparator { o1, o2 ->  val length = o2.length - o1.length  if (length == 0) o1.compareTo(o2) else length  })  map.forEach {  for (i in it.length downTo 1) {  if (!map.contains(it.substring(0, i))) {  break  }  if (i == 1) return (it)  }  }  return ""  }  -------------------permutation------------------  fun permute(nums: IntArray): List<List<Int>> {  val retVal: MutableList<List<Int>> = mutableListOf()  fun generate(k: Int, list: List<Int>) {  println("k:$k," + list.toString())  // If only 1 element, just output the array  if (k == 1) {  retVal.add(list.toList())  println("added" + list.toString())  } else {  for (i in 0 until k) {  generate(k - 1, list)  if (k % 2 == 0) {  swap(list, i, k - 1)  } else {  swap(list, 0, k - 1)  }  }  }  }  generate(nums.count(), nums.toList())  return retVal  }  ----------Sorted LinkedList remoteDuplicates/mergeTwoLists-----------  class ListNode(var value: Int) {  var next: ListNode? = null  fun printString() {  print("->$value");  if(next == null) {  println()  }  else {  print(next?.printString())  }  }  }  fun deleteDuplicates(head: ListNode?): ListNode? {  var curNode = head  if(curNode == null) return head  while(curNode?.next != null) {  if(curNode.value == curNode.next?.value) {  curNode.next = curNode.next?.next  }  else {  curNode = curNode.next  }  }  return head  }  fun mergeTwoLists(list1: ListNode?, list2: ListNode?): ListNode? {  var head1 = list1  var head2 = list2  var newList:ListNode? = null  var newHead:ListNode? = null  while(head1 != null || head2 != null) {  var processNode:ListNode? = null  if(head1 == null) {  processNode = head2  head2 = head2?.next  } else if(head2 == null) {  processNode = head1  head1 = head1?.next  }  else {  if(head1.value < head2.value) {  processNode = head1  head1 = head1.next  }  else if(head1.value > head2.value) {  processNode = head2  head2 = head2.next  }  else {  processNode = ListNode(head1.value)  head1 = head1.next  head2 = head2.next  }  }  processNode?.let {  if(newList == null){  newList = ListNode(processNode.value)  if(newHead == null) {  newHead = newList  }  } else {  newList?.next = ListNode(processNode.value)  newList = newList?.next  }  }  }  return newHead  }  fun main() {  val list = mutableListOf<ListNode>()  for(i in 0..5) {  list.add(ListNode(i))  }  for(i in 0..4) {  list[i].next = list[i+1]  }  list[2].value = 1  list[0].printString()  println()  deleteDuplicates(list[0])  list[0].printString()  println()  val list2 = mutableListOf<ListNode>()  for(i in 3..8) {  list2.add(ListNode(i))  }  for(i in 0..list2.lastIndex-1) {  list2[i].next = list2[i+1]  }  list2[0].printString()  println()  val mergedList = mergeTwoLists(list[0], list2[0])  mergedList?.printString()  println()  }  -----------------generateCombination---------------  fun generateCombination(n: Int, r: Int): List<IntArray>? {  val combinations: MutableList<IntArray> = ArrayList()  val combination = IntArray(r)  // initialize with lowest lexicographic combination  for (i in 0 until r) {  combination[i] = i  }  while (combination[r - 1] < n) {  combinations.add(combination.clone())  // generate next combination in lexicographic order  var t = r - 1  while (t != 0 && combination[t] == n - r + t) {  t--  }  combination[t]++  for (i in t + 1 until r) {  combination[i] = combination[i - 1] + 1  }  }  combinations.map {  println(it.toList().toString())  }  return combinations  }  fun main() {  println(threeSum(intArrayOf(3,5,2,3,6,4,6,2,-6)))  // println(combinationSum(intArrayOf(3,5,2,3,6,4,6,2,-6), 8))  generateCombination(9, 3)  }  ---------------MaximumBinaryTree/UniqueBinarySearchTrees-----------  class TreeNode(var `val`: Int) {  var left: TreeNode? = null  var right: TreeNode? = null  }  --------------constructMaximumBinaryTree----------  fun constructMaximumBinaryTree(nums: IntArray): TreeNode? {  if(nums == null || nums.isEmpty()) return null  var lo = 0  var hi = nums.size-1  var max = 0  while (lo <= hi) {  if(max < nums[lo] ) {  max = nums[lo]  }  if (max < nums[hi]) {  max = nums[hi]  }  lo++  hi--  }  val maxNo = nums.indexOf(max)  val ti = TreeNode(max)  ti.left = if (maxNo-1 >= 0) constructMaximumBinaryTree(nums.copyOfRange(0, maxNo)) else null  ti.right = if (nums.size-1 >= maxNo+1) constructMaximumBinaryTree(nums.copyOfRange(maxNo+1, nums.size)) else null  return ti  }  --------------numTrees------------  fun numTrees(n: Int): Int {  if (n == 0) return 1  var sum = 0  for (i in 0 until n) {  sum += (numTrees(i) \* numTrees(n - 1 - i))  }  return sum  }  fun main() {  constructMaximumBinaryTree(intArrayOf(1,2,3))  numTrees(10)  }  ----------mutiplyOriginal------------  fun multiplyOriginal(num1: String, num2: String): String {  val digits1 = num1.toCharArray().reversed().map { it.toString().toInt() }  val digits2 = num2.toCharArray().reversed().map { it.toString().toInt() }  var sum: BigInteger = "0".toBigInteger()  digits1.forEachIndexed() { id1, num1 ->  digits2.forEachIndexed { id2, num2 ->  var x = (num1 \* num2).toString() + List<String>(id1+id2){"0"}.joinToString("")  println(x)  sum += x.toBigInteger()  }  }  return sum.toString()  }  fun main() {  println(multiplyOriginal ("526", "534"))  } | </end> |
| <hitle> | AndroidLayouts | <chare> | 1 | <pext> | 01-16/ AndroidLayouts  ---------------LinearLayout Demo-----------------  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:orientation="vertical"  android:weightSum="4"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <LinearLayout  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="@color/teal\_200"  android:weightSum="2"  >  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="1"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="2"  android:textSize="40sp"  />  </LinearLayout>  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="1"  android:textSize="40sp"  />  <LinearLayout  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:weightSum="3"  android:background="#8F4040"  >  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="1"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="2"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="3"  android:textSize="40sp"  />  </LinearLayout>  <LinearLayout  android:background="#B2B678"  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:weightSum="6"  >  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="1"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="2"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="3"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="4"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="5"  android:textSize="40sp"  />  <Button  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="6"  android:textSize="40sp"  />  </LinearLayout>  </LinearLayout>  ------------------RelativeLayout Demo--------------  <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <Button  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="1"/>  <Button  android:layout\_alignParentRight="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="2"/>  <Button  android:layout\_alignParentBottom="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="3"/>  <Button  android:layout\_alignParentBottom="true"  android:layout\_alignParentRight="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="4"/>  <Button  android:layout\_centerInParent="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="5"/>  <Button  android:layout\_centerHorizontal="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="6"/>  <Button  android:layout\_centerVertical="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="7"/>  <Button  android:layout\_centerVertical="true"  android:layout\_alignParentEnd="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="8"/>  <Button  android:layout\_centerHorizontal="true"  android:layout\_alignParentBottom="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="9"/>  </RelativeLayout>  <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <Button  android:id="@+id/button1"  android:layout\_centerInParent="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="1"/>  <Button  android:layout\_centerHorizontal="true"  android:layout\_above="@+id/button1"  android:layout\_centerInParent="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="2"/>  <Button  android:layout\_centerHorizontal="true"  android:layout\_below="@+id/button1"  android:layout\_centerInParent="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="3"/>  <Button  android:layout\_centerVertical="true"  android:layout\_toLeftOf="@+id/button1"  android:layout\_centerInParent="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="3"/>  <Button  android:layout\_centerVertical="true"  android:layout\_toRightOf="@+id/button1"  android:layout\_centerInParent="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:text="4"/>  </RelativeLayout>  <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <Button  android:id="@+id/button1"  android:layout\_centerVertical="true"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="1"/>  <Button  android:id="@+id/button2"  android:layout\_above="@id/button1"  android:layout\_toEndOf="@id/button1"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="2"/>  <Button  android:id="@+id/button3"  android:layout\_below="@id/button1"  android:layout\_toEndOf="@id/button1"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="3"/>  <Button  android:id="@+id/button4"  android:layout\_above="@id/button2"  android:layout\_toEndOf="@id/button2"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="4"/>  <Button  android:id="@+id/button5"  android:layout\_below="@id/button3"  android:layout\_toEndOf="@id/button3"  android:layout\_width="40dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="5"/>  </RelativeLayout>  ---------------- RelativeLayout Login Practice------------  <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <TextView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textSize="40dp"  android:textAlignment="center"  android:text="Login"  android:layout\_above="@id/editEmail"  android:layout\_marginBottom="20dp"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editEmail"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_above="@id/editPassword"  android:layout\_margin="20dp"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editPassword"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_above="@id/btnLogin"  android:layout\_margin="20dp"  />  <Button  android:id="@+id/btnLogin"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Login"  android:textSize="20sp"  android:layout\_alignBottom="@id/centerLine"  android:layout\_centerInParent="true"  android:layout\_marginBottom="30dp"  />  <View  android:id="@+id/centerLine"  android:layout\_width="match\_parent"  android:layout\_height="10dp"  android:layout\_centerInParent="true"  />  <TextView  android:layout\_below="@id/centerLine"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textAlignment="center"  android:layout\_margin="20dp"  android:text="Login"  android:textSize="30dp"  android:id="@+id/bottomLogin"  />  </RelativeLayout>  ---------------- RelativeLayout MovieDetailPractice------------  <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <ImageView  android:id="@+id/imgProfile"  android:layout\_width="140dp"  android:layout\_height="200dp"  android:layout\_alignParentStart="true"  android:layout\_margin="10dp"  android:src="@color/black" />  <TextView  android:id="@+id/titleText"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Music"  android:textSize="20dp"  android:layout\_toEndOf="@+id/imgProfile"  android:layout\_margin="10dp"  />  <TextView  android:id="@+id/topText1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Action"  android:layout\_below="@+id/titleText"  android:layout\_toEndOf="@id/imgProfile"  android:layout\_margin="10dp"  />  <TextView  android:id="@+id/topText2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_toEndOf="@+id/topText1"  android:layout\_below="@+id/titleText"  android:layout\_margin="10dp"  android:text="Crime"  />  <TextView  android:id="@+id/topText3"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Drama"  android:layout\_toEndOf="@+id/topText2"  android:layout\_below="@+id/titleText"  android:layout\_margin="10dp"  />  <Button  android:id="@+id/topButton1"  android:layout\_width="50dp"  android:layout\_height="wrap\_content"  android:text="1"  android:layout\_toEndOf="@id/imgProfile"  android:layout\_below="@id/topText1"  android:layout\_margin="10dp"  />  <Button  android:id="@+id/topButton2"  android:layout\_width="50dp"  android:padding="0dp"  android:layout\_height="wrap\_content"  android:text="2"  android:layout\_toEndOf="@id/topButton1"  android:layout\_below="@id/topText1"  android:layout\_margin="10dp"  />  <Button  android:id="@+id/topButton3"  android:layout\_width="50dp"  android:layout\_height="wrap\_content"  android:text="3"  android:layout\_toEndOf="@id/topButton2"  android:layout\_below="@id/topText1"  android:layout\_margin="10dp"  />  <Button  android:id="@+id/topButton4"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="4"  android:layout\_below="@id/topButton1"  android:layout\_toEndOf="@id/imgProfile"  android:layout\_margin="10dp"  />  <Button  android:id="@+id/topButton5"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="5"  android:layout\_toEndOf="@id/topButton4"  android:layout\_below="@id/topButton1"  android:layout\_margin="10dp"  />  <RelativeLayout  android:layout\_below="@id/imgProfile"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  >  <TextView  android:id="@+id/bottomText1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Music Dsecription"  android:textSize="16sp"  android:layout\_margin="10dp"  android:layout\_alignParentStart="true"  />  <TextView  android:id="@+id/bottomText2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/bottomText1"  android:text="Do you want to take screenshots outside of your browser? Choose the most functional Lightshot download option in order to get this opportunity. This application allows you to take screenshots directly from your desktop."  android:layout\_margin="10dp"  />  <ImageView  android:id="@+id/star1"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_below="@+id/bottomText2"  android:layout\_margin="5dp"  android:src="@mipmap/ic\_star\_full"  />  <ImageView  android:id="@+id/star2"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_below="@+id/bottomText2"  android:layout\_toEndOf="@+id/star1"  android:layout\_margin="5dp"  android:src="@mipmap/ic\_star\_full"  />  <ImageView  android:id="@+id/star3"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_below="@+id/bottomText2"  android:layout\_toEndOf="@+id/star2"  android:layout\_margin="5dp"  android:src="@mipmap/ic\_star\_full"  />  <ImageView  android:id="@+id/star4"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_below="@+id/bottomText2"  android:layout\_toEndOf="@+id/star3"  android:layout\_margin="5dp"  android:src="@mipmap/ic\_star\_full"  />  <ImageView  android:id="@+id/star5"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_below="@+id/bottomText2"  android:layout\_toEndOf="@+id/star4"  android:layout\_margin="5dp"  android:src="@mipmap/ic\_star\_full"  />  <ImageView  android:id="@+id/imgTrailer"  android:layout\_width="20dp"  android:layout\_height="20dp"  android:layout\_below="@+id/star1"  android:layout\_margin="10dp"  android:layout\_marginRight="0dp"  android:src="@mipmap/ic\_trailer"  />  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="14sp"  android:text="Trailer"  android:layout\_below="@id/star1"  android:layout\_toEndOf="@+id/imgTrailer"  android:layout\_marginLeft="0dp"  android:layout\_marginTop="10dp"  android:layout\_marginBottom="10dp"  />  </RelativeLayout>  </RelativeLayout>  ---------------ConstraintLayout Demo-----------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30dp"  android:text="1"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30dp"  android:text="2"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30dp"  android:text="3"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30dp"  android:text="4"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30dp"  android:text="5"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <Button  android:id="@+id/button1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  android:text="1"  android:textSize="20dp"  android:layout\_margin="10dp"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="2"  android:textSize="20dp"  app:layout\_constraintEnd\_toStartOf="@+id/button1"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  android:layout\_margin="10dp"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="3"  android:textSize="20dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toTopOf="@+id/button1"  android:layout\_margin="10dp"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="4"  android:textSize="20dp"  app:layout\_constraintStart\_toEndOf="@id/button1"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  android:layout\_margin="10dp"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="5"  android:textSize="20dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/button1"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_margin="10dp"  />  </androidx.constraintlayout.widget.ConstraintLayout>  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <Button  android:id="@+id/button1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintLeft\_toLeftOf="parent"  android:text="1"  android:textSize="20dp"  />  <Button  android:id="@+id/button2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintLeft\_toRightOf="@+id/button1"  app:layout\_constraintBottom\_toTopOf="@+id/button1"  android:text="2"  android:textSize="20dp"  />  <Button  android:id="@+id/button3"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@+id/button1"  app:layout\_constraintLeft\_toRightOf="@+id/button1"  android:text="3"  android:textSize="20dp"  />  <Button  android:id="@+id/button4"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintLeft\_toRightOf="@+id/button2"  app:layout\_constraintBottom\_toTopOf="@+id/button2"  android:text="4"  android:textSize="20dp"  />  <Button  android:id="@+id/button5"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintLeft\_toRightOf="@+id/button3"  app:layout\_constraintTop\_toBottomOf="@+id/button3"  android:text="4"  android:textSize="20dp"  />  </androidx.constraintlayout.widget.ConstraintLayout>  ---------------ConstraintLayout LoginDemo----------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="50dp"  android:text="Login"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@id/editName"  android:textColor="@color/black"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editName"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginLeft="20dp"  android:layout\_marginRight="20dp"  android:hint="\*\*\*\*\*\*\*\*\*"  app:layout\_constraintBottom\_toTopOf="@+id/editPassword"  android:drawableLeft="@drawable/baseline\_account\_box\_24"  android:drawablePadding="10dp"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editPassword"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginLeft="20dp"  android:layout\_marginRight="20dp"  android:hint="\*\*\*\*\*\*\*\*\*"  app:layout\_constraintBottom\_toTopOf="@+id/rememberSet"  android:drawableLeft="@drawable/baseline\_key\_24"  android:drawablePadding="10dp"  />  <Button  android:id="@+id/btnLogin"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Log in"  android:textSize="20sp"  android:backgroundTint="#23CAD8"  app:layout\_constraintBottom\_toTopOf="@+id/centerLine"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  <androidx.constraintlayout.widget.ConstraintLayout  android:id="@+id/rememberSet"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginLeft="20dp"  android:layout\_marginRight="20dp"  app:layout\_constraintBottom\_toTopOf="@id/btnLogin"  >  <CheckBox  android:id="@+id/chkRemeber"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Remember Me"  app:layout\_constraintStart\_toEndOf="@id/chkRemeber"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  android:textColor="#5E5B5B"  />  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Forgot Password?"  android:textColor="#FF0000"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  <View  android:id="@+id/centerLine"  android:layout\_width="match\_parent"  android:layout\_height="10dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <TextView  android:id="@+id/txtSmallLogin"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="30dp"  android:text="Login"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="@id/centerLine"  app:layout\_constraintBottom\_toTopOf="@id/txtLoginFollow"  android:textColor="@color/black"  />  <TextView  android:id="@+id/txtLoginFollow"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="With your social media account"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/txtSmallLogin"  app:layout\_constraintBottom\_toTopOf="@+id/socialSet"  android:textColor="@color/black"  />  <androidx.constraintlayout.widget.ConstraintLayout  android:id="@+id/socialSet"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/txtLoginFollow"  app:layout\_constraintBottom\_toTopOf="@+id/dontSet"  android:layout\_marginLeft="20dp"  android:layout\_marginRight="20dp"  >  <Button  android:id="@+id/btnTwitter"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  android:text="Twitter"  android:textSize="20sp"  android:backgroundTint="@color/teal\_200"  android:padding="10dp"  />  <Button  android:id="@+id/btnFacebook"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@+id/btnTwitter"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/btnGoogle"  android:text="Twitter"  android:textSize="20sp"  android:backgroundTint="#273251"  android:padding="10dp"  />  <Button  android:id="@+id/btnGoogle"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@+id/btnFacebook"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:text="Twitter"  android:textSize="20sp"  android:backgroundTint="#A84545"  android:padding="10dp"  />  </androidx.constraintlayout.widget.ConstraintLayout>  <LinearLayout  android:id="@+id/dontSet"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/socialSet"  app:layout\_constraintBottom\_toTopOf="@+id/btnRegister"  >  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Don't have an Account?"  />  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Register Now!"  android:textColor="#FF0000"  android:layout\_marginLeft="10dp"  />  </LinearLayout>  <Button  android:id="@+id/btnRegister"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Register"  android:textSize="20sp"  android:backgroundTint="#23CAD8"  app:layout\_constraintTop\_toBottomOf="@id/dontSet"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  ---------------ConstraintLayout Practice1------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@color/black"  xmlns:app="http://schemas.android.com/apk/res-auto">    <androidx.constraintlayout.widget.Guideline  android:id="@+id/glVert5"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="vertical"  app:layout\_constraintGuide\_percent="0.05"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glVert50"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="vertical"  app:layout\_constraintGuide\_percent="0.5"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glVert95"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="vertical"  app:layout\_constraintGuide\_percent="0.95"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori20"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.2"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori50"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.5"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori80"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.8"  />  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="@+id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  app:layout\_constraintBottom\_toTopOf="@+id/editSearch"  >  <androidx.constraintlayout.widget.Guideline  android:id="@+id/proGLVert"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.5"  />  <ImageView  android:id="@+id/imgProfile"  android:layout\_width="70dp"  android:layout\_height="70dp"  android:src="@drawable/baseline\_account\_box\_24"  android:layout\_marginBottom="10dp"  android:layout\_marginRight="10dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <ImageView  android:id="@+id/imgNoti"  android:layout\_width="40dp"  android:layout\_height="40dp"  android:src="@drawable/baseline\_alarm\_24"  android:layout\_marginBottom="10dp"  android:layout\_marginRight="10dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  <TextView  android:id="@+id/txtAurthName"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Hello Jacob"  android:textColor="@color/white"  android:textSize="20sp"  app:layout\_constraintStart\_toEndOf="@+id/imgProfile"  app:layout\_constraintBottom\_toTopOf="@+id/proGLVert"  />  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="He is a good musician"  android:textColor="@color/white"  android:textSize="16sp"  app:layout\_constraintStart\_toEndOf="@+id/imgProfile"  app:layout\_constraintTop\_toBottomOf="@+id/proGLVert"  />  </androidx.constraintlayout.widget.ConstraintLayout>  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editSearch"  android:layout\_width="0dp"  android:layout\_height="50dp"  android:hint="Search movie..."  android:textColor="@color/white"  app:layout\_constraintBottom\_toTopOf="@+id/glHori20"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  android:drawableLeft="@drawable/baseline\_account\_box\_24"  android:drawablePadding="10dp"  android:background="#373434"  android:foregroundTint="@color/white"  android:layout\_marginBottom="10dp"  />  <TextView  android:id="@+id/txtPopular"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Popular movies"  android:textColor="@color/white"  android:textSize="26sp"  app:layout\_constraintStart\_toEndOf="@+id/glVert5"  app:layout\_constraintTop\_toBottomOf="@+id/glHori20"  />  <ImageView  android:id="@+id/imgMain"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:src="@drawable/ic\_launcher\_background"  app:layout\_constraintStart\_toStartOf="@+id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  app:layout\_constraintTop\_toBottomOf="@id/txtPopular"  android:scaleType="centerCrop"  />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Popular movies"  android:textColor="@color/white"  android:textSize="22sp"  app:layout\_constraintStart\_toEndOf="@+id/glVert5"  app:layout\_constraintTop\_toBottomOf="@+id/imgMain"  />  <TextView  android:id="@+id/txtDesc"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Popular movies"  android:textColor="@color/white"  android:textSize="16sp"  app:layout\_constraintStart\_toEndOf="@+id/glVert5"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  app:layout\_constraintBottom\_toTopOf="@id/glHori50"  />  <Button  android:layout\_width="0dp"  android:layout\_height="60dp"  android:text="Button"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintEnd\_toEndOf="@+id/glVert95"  app:layout\_constraintTop\_toTopOf="@id/glHori80"  app:layout\_constraintBottom\_toBottomOf="parent"  android:backgroundTint="#FF0000"  />  <TextView  android:id="@+id/txtTVShows"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="TV Shows"  android:textColor="@color/white"  android:textSize="22sp"  app:layout\_constraintTop\_toBottomOf="@id/glHori50"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintBottom\_toTopOf="@+id/imgSub1"  />  <ImageView  android:id="@+id/imgSub1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:src="@drawable/ic\_launcher\_background"  app:layout\_constraintStart\_toStartOf="@+id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert50"  app:layout\_constraintTop\_toBottomOf="@id/txtTVShows"  app:layout\_constraintBottom\_toTopOf="@+id/txtSubTitle1"  android:scaleType="centerCrop"  android:layout\_marginRight="5dp"  />  <TextView  android:id="@+id/txtSubTitle1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Stranger Things"  android:textColor="@color/white"  android:textSize="16sp"  app:layout\_constraintTop\_toBottomOf="@id/imgSub1"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert50"  app:layout\_constraintBottom\_toTopOf="@+id/txtSubDesc1"  />  <TextView  android:id="@+id/txtSubDesc1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Stranger Things"  android:textColor="@color/white"  android:textSize="12sp"  app:layout\_constraintTop\_toBottomOf="@id/txtSubTitle1"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert50"  app:layout\_constraintBottom\_toTopOf="@id/glHori80"  />  <ImageView  android:id="@+id/imgSub2"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:src="@drawable/ic\_launcher\_background"  app:layout\_constraintStart\_toStartOf="@+id/glVert50"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  app:layout\_constraintTop\_toTopOf="@id/imgSub1"  android:scaleType="centerCrop"  android:layout\_marginLeft="5dp"  />  <TextView  android:id="@+id/txtSubTitle2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Stranger Things"  android:textColor="@color/white"  android:textSize="16sp"  app:layout\_constraintTop\_toBottomOf="@id/imgSub2"  app:layout\_constraintStart\_toStartOf="@id/glVert50"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  app:layout\_constraintBottom\_toTopOf="@+id/txtSubDesc2"  />  <TextView  android:id="@+id/txtSubDesc2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Stranger Things"  android:textColor="@color/white"  android:textSize="12sp"  app:layout\_constraintTop\_toBottomOf="@id/txtSubTitle2"  app:layout\_constraintStart\_toStartOf="@id/glVert50"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  app:layout\_constraintBottom\_toTopOf="@id/glHori80"  />  </androidx.constraintlayout.widget.ConstraintLayout>  -----------------ConstraintLayout Practice2-----------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@color/black"  xmlns:app="http://schemas.android.com/apk/res-auto">  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glVert5"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="vertical"  app:layout\_constraintGuide\_percent="0.05"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glVert50"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="vertical"  app:layout\_constraintGuide\_percent="0.5"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glVert95"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="vertical"  app:layout\_constraintGuide\_percent="0.95"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori20"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.2"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori50"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.5"  />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori80"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  app:layout\_constraintGuide\_percent="0.8"  />  <ImageView  android:layout\_width="match\_parent"  android:layout\_height="0dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@+id/glHori50"  android:src="@drawable/ic\_launcher\_background"  android:scaleType="centerCrop"  />  <Button  android:layout\_width="0dp"  android:layout\_height="60dp"  android:text="Button"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintEnd\_toEndOf="@+id/glVert95"  app:layout\_constraintTop\_toTopOf="@id/glHori80"  app:layout\_constraintBottom\_toBottomOf="parent"  android:backgroundTint="#FF0000"  />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Popular movies"  android:textColor="@color/white"  android:textSize="26sp"  app:layout\_constraintStart\_toEndOf="@+id/glVert5"  app:layout\_constraintTop\_toBottomOf="@+id/glHori50"  app:layout\_constraintBottom\_toTopOf="@id/txtCat"  />  <TextView  android:id="@+id/txtCat"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Popular movies"  android:textColor="@color/white"  android:textSize="10sp"  app:layout\_constraintStart\_toEndOf="@+id/glVert5"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  app:layout\_constraintBottom\_toTopOf="@id/txtContent"  />  <TextView  android:id="@+id/txtContent"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Popular movies This is great movie I have ever seeen you ccould please see what it is hello saojfsajfdosjdf sfjaosjdf sdj osijfoas sojfoasfjd oijsdf sifjs sdfjos "  android:textColor="@color/white"  android:textSize="18sp"  app:layout\_constraintStart\_toEndOf="@+id/glVert5"  app:layout\_constraintTop\_toBottomOf="@+id/txtCat"  app:layout\_constraintBottom\_toTopOf="@+id/buttonSet"  />  <androidx.constraintlayout.widget.ConstraintLayout  android:id="@+id/buttonSet"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="@id/glVert5"  app:layout\_constraintEnd\_toEndOf="@id/glVert95"  app:layout\_constraintTop\_toBottomOf="@id/txtContent"  app:layout\_constraintBottom\_toTopOf="@id/glHori80"  >  <Button  android:id="@+id/button1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="16+"  android:padding="0dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <Button  android:id="@id/button2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="2016"  android:padding="0dp"  app:layout\_constraintStart\_toEndOf="@id/button1"  app:layout\_constraintEnd\_toStartOf="@+id/button3"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <Button  android:id="@id/button3"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="8.8"  android:padding="0dp"  app:layout\_constraintStart\_toEndOf="@id/button2"  app:layout\_constraintEnd\_toStartOf="@+id/button4"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <Button  android:id="@id/button4"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="45-49min"  android:padding="0dp"  app:layout\_constraintStart\_toEndOf="@id/button3"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.constraintlayout.widget.ConstraintLayout>  -------------Barrier Line Demo---------------  <?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">  <TextView  android:id="@+id/txtLeft"  android:layout\_width="200dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:background="#DD221E"  android:textColor="#FFFFFF"  android:textSize="20sp"  android:text="hey i am here how are youd/fgd;l,g;dl,g;dl,g;fld,;g,d;lf,g;dl,fg;ld,fg"  app:layout\_constraintEnd\_toStartOf="@+id/txtRight"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtRight"  android:layout\_width="200dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:background="#0521CF"  android:textColor="#FFFFFF"  android:text="sdfas f laskjfdoasijdf oaijflksj fo sdfij alsjdf a faisd flajksfd oa fk faaf askdf oasf faksf f klf a faf sklf sl fl fso sfk "  android:textSize="20sp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/txtLeft"  app:layout\_constraintTop\_toTopOf="parent" />    <androidx.constraintlayout.widget.Barrier  android:id="@+id/barrier1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:barrierDirection="bottom"  app:constraint\_referenced\_ids="txtLeft,txtRight" />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/barrier1" />  </androidx.constraintlayout.widget.ConstraintLayout>  ------------TableLayout Demo--------------  <?xml version="1.0" encoding="utf-8"?>  <TableLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Country Cricket Ranking"  android:textSize="30sp"  />  <TableRow>  <TextView  android:layout\_weight="1"  android:text="Rank"  android:textSize="20sp"  android:background="@color/purple\_700"  android:textColor="@color/white"  />  <TextView  android:layout\_weight="1"  android:text="Country"  android:textSize="20sp"  android:background="@color/purple\_700"  android:textColor="@color/white"  />  <TextView  android:layout\_weight="1"  android:text="Player"  android:textSize="20sp"  android:background="@color/purple\_700"  android:textColor="@color/white"  />  <TextView  android:layout\_weight="1"  android:text="Average"  android:textSize="20sp"  android:background="@color/purple\_700"  android:textColor="@color/white"  />  </TableRow>  <TableRow>  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="1"  android:background="#D8D7D7"  />  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="USA"  android:background="#D8D7D7"  />  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="Player1"  android:background="#D8D7D7"  />  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="63.00"  android:background="#D8D7D7"  />  </TableRow>  <TableRow>  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="2"  android:background="#D8D7D7"  />  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="India"  android:background="#D8D7D7"  />  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="Player2"  android:background="#D8D7D7"  />  <TextView  android:layout\_weight="1"  android:textSize="16sp"  android:text="63.00"  android:background="#D8D7D7"  />  </TableRow>  </TableLayout>  -------------FrameLayout Demo Obama player--------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <FrameLayout  android:layout\_width="match\_parent"  android:layout\_height="300dp"  app:layout\_constraintTop\_toTopOf="parent"  >  <ImageView  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:scaleType="centerCrop"  android:src="@drawable/obama"  />  <LinearLayout  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  android:layout\_gravity="center"  >  <ImageView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:elevation="10dp"  android:scaleType="centerCrop"  android:src="@drawable/baseline\_play\_circle\_filled\_24"  android:layout\_gravity="center"  android:layout\_margin="20dp"  />  <ImageView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:elevation="10dp"  android:scaleType="centerCrop"  android:src="@drawable/baseline\_play\_circle\_filled\_24"  android:layout\_gravity="center"  android:layout\_margin="20dp"  />  <ImageView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:elevation="10dp"  android:scaleType="centerCrop"  android:src="@drawable/baseline\_play\_circle\_filled\_24"  android:layout\_gravity="center"  android:layout\_margin="20dp"  />  </LinearLayout>  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  android:layout\_gravity="bottom"  >  <TextView  android:id="@+id/playingTime"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="00:00"  android:textColor="@color/white"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent" />  <SeekBar  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@id/playingTime"  app:layout\_constraintEnd\_toStartOf="@id/totalTime"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"/>  <TextView  android:id="@+id/totalTime"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="50:00"  android:textColor="@color/white"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toStartOf="@id/btnFullscreen" />  <ImageView  android:id="@+id/btnFullscreen"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:elevation="10dp"  android:scaleType="centerCrop"  android:src="@drawable/baseline\_fullscreen\_24"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  </FrameLayout>  </androidx.constraintlayout.widget.ConstraintLayout>  ---------------custom buttons demo-------------------  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="@dimen/size\_10dp"  android:text="@string/style\_1"  android:background="@color/background\_for\_style\_1"  style="@style/custom\_button\_Style"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <resources xmlns:tools="http://schemas.android.com/tools">  <!-- Base application theme. -->  <style name="Theme.AndroidLayouts" parent="Theme.MaterialComponents.DayNight.DarkActionBar">  <!-- Primary brand color. -->  <item name="colorPrimary">@color/purple\_500</item>  <item name="colorPrimaryVariant">@color/purple\_700</item>  <item name="colorOnPrimary">@color/white</item>  <!-- Secondary brand color. -->  <item name="colorSecondary">@color/teal\_200</item>  <item name="colorSecondaryVariant">@color/teal\_700</item>  <item name="colorOnSecondary">@color/black</item>  <!-- Status bar color. -->  <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>  <!-- Customize your theme here. -->  </style>  <style name="custom\_button\_Style" parent="TextAppearance.AppCompat.Button"  >  <item name="android:textColor">@color/primary\_text\_color</item>  <item name="android:fontFamily">sans-serif</item>  <item name="android:textSize">@dimen/size\_20sp</item>  <item name="android:textStyle">bold</item>  </style>>  </resources>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="@dimen/size\_20sp"  android:layout\_margin="@dimen/size\_10dp"  android:text="@string/style\_2"  android:textColor="@color/secondary\_text\_color"  android:background="@drawable/secondary\_button\_background"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android">  <stroke  android:color="@color/background\_for\_style\_1"  android:dashWidth="1dp"  />  <solid  android:color="@color/primary\_text\_color"  />  </shape>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="@dimen/size\_20sp"  android:layout\_margin="@dimen/size\_10dp"  android:text="@string/style\_3"  android:textColor="@color/primary\_text\_color"  android:background="@drawable/button\_style3"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android">  <solid  android:color="@color/style\_3"  />  <corners  android:radius="@dimen/size\_80dp"  />  </shape>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="@dimen/size\_20sp"  android:layout\_margin="@dimen/size\_10dp"  android:text="@string/style\_4"  android:textColor="@color/secondary\_text\_color"  android:background="@drawable/button\_style4"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android">  <stroke  android:width="2dp"  android:color="@color/background\_for\_style\_1"  android:dashWidth="@dimen/size\_10dp"  android:dashGap="@dimen/size\_10dp"  />  <solid  android:color="@color/primary\_text\_color"  />  </shape>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="@dimen/size\_20sp"  android:layout\_margin="@dimen/size\_10dp"  android:text="@string/style\_5"  android:textColor="@color/secondary\_text\_color"  android:background="@drawable/button\_style5"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item android:state\_pressed="true">  <shape android:shape="rectangle">  <corners  android:radius="@dimen/size\_80dp"  />  <stroke  android:width="2dp"  android:color="@color/background\_for\_style\_1"  android:dashWidth="@dimen/size\_10dp"  android:dashGap="@dimen/size\_10dp"  />  <solid  android:color="@color/primary\_text\_color"  />  </shape>  </item>  <item android:state\_pressed="false">  <shape>  <solid  android:color="@color/style\_3"  />  <corners  android:radius="@dimen/size\_80dp"  />  </shape>  </item>  </selector>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="@dimen/size\_20sp"  android:layout\_margin="@dimen/size\_10dp"  android:text="@string/style\_6"  android:textColor="@color/primary\_text\_color"  android:background="@drawable/button\_style6"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android">  <gradient  android:startColor="@color/background\_for\_style\_1"  android:centerColor="@color/secondary\_text\_color"  android:endColor="@color/primary\_text\_color"  />  <corners  android:radius="@dimen/size\_80dp"  />  </shape>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="@dimen/size\_20sp"  android:layout\_margin="@dimen/size\_10dp"  android:padding="@dimen/size\_20dp"  android:text="@string/style\_7"  android:textColor="@color/primary\_text\_color"  android:background="@drawable/button\_style7"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android"  android:shape="rectangle"  >  <solid  android:color="@color/style\_7"  />  <corners  android:bottomLeftRadius="@dimen/size\_20dp"  android:bottomRightRadius="@dimen/size\_20dp"  android:topRightRadius="@dimen/size\_20dp"  />  </shape>  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <androidx.appcompat.widget.AppCompatButton  android:clickable="true"  android:focusable="true"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="@dimen/size\_10dp"  android:background="@drawable/button\_style8"  />  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item  android:state\_pressed="false"  android:drawable="@drawable/baseline\_wifi\_24"  />  <item  android:state\_pressed="true"  android:drawable="@drawable/baseline\_wifi\_off\_24"  />  </selector> | </end> |
| <hitle> | SharedPreferenceDemo Login | <chare> | 1 | <pext> | 01-24/SharedPreferenceDemo  class SplashActivity : AppCompatActivity() {  private lateinit var sharedPreferences: SharedPreferences  private lateinit var editor: SharedPreferences.Editor  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_splash)  doInitializeSharedPref()  checkWhereToGo()  }  private fun checkWhereToGo() {  if (sharedPreferences.getString(EMAIL, "") == "") {  startActivity(Intent(this, MainActivity::class.java ))  } else {  startActivity(Intent(this, DashboardActivity::class.java))  }  }  private fun doInitializeSharedPref() {  sharedPreferences = getSharedPreferences(MainActivity.LOGIN\_FILE, MODE\_PRIVATE)  editor = sharedPreferences.edit()  }  }  class MainActivity : AppCompatActivity() {  private lateinit var sharedPreferences: SharedPreferences  private lateinit var editor: SharedPreferences.Editor  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  doInitializeSharedPref()  setUpViews()  }  private fun setUpViews() {  val editEmail = findViewById<EditText>(R.id.editEmail)  val editPassword = findViewById<EditText>(R.id.editPassword)  val btnLogin = findViewById<Button>(R.id.btnLogin)  btnLogin.setOnClickListener {  val email = editEmail.text.toString()  val password = editPassword.text.toString()  doLogin(email, password)  editEmail.text?.clear()  editPassword.text?.clear()  startActivity(Intent(this, DashboardActivity::class.java))  }  }  private fun doLogin(email: String, password: String) {  editor.apply {  putString(EMAIL, email)  putString(PASSWORD, password)  apply()  }  }  private fun doInitializeSharedPref() {  sharedPreferences = getSharedPreferences(LOGIN\_FILE, MODE\_PRIVATE)  editor = sharedPreferences.edit()  }  companion object {  const val LOGIN\_FILE = "loginFile"  const val EMAIL = "email"  const val PASSWORD = "password"  }  }  class DashboardActivity : AppCompatActivity() {  private lateinit var sharedPreferences: SharedPreferences  private lateinit var editor: SharedPreferences.Editor  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_dashboard)  doInitializeSharedPref()  findViewById<Button>(R.id.btnLogout).setOnClickListener {  doLogout()  }  }  private fun doLogout() {  editor.apply {  putString(EMAIL, "")  apply()  startActivity(Intent(this@DashboardActivity, MainActivity::class.java))  }  }  private fun doInitializeSharedPref() {  sharedPreferences = getSharedPreferences(MainActivity.LOGIN\_FILE, MODE\_PRIVATE)  editor = sharedPreferences.edit()  }  } | </end> |
| <hitle> | SharedPreferenceSecurityEncrypt Login | <chare> | 1 | <pext> | 01-24/SharedPreferenceSecurityEncrypt  -----------Gradle-----------  //Security Encrypted Shared Preference  implementation("androidx.security:security-crypto:1.0.0")  // For Identity Credential APIs  implementation("androidx.security:security-identity-credential:1.0.0-alpha03")  // For App Authentication APIs  implementation("androidx.security:security-app-authenticator:1.0.0-alpha02")  // For App Authentication API testing  androidTestImplementation("androidx.security:security-app-authenticator:1.0.0-alpha01")  -----------Code---------------  class SplashActivity : AppCompatActivity() {  private lateinit var encryptedSharedPreferences: SharedPreferences  private lateinit var editor: SharedPreferences.Editor  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_splash)  initEncryptedSharedPref()  checkWhereToGo()  }  private fun initEncryptedSharedPref() {  val keyGenerateParameters = MasterKeys.AES256\_GCM\_SPEC  val mainKeyAlias = MasterKeys.getOrCreate(keyGenerateParameters)  encryptedSharedPreferences = EncryptedSharedPreferences.create(  MainActivity.ENCRYPTED\_LOGIN\_FILE,  mainKeyAlias,  this,  EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256\_SIV,  EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256\_GCM  )  editor = encryptedSharedPreferences.edit()  }  private fun checkWhereToGo() {  if (encryptedSharedPreferences.getString(EMAIL, "") == "") {  startActivity(Intent(this, MainActivity::class.java ))  } else {  startActivity(Intent(this, DashboardActivity::class.java))  }  }  }  class MainActivity : AppCompatActivity() {  private lateinit var encryptedSharedPreferences: SharedPreferences  private lateinit var editor: SharedPreferences.Editor  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  initEncryptedSharedPref()  setUpViews()  }  private fun setUpViews() {  val editEmail = findViewById<EditText>(R.id.editEmail)  val editPassword = findViewById<EditText>(R.id.editPassword)  val btnLogin = findViewById<Button>(R.id.btnLogin)  btnLogin.setOnClickListener {  val email = editEmail.text.toString()  val password = editPassword.text.toString()  doLogin(email, password)  editEmail.text?.clear()  editPassword.text?.clear()  }  }  private fun doLogin(email: String, password: String) {  editor.apply {  putString(EMAIL, email)  putString(PASSWORD, password)  if (commit()) {  startActivity(Intent(this@MainActivity, DashboardActivity::class.java))  }  }  }  private fun initEncryptedSharedPref() {  val keyGenerateParameters = MasterKeys.AES256\_GCM\_SPEC  val mainKeyAlias = MasterKeys.getOrCreate(keyGenerateParameters)  encryptedSharedPreferences = EncryptedSharedPreferences.create(  ENCRYPTED\_LOGIN\_FILE,  mainKeyAlias,  this,  EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256\_SIV,  EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256\_GCM  )  editor = encryptedSharedPreferences.edit()  }  companion object {  const val ENCRYPTED\_LOGIN\_FILE = "encryptedLoginFile"  const val EMAIL = "email"  const val PASSWORD = "password"  }  }  class DashboardActivity : AppCompatActivity() {  private lateinit var encryptedSharedPreferences: SharedPreferences  private lateinit var editor: SharedPreferences.Editor  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_dashboard)  initEncryptedSharedPref()  findViewById<Button>(R.id.btnLogout).setOnClickListener {  doLogout()  }  }  private fun doLogout() {  editor.apply {  putString(EMAIL, "")  apply()  startActivity(Intent(this@DashboardActivity, MainActivity::class.java))  }  }  private fun initEncryptedSharedPref() {  val keyGenerateParameters = MasterKeys.AES256\_GCM\_SPEC  val mainKeyAlias = MasterKeys.getOrCreate(keyGenerateParameters)  encryptedSharedPreferences = EncryptedSharedPreferences.create(  MainActivity.ENCRYPTED\_LOGIN\_FILE,  mainKeyAlias,  this,  EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256\_SIV,  EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256\_GCM  )  editor = encryptedSharedPreferences.edit()  }  } | </end> |
| <hitle> | CreateFragmentDemo | <chare> | 1 | <pext> | 01-25/CreateFragmentDemo  ------------by XML-----------------  <fragment  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:name="com.example.FirstFragment"  />  -------------by programmatically at runtime----------  findViewById<Button>(R.id.btn\_add).setOnClickListener {  var fragmentManager = supportFragmentManager  var fragmentTransaction = fragmentManager.beginTransaction()  fragmentTransaction.add(R.id.first\_fragment, FirstFragment()).commit()  }  <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="20dp"  tools:context=".MainActivity">  <Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/btn\_add"  android:text="Add Fragment"/>  <FrameLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:id="@+id/first\_fragment"  />  </LinearLayout> | </end> |
| <hitle> | IntentDemo | <chare> | 1 | <pext> | 01-25/IntentDemo  ---------------startActivity pass data Intent----------  private fun openActivity() {  val intent = Intent(this@MainActivity, SecondScreen::class.java)  intent.putExtra("key1", "How")  intent.putExtra("key2", "to")  intent.putExtra("key3", "pass")  intent.putExtra("key4", "data")  startActivity(intent)  }  class SecondScreen : AppCompatActivity() {  private lateinit var binding: ActivitySecondScreenBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivitySecondScreenBinding.inflate(layoutInflater)  setContentView(binding.root)  val pass1 = intent.getStringExtra("key1")  binding.pass1.text = pass1  val pass2 = intent.getStringExtra("key2")  val pass3 = intent.getStringExtra("key3")  val pass4 = intent.getStringExtra("key4")  }  }  ---------------open browser Intent----------  private fun openBrowser() {  val intent = Intent(Intent.ACTION\_VIEW)  intent.data = Uri.parse("https://google.com")  startActivity(intent)  }  ---------------open camera Intent----------  private fun openCamera() {  val intent = Intent(MediaStore.ACTION\_IMAGE\_CAPTURE)  startActivity(intent)  }  ---------------open image/media gallery Intent----------  private fun openGallery() {  val intent = Intent(Intent.ACTION\_VIEW)  intent.data = Uri.parse("content://media/external/images/media/")  startActivity(intent)  }  ---------------open/check contacts Intent----------  private fun checkContact() {  val intent = Intent(Intent.ACTION\_PICK)  intent.type = ContactsContract.Contacts.CONTENT\_TYPE  startActivity(intent)  }  ---------------open/check callLogs Intent----------  private fun checkCallLogs() {  val intent = Intent(Intent.ACTION\_VIEW)  intent.type = CallLog.Calls.CONTENT\_TYPE  startActivity(intent)  }  ---------------open Alarm Intent----------  private fun openAlam() {  val intent = Intent(AlarmClock.ACTION\_SHOW\_ALARMS)  startActivity(intent)  }  ---------------open GoogleMap Intent----------  private fun openGoogleMap() {  val latitude = 37.7749  val longitude = -122.4192  val uri = String.format(Locale.ENGLISH, "geo:%f,%f", latitude, longitude)  val intent = Intent(Intent.ACTION\_VIEW, Uri.parse(uri))  startActivity(intent)  }  ---------------open makeCall Intent----------  private fun makeCall() {  val intent = Intent(Intent.ACTION\_DIAL)  intent.data = Uri.fromParts("tel","+1 331 551 8086", null)  startActivity(intent)  } | </end> |
| <hitle> | IntentDemo Serialize/Parcelize | <chare> | 1 | <pext> | 01-25/IntentDemo Serialize/Parcelize  ----------------Gradle-------------  plugins {  id 'kotlin-parcelize'  }  ------------Data class----------------  @Parcelize  data class Student(val name: String, val age: Int, val salary: Double): Parcelable  data class Employee(val name: String, val age: Int, val salary: Double): Serializable  ---------------startActivity Intent---------------  private fun openActivity() {  val intent = Intent(this@MainActivity, SecondScreen::class.java)  val employee = Employee("Thomas", 33, 100.0)  intent.putExtra("data\_employee", employee)  val student = Student("Thomas", 33, 100.0)  intent.putExtra("data\_student", student)  startActivity(intent)  }  ----------------receive Intent Data----------------  val employeeData = intent.extras?.get("data\_employee") as Employee?  employeeData?.let {  val data = "${employeeData.name}, ${employeeData.age}, ${employeeData.salary}"  binding.pass1.text = data  }  val student = intent.getParcelableExtra<Student>("data\_student")  student?.let {  val data = "${student.name}, ${student.age}, ${student.salary}"  binding.pass1.text = data  } | </end> |
| <hitle> | DialogBoxesDemo | <chare> | 1 | <pext> | 01-26/DialogBoxesDemo  ----------------------AlertDialog Confirmation--------------  private fun openConirmationDialg() {  val builder = AlertDialog.Builder(this)  .setTitle("Delete")  .setMessage("Would you like to delete?")  .setPositiveButton("Confirm") {\_,\_ ->  showToast("Deleted")  }  .setNegativeButton("Cancel") {\_,\_ ->  showToast("Canceled")  }  val alertDialog = builder.create()  alertDialog.setCancelable(false)  alertDialog.show()  }  ---------------------- SelectSingleChoice AlertDialog--------------  private fun selectSingleChoiceUsingDialog() {  val language = arrayOf(  "Language1",  "Language2",  "Language3",  "Language4",  "Language5",  "Language6",  "Language7",  "Language8",  )  val builder = AlertDialog.Builder(this).apply {  setIcon(R.drawable.ic\_launcher\_background)  setTitle("Select your Language")  setSingleChoiceItems(language, 0) { dialog, position ->  val selectedLanguage = language[position]  showToast("You had selected ${language[position]}")  dialog.dismiss()  }  }  val dialog = builder.create()  dialog.setCancelable(false)  dialog.show()  }  -----------------------SelectMutipleChoice AlertDialog------------------------------  private fun selectMutipleChoiceUsingDialog() {  val skills = arrayOf(  "Skill1",  "Skill2",  "Skill3",  "Skill4",  "Skill5",  "Skill6",  "Skill7",  )  val checkedItems = BooleanArray(skills.size) { false }  checkedItems[1] = true  checkedItems[3] = true  val builder = AlertDialog.Builder(this).apply {  setIcon(R.drawable.ic\_launcher\_background)  setTitle("Select your Language")  setMultiChoiceItems(skills, checkedItems) { \_, position, checked ->  checkedItems[position] = true  }  setPositiveButton("Done") { dialog, \_ ->  val output = StringBuilder( "Selected skills are \n")  for (i in checkedItems.indices) {  if(checkedItems[i]) {  output.append("${skills[i]}\n")  }  }  showToast(output.toString())  dialog.dismiss()  }  }  val dialog = builder.create()  dialog.setCancelable(false)  dialog.show()  }  --------------------AlertDialog by XML-------------------------------  private fun openLoginDialog() {  val dialogLoginBinding: LoginFormBinding = LoginFormBinding.inflate(layoutInflater)  val builder = AlertDialog.Builder(this).apply {  setView(dialogLoginBinding.root)  setCancelable(false)  }  val dialog = builder.create()  dialog.window?.setGravity(Gravity.CENTER)  dialogLoginBinding.apply {  dlgBtnCancel.setOnClickListener { dialog.dismiss() }  dlgBtnLogin.setOnClickListener {  val email = editName.text.toString()  val password = editPass.text.toString()  if(email.isNotEmpty() && password.isNotEmpty()) {  dialog.dismiss()  showToast("Logged in")  } else {  showToast("Please fill the login fields")  }  }  }  dialog.show()  }  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  android:layout\_margin="15dp"  android:elevation="10dp"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@color/purple\_200"  >  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editName"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Enter your name"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_margin="10dp"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editPass"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Enter your password"  app:layout\_constraintTop\_toBottomOf="@+id/editName"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_margin="10dp"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/dlgBtnLogin"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/dlgBtnCancel"  app:layout\_constraintTop\_toBottomOf="@+id/editPass"  android:text="Login"  android:textSize="20dp"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/dlgBtnCancel"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@+id/dlgBtnLogin"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/editPass"  android:text="Cancel"  android:textSize="20dp"  />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  ------------------DialogFragment-----------------------  binding.btnSignUp.setOnClickListener {  SignUpDialog().show(supportFragmentManager, "SignUpDialog")  }  class SignUpDialog : DialogFragment() {  private lateinit var binding: FragmentSignUpDialogBinding  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  binding = FragmentSignUpDialogBinding.inflate(layoutInflater)  return binding.root  }  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  binding.dlgBtnSignup.setOnClickListener {  register()  }  }  private fun register() {  val email = binding.editName.text.toString()  val password = binding.editPass.text.toString()  val rePassword = binding.editConfirmPass.text.toString()  if(email.isNotEmpty() && password.length> 5 && rePassword == password) {  dialog?.dismiss()  Toast.makeText(requireContext(), " Registerd successfully", Toast.LENGTH\_SHORT).show()  } else {  dialog?.dismiss()  Toast.makeText(requireContext(), "Please enter valid input", Toast.LENGTH\_SHORT).show()  }  binding.dlgBtnCancel.setOnClickListener {  dialog?.dismiss()  Toast.makeText(requireContext(), "Working?", Toast.LENGTH\_SHORT).show()  }  }  override fun onResume() {  super.onResume()  val params = dialog?.window?.attributes  params?.height = WindowManager.LayoutParams.MATCH\_PARENT  params?.width = WindowManager.LayoutParams.MATCH\_PARENT  dialog?.window?.attributes = params  }  override fun onStop() {  super.onStop()  Log.i("tag", "Hey I am dialog and going to die")  }  }  <?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\_gravity="center"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@color/white"  android:padding="15dp"  >  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editName"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Enter your name"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_margin="10dp"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editPass"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Enter your password"  app:layout\_constraintTop\_toBottomOf="@+id/editName"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_margin="10dp"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editConfirmPass"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Confirm your password"  app:layout\_constraintTop\_toBottomOf="@+id/editPass"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_margin="10dp"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/dlgBtnSignup"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/dlgBtnCancel"  app:layout\_constraintTop\_toBottomOf="@+id/editConfirmPass"  android:text="Signup"  android:textSize="20dp"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/dlgBtnCancel"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@id/dlgBtnSignup"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/editConfirmPass"  android:text="Cancel"  android:textSize="20dp"  />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.constraintlayout.widget.ConstraintLayout>  ------------------BottomSheetDialog--------------  binding.btnBottomSheet.setOnClickListener {  val bottomSheetDialog = BottomSheetDialog(this)  val bottomviewBinding = BottomSheetUiBinding.inflate(layoutInflater)  bottomSheetDialog.setContentView(bottomviewBinding.root)  bottomSheetDialog.show()  }  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  xmlns:app="http://schemas.android.com/apk/res-auto">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  android:orientation="vertical"  >  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  android:weightSum="3"  >  <TextView  android:id="@+id/btnBottom1"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:elevation="0dp"  android:text="Gmail"  android:gravity="center"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_attach\_email\_24"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom2"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Hangout"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_baby\_changing\_station\_24"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom3"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Google+"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_sports\_golf\_24"  />  </LinearLayout>  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  android:weightSum="3"  >  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom4"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Mail"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_mail\_24"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom5"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Message"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_message\_24"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom6"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Copy"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_content\_copy\_24"  />  </LinearLayout>  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  android:weightSum="3"  >  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom7"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Facebook"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_tag\_faces\_24"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBottom8"  android:layout\_weight="1"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="Twitter"  android:background="@color/white"  android:textStyle="bold"  android:drawableTop="@drawable/ic\_baseline\_wb\_twilight\_24"  />  </LinearLayout>  </LinearLayout>  </androidx.constraintlayout.widget.ConstraintLayout>  ------------------------DatePickerDialog-----------------  binding.btnSelectBirthday.setOnClickListener {  val calendar = Calendar.getInstance()  val year = calendar.get(Calendar.YEAR)  val month = calendar.get(Calendar.MONTH)  val day = calendar.get(Calendar.DAY\_OF\_MONTH)  val datePicker = DatePickerDialog(  this,  { \_: DatePicker, selectedYear: Int, selectedMonth: Int, selectedDay: Int ->  binding.btnSelectBirthday.text = "Date: ${selectedMonth + 1} - $selectedDay - $selectedYear"  }, year, month, day  )  datePicker.show()  }  ------------------TimePickerDialog---------------------  binding.btnSelectPartyTime.setOnClickListener {  val calendar = Calendar.getInstance()  val hour = calendar.get(Calendar.HOUR)  val minute = calendar.get(Calendar.MINUTE)  val timePicker = TimePickerDialog(  this,  { \_, selectedHour, selectedMin ->  binding.btnSelectPartyTime.text = "Time: $selectedHour : $selectedMin"  }, hour, minute, true  )  timePicker.show()  }  --------------------ProgressDialog Download-------------------------  binding.btnDownload.setOnClickListener {  val progressDialog = ProgressDialog(this)  progressDialog.apply {  setTitle("How long it will take?")  setMessage("It will take 10 minutes!")  show()  }  } | </end> |
| <hitle> | StartActivityDemo | <chare> | 1 | <pext> | 01-30/StartActivityForResultDemo  -----------MainActivity--------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  binding.btn100.setOnClickListener {  makeRecharge(100)  }  binding.btn250.setOnClickListener {  makeRecharge(250)  }  binding.btn500.setOnClickListener {  makeRecharge(500)  }  }  private fun makeRecharge(rechargeValue: Int) {  val intent = Intent(this, SecondActivity::class.java)  intent.putExtra(RECHARGE\_AMOUNT, rechargeValue)  startActivityForResult(intent, REQUEST\_CODE)  }  override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  super.onActivityResult(requestCode, resultCode, data)  if(requestCode == REQUEST\_CODE && resultCode == Activity.RESULT\_OK) {  val result = data?.getBooleanExtra(RECHARGE\_STATUS, false)  if(result == true) {  binding.imgStatus.setImageResource(R.drawable.ic\_baseline\_done\_outline\_24)  binding.imgStatus.visibility = View.VISIBLE  } else {  binding.imgStatus.setImageResource(R.drawable.ic\_baseline\_close\_24)  binding.imgStatus.visibility = View.VISIBLE  }  }  }  companion object {  const val RECHARGE\_AMOUNT = "rechargeAmount"  const val REQUEST\_CODE = 100  }  }  -------------SecondActivity-----------  class SecondActivity : AppCompatActivity() {  private lateinit var binding: ActivitySecondBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivitySecondBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  val valueAmount = intent.extras?.get(RECHARGE\_AMOUNT)?.toString()  binding.apply {  txtRechargeValue.text = "Recharge for $valueAmount"  btnSuccess.setOnClickListener { sendingStatus(true) }  btnFailure.setOnClickListener { sendingStatus(false) }  }  }  private fun sendingStatus(status: Boolean) {  val intent = Intent()  intent.putExtra(RECHARGE\_STATUS, status)  setResult(Activity.RESULT\_OK, intent)  finish()  }  companion object {  const val RECHARGE\_STATUS = "rechargeStatus"  }  }  <?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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Recharge App"  android:textSize="30sp"  android:textStyle="bold"  android:textColor="@color/black"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtRecharge"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="40dp"  android:text="Select your recharge pack"  android:textSize="25sp"  android:textColor="@color/black"  android:textAlignment="center"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <Button  android:id="@+id/btn100"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="100"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/btn250"  />  <Button  android:id="@+id/btn250"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="250"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/btn100"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/btn500"  />  <Button  android:id="@+id/btn500"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="250"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/btn250"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgStatus"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="100dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/txtRecharge"  />  </androidx.constraintlayout.widget.ConstraintLayout>  ---------------------activity\_second.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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".SecondActivity">  <TextView  android:id="@+id/txtRechargeValue"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Recharge Value"  android:textSize="30sp"  android:textAlignment="center"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSuccess"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Success"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@+id/btnFailure"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnFailure"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Failure"  app:layout\_constraintTop\_toBottomOf="@+id/btnSuccess"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | ListViewDemo | <chare> | 1 | <pext> | 01-30/ListViewDemo  val fruits = arrayOf("Mango", "Apple", "Lemon", "Grapes", "Orange")  val fruitAdapter = ArrayAdapter(this, android.R.layout.simple\_list\_item\_1, fruits)  binding.listview.adapter = fruitAdapter | </end> |
| <hitle> | ListViewSpinnerDemo | <chare> | 1 | <pext> | 01-30/ListViewSpinnerDemo  -------------MainActivity-------------  val fruits = arrayOf("Mango", "Grapes", "Lemon", "Orange", "Apple", "Banana")  val fruitAdapter = ArrayAdapter(this, android.R.layout.simple\_list\_item\_1, fruits)  binding.listView.adapter = fruitAdapter  binding.spinner.adapter = fruitAdapter  ------------activity\_main.xml---------  <Spinner  android:id="@+id/spinner"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  />  <ListView  android:id="@+id/listView"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  app:layout\_constraintTop\_toBottomOf="@id/spinner"  app:layout\_constraintBottom\_toBottomOf="parent"  />  ----------UserAdapter-----------  class UserAdapter(private val userList: ArrayList<User>): BaseAdapter() {  override fun getCount(): Int = userList.size  override fun getItem(position: Int) = position  override fun getItemId(position: Int) = position.toLong()  override fun getView(position: Int, convertView: View?, parent: ViewGroup?): View {  val binding = UserItemBinding.inflate(LayoutInflater.from(parent?.context), parent, false)  binding.apply {  txtUserName.text = userList[position].name  txtUserInfo.text = userList[position].userInfo  txtUserAge.text = userList[position].userAge  imageUser.setImageResource(userList[position].image)  }  return binding.root  }  }  ----------------UserActivity------------  val userList = ArrayList<User>()  userList.apply {  add(User(R.drawable.ic\_launcher\_background, "Name1", "UserInfo1", "age1"))  add(User(R.drawable.ic\_launcher\_background, "Name2", "UserInfo2", "age2"))  add(User(R.drawable.ic\_launcher\_background, "Name3", "UserInfo3", "age3"))  …  }  binding.listView.adapter = UserAdapter(userList)  binding.userSpinner.adapter = UserAdapter(userList)  -----------activity\_user.xml------------  <ListView  android:id="@+id/listView"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  tools:listitem="@layout/user\_item"  android:layout\_margin="5dp"  android:dividerHeight="5dp"  app:layout\_constraintTop\_toBottomOf="@id/userSpinner"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <Spinner  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  app:layout\_constraintTop\_toTopOf="parent"  android:id="@+id/userSpinner"  /> | </end> |
| <hitle> | RecyclerViewDemo | <chare> | 1 | <pext> | 01-31/RecyclerViewDemo  -----------MainActivity----------------  private fun initViews() {  binding.recyclerView.layoutManager = LinearLayoutManager(this)  binding.recyclerView.adapter = NotesAdapter(getNotes())  binding.btnTest1.setOnClickListener {  val intent = Intent(this@MainActivity, SecondActivity::class.java)  startActivity(intent)  }  }  private fun getNotes(): List<Note> {  val listOfNotes = ArrayList<Note>()  listOfNotes.apply {  add(Note(1, "Name1", "this is the description", true, true, R.drawable.british))  add(Note(2, "Name2", "this is the description", false, true, R.drawable.us))  add(Note(3, "Name3", "this is the description", true, true, R.drawable.spain))  …  }  return listOfNotes  }  ---------------------NoteAdapter--------------  class NotesAdapter(private val notes: List<Note>): RecyclerView.Adapter<NotesAdapter.NotesViewHolder>() {  private lateinit var binding: NoteItemBinding  override fun getItemCount() = notes.size  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NotesViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NoteItemBinding.inflate(layoutInflater, parent, false)  return NotesViewHolder(binding.root)  }  override fun onBindViewHolder(holder: NotesViewHolder, position: Int) {  holder.bind(notes[position])  }  inner class NotesViewHolder(val view: View): RecyclerView.ViewHolder(view) {  fun bind(note: Note) {  note.apply {  binding.apply {  countryFlag?.let {  imgFlag.setImageResource(it)  }  if(isLocked) {  imgLock.setImageResource(R.drawable.ic\_baseline\_lock\_24)  }  else {  imgLock.setImageResource(R.drawable.ic\_baseline\_lock\_open\_24)  }  if(isFavorite) {  imgStar.setImageResource(R.drawable.ic\_baseline\_star\_24)  }  else {  imgStar.setImageResource(R.drawable.ic\_baseline\_star\_border\_24)  }  txtTitle.text = title  txtDesc.text = description  }  }  }  }  }  ---------------Note-------------  data class Note(  val id: Int,  val title: String,  val description: String,  val isLocked: Boolean,  val isFavorite: Boolean,  @DrawableRes val countryFlag: Int? = null  )  ------------note\_item.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="#009688"  android:layout\_margin="5dp"  xmlns:app="http://schemas.android.com/apk/res-auto">  <ImageView  android:id="@+id/imgFlag"  android:layout\_width="90dp"  android:layout\_height="0dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <TextView  android:id="@+id/txtTitle"  android:textSize="40sp"  android:text="Title"  android:layout\_width="0dp"  android:layout\_margin="10dp"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgFlag"  app:layout\_constraintEnd\_toStartOf="@+id/imgLock"  app:layout\_constraintBottom\_toTopOf="@+id/txtDesc"  />  <TextView  android:id="@+id/txtDesc"  android:textSize="20sp"  android:text="Description"  android:layout\_margin="10dp"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@id/imgFlag"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  app:layout\_constraintEnd\_toStartOf="@+id/imgLock"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <ImageView  android:id="@+id/imgLock"  android:layout\_margin="10dp"  android:layout\_width="30dp"  android:layout\_height="30dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@id/txtTitle"  app:layout\_constraintBottom\_toBottomOf="@+id/imgStar"  />  <ImageView  android:id="@+id/imgStar"  android:layout\_margin="10dp"  android:layout\_width="30dp"  android:layout\_height="30dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/imgLock"  app:layout\_constraintStart\_toEndOf="@id/txtTitle"  app:layout\_constraintBottom\_toBottomOf="parent"  android:src="@drawable/ic\_launcher\_background"  />  </androidx.constraintlayout.widget.ConstraintLayout>  --------------SecondActivity------------------  private fun initViews() {  binding.btnLinear.setOnClickListener {  binding.recyclerView.layoutManager = LinearLayoutManager(this)  binding.recyclerView.adapter = NotesAdapter1(getNotes())  }  binding.btnGrid.setOnClickListener {  binding.recyclerView.layoutManager = GridLayoutManager(this, 2)  binding.recyclerView.adapter = NotesAdapter1(getNotes())  }  binding.btnSlide.setOnClickListener {  binding.recyclerView.layoutManager =  StaggeredGridLayoutManager(2, LinearLayoutManager.HORIZONTAL)  binding.recyclerView.adapter = NotesAdapter1(getNotes())  }  }  private fun getNotes(): List<Note> {  val listOfNotes = ArrayList<Note>()  listOfNotes.apply {  add(Note(1, "Name1", "this is the description", true, true, R.drawable.british))  add(Note(2, "Name2", "this is the description", false, true, R.drawable.us))  add(Note(3, "Name3", "this is the description", true, true, R.drawable.spain))  …  }  return listOfNotes  }  ----------------NoteAdapter1-------------  class NotesAdapter1(private val notes: List<Note>): RecyclerView.Adapter<NotesAdapter1.NotesViewHolder1>() {  private lateinit var binding: NoteItem1Binding  override fun getItemCount() = notes.size  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NotesViewHolder1 {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NoteItem1Binding.inflate(layoutInflater, parent, false)  return NotesViewHolder1(binding.root)  }  override fun onBindViewHolder(holder: NotesViewHolder1, position: Int) {  holder.bind(notes[position])  }  inner class NotesViewHolder1(val view: View): RecyclerView.ViewHolder(view) {  fun bind(note: Note) {  note.apply {  binding.apply {  countryFlag?.let {  imgFlag.setImageResource(it)  }  txtTitle.text = title  }  }  }  }  }  ---------------note\_item1.xml-----------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:background="#EDBFC8"  android:layout\_margin="5dp"  xmlns:app="http://schemas.android.com/apk/res-auto">  <ImageView  android:id="@+id/imgFlag"  android:layout\_width="90dp"  android:layout\_height="60dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@id/txtTitle"  />  <TextView  android:id="@+id/txtTitle"  android:textSize="20sp"  android:text="Title"  android:layout\_width="0dp"  android:layout\_margin="10dp"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@id/imgFlag"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | RecyclerViewAdvanced | <chare> | 1 | <pext> | 01-13/ RecyclerViewAdvanced  -----------MainActivity----------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var gmailList: ArrayList<Email>  private lateinit var gmailAdapter: GmailAdapter  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  getGmailList()  gmailAdapter = GmailAdapter(this, gmailList)  binding.gmailRecyclerview.layoutManager = LinearLayoutManager(this)  binding.gmailRecyclerview.adapter = gmailAdapter  ItemTouchHelper(object: ItemTouchHelper.SimpleCallback(0, ItemTouchHelper.LEFT) {  override fun onSwiped(viewHolder: RecyclerView.ViewHolder, direction: Int) {  val position = viewHolder.adapterPosition  gmailList.removeAt(position)  gmailAdapter.notifyItemRemoved(position)  }  override fun onMove(  recyclerView: RecyclerView,  viewHolder: RecyclerView.ViewHolder,  target: RecyclerView.ViewHolder  ): Boolean {  return false  }  }).attachToRecyclerView(binding.gmailRecyclerview)  }  private fun getGmailList() {  gmailList = ArrayList<Email>()  gmailList.apply {  add(Email("sample1@gmail.com", "Interview sith TCS", "This is an important email", true, "10:00AM", R.drawable.actor1))  add(Email("sample2@gmail.com", "Interview sith TCS", "This is an important email", true, "10:00AM", R.drawable.actor1))  add(Email("sample3@gmail.com", "Interview sith TCS", "This is an important email", true, "10:00AM", R.drawable.actor1))  … }  }  }  -------------GmailAdapter--------------  class GmailAdapter(private val context: Context, private val gmailItems: List<Email>): RecyclerView.Adapter<GmailAdapter.EmailViewHolder>() {  private lateinit var binding: EmailItemBinding  override fun getItemCount() = gmailItems.size  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): EmailViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = EmailItemBinding.inflate(layoutInflater, parent, false)  return EmailViewHolder(binding.root)  }  override fun onBindViewHolder(holder: EmailViewHolder, position: Int) {  val email = gmailItems[position]  holder.bind(gmailItems[position])  holder.itemView.setOnClickListener {  val intent = Intent(context, GmailDetailActivity::class.java)  intent.putExtra(EMAIL\_DATA, email)  context.startActivity(intent)  }  }  inner class EmailViewHolder(view: View) : RecyclerView.ViewHolder(view) {  fun bind(email: Email) {  email.apply {  binding.apply {  txtSender.text = sender  txtTime.text = title  txtBody.text = body  txtTime.text = time  senderImage.setImageResource(senderPic)  }  }  }  }  companion object {  const val EMAIL\_DATA = "emailData"  }  }  --------------Email--------------  data class Email (  val sender: String,  val title: String,  val body: String,  val isFavorite: Boolean,  val time: String,  @DrawableRes val senderPic: Int  ): Serializable  ------------email\_item.xml-------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  xmlns:app="http://schemas.android.com/apk/res-auto"  android:padding="5dp"  android:elevation="15dp"  android:layout\_margin="10dp"  app:cardCornerRadius="10dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  >  <androidx.cardview.widget.CardView  android:id="@+id/cardImage"  android:layout\_width="60dp"  android:layout\_height="60dp"  android:layout\_margin="10dp"  app:cardCornerRadius="30dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  >  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/senderImage"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:elevation="20dp"  android:src="@drawable/actor1"  android:scaleType="centerCrop"  />  </androidx.cardview.widget.CardView>  <TextView  android:id="@+id/txtSender"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:padding="5dp"  android:textSize="18sp"  android:textColor="@color/black"  android:textStyle="bold"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toTopOf="parent"  android:text="Sender name"  />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="5dp"  android:padding="5dp"  android:textSize="13sp"  android:maxLines="1"  android:textColor="@color/black"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toBottomOf="@id/txtSender"  app:layout\_constraintEnd\_toStartOf="@+id/glHori80"  android:text="Title"  />  <TextView  android:id="@+id/txtBody"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="5dp"  android:padding="5dp"  android:textSize="13sp"  android:textColor="@color/black"  android:gravity="start"  android:maxLines="2"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toBottomOf="@id/txtTitle"  android:text="Email body"  />  <TextView  android:id="@+id/txtTime"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="10:00AM"  android:textColor="@color/black"  android:textSize="14sp"  app:layout\_constraintBottom\_toTopOf="@+id/toggleButton"  app:layout\_constraintStart\_toEndOf="@+id/glHori80"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori80"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintGuide\_percent=".8"  android:orientation="vertical"  />  <ToggleButton  android:id="@+id/toggleButton"  android:layout\_width="40dp"  android:layout\_height="40dp"  android:background="@drawable/favorite\_toggle\_selector\_button\_bg"  android:textOff=""  android:textOn=""  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/glHori80"  app:layout\_constraintTop\_toBottomOf="@id/txtTime" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  ----------------activity\_main.xml------------  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/gmailRecyclerview"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:layout\_margin="5dp"  />  --------------GmailDetailActivity------------  class GmailDetailActivity : AppCompatActivity() {  private lateinit var binding: ActivityGmailDetailBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityGmailDetailBinding.inflate(layoutInflater)  setContentView(binding.root)  val email = intent.extras?.get(EMAIL\_DATA) as Email  email.apply {  binding.apply {  txtSender.text = sender  txtTime.text = time  txtBody.text = body  txtTitle.text = title  senderImage.setImageResource(senderPic)  }  }  }  }  ----------------activity\_gmail\_detail.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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".GmailDetailActivity">  <androidx.cardview.widget.CardView  android:id="@+id/cardImage"  android:layout\_width="60dp"  android:layout\_height="60dp"  android:layout\_margin="10dp"  app:cardCornerRadius="30dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  >  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/senderImage"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:elevation="20dp"  android:src="@drawable/actor1"  android:scaleType="centerCrop"  />  </androidx.cardview.widget.CardView>  <TextView  android:id="@+id/txtSender"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:padding="5dp"  android:textSize="18sp"  android:textColor="@color/black"  android:textStyle="bold"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toTopOf="parent"  android:text="Sender name"  />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="5dp"  android:padding="5dp"  android:textSize="13sp"  android:textColor="@color/black"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toBottomOf="@id/txtSender"  app:layout\_constraintEnd\_toStartOf="@+id/glHori80"  android:text="Title"  />  <TextView  android:id="@+id/txtBody"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="5dp"  android:padding="5dp"  android:textSize="13sp"  android:textColor="@color/black"  android:gravity="start"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toBottomOf="@id/txtTitle"  android:text="Email body"  />  <TextView  android:id="@+id/txtTime"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:text="10:00AM"  android:textColor="@color/black"  android:textSize="14sp"  app:layout\_constraintBottom\_toTopOf="@+id/toggleButton"  app:layout\_constraintStart\_toEndOf="@+id/glHori80"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.constraintlayout.widget.Guideline  android:id="@+id/glHori80"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintGuide\_percent=".8"  android:orientation="vertical"  />  <ToggleButton  android:id="@+id/toggleButton"  android:layout\_width="40dp"  android:layout\_height="40dp"  android:background="@drawable/favorite\_toggle\_selector\_button\_bg"  android:textOff=""  android:textOn=""  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/glHori80"  app:layout\_constraintTop\_toBottomOf="@id/txtTime" />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | WhatAppStart | <chare> | 1 | <pext> | 02-01/WhatAppStart  -------------------MainActivity--------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var chatAdapter: ChatAdapter  private lateinit var chats: ArrayList<Chat>  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  prepareDataSet()  makeChattingButtonsActive()  }  private fun makeChattingButtonsActive() {  binding.btnReceiver.setOnClickListener {  chats.add(Chat(1, binding.txtMessage.text.toString()))  chatAdapter.notifyItemInserted(chats.size)  binding.txtMessage.text?.clear()  }  binding.btnSender.setOnClickListener {  chats.add(Chat(2, binding.txtMessage.text.toString()))  chatAdapter.notifyItemInserted(chats.size)  binding.txtMessage.text?.clear()  }  }  private fun prepareDataSet() {  chats = ArrayList()  chats.apply {  add(Chat(1, "Hey"))  add(Chat(2, "Hi"))  }  chatAdapter = ChatAdapter(chats)  setUpRecyclerView()  }  private fun setUpRecyclerView() {  binding.chatRecyclerView.layoutManager = LinearLayoutManager(this)  binding.chatRecyclerView.adapter = chatAdapter  }  }  ----------ChatAdapter---------------  class ChatAdapter(private val chats: ArrayList<Chat>): RecyclerView.Adapter<RecyclerView.ViewHolder>() {  private lateinit var bindingSender: SenderBinding  private lateinit var bindingReceiver: ReceiverBinding  override fun getItemCount() = chats.size  override fun getItemViewType(position: Int): Int {  return chats[position].viewType  }  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): RecyclerView.ViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  return if (viewType == SENDER\_VIEW) {  bindingSender = SenderBinding.inflate(layoutInflater, parent, false)  SenderViewHolder(bindingSender.root)  } else {  bindingReceiver = ReceiverBinding.inflate(layoutInflater, parent, false)  ReceiverViewHolder(bindingReceiver.root)  }  }  override fun onBindViewHolder(holder: RecyclerView.ViewHolder, position: Int) {  if(chats[position].viewType == SENDER\_VIEW) {  (holder as SenderViewHolder).bind(position)  } else {  (holder as ReceiverViewHolder).bind(position)  }  }  inner class ReceiverViewHolder(view: View): RecyclerView.ViewHolder(view) {  private val receiver = bindingReceiver.txtReceiver  fun bind(position: Int) {  receiver.text = chats[position].text  }  }  inner class SenderViewHolder(view: View): RecyclerView.ViewHolder(view) {  private val sender = bindingSender.txtSender  fun bind(position: Int) {  sender.text = chats[position].text  }  }  companion object {  const val SENDER\_VIEW = 1  }  }  --------------Chat----------------  data class Chat(val viewType: Int, val text: String)  ------------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@drawable/whatsapp\_bg"  tools:context=".MainActivity">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/chatRecyclerView"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  app:layout\_constraintTop\_toTopOf="parent"  tools:listitem="@layout/receiver"  app:layout\_constraintBottom\_toTopOf="@+id/btnReceiver"  />  <androidx.appcompat.widget.AppCompatImageButton  android:id="@+id/btnReceiver"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:padding="10dp"  android:layout\_margin="5dp"  android:clickable="true"  android:focusable="true"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:background="@drawable/receiver\_button\_bg"  android:src="@drawable/ic\_baseline\_send\_24"  />  <androidx.appcompat.widget.AppCompatImageButton  android:id="@+id/btnSender"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:padding="10dp"  android:layout\_margin="5dp"  android:clickable="true"  android:focusable="true"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  android:background="@drawable/sender\_button\_bg"  android:src="@drawable/ic\_baseline\_send\_24"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/txtMessage"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:padding="10dp"  android:hint="Enter your message"  app:layout\_constraintStart\_toEndOf="@+id/btnSender"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/btnReceiver"  android:background="@drawable/edittext\_shape"  />  </androidx.constraintlayout.widget.ConstraintLayout>  -------------------sender.xml---------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  xmlns:app="http://schemas.android.com/apk/res-auto">  <androidx.cardview.widget.CardView  android:id="@+id/cardImage"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  app:cardCornerRadius="30dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  >  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/senderImage"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:scaleType="centerCrop"  android:src="@drawable/actor2"  />  </androidx.cardview.widget.CardView>  <TextView  android:id="@+id/txtSender"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="20dp"  android:paddingStart="10dp"  android:paddingEnd="5dp"  android:paddingBottom="5dp"  android:gravity="center\_vertical"  android:text="Hello"  android:background="@drawable/reciver\_shape"  android:textSize="20dp"  android:textStyle="bold"  app:layout\_constraintEnd\_toStartOf="@+id/cardImage"  app:layout\_constraintTop\_toTopOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  -----------receiver.xml----------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  xmlns:app="http://schemas.android.com/apk/res-auto">  <androidx.cardview.widget.CardView  android:id="@+id/cardImage"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  app:cardCornerRadius="30dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  >  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/senderImage"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:scaleType="centerCrop"  android:src="@drawable/actor1"  />  </androidx.cardview.widget.CardView>  <TextView  android:id="@+id/txtReceiver"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="20dp"  android:paddingStart="10dp"  android:paddingEnd="5dp"  android:paddingBottom="5dp"  android:gravity="center\_vertical"  android:text="Hello"  android:background="@drawable/sender\_shape"  android:textSize="20dp"  android:textStyle="bold"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toTopOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  -------------------sender\_shape.xml-------------  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android">  <solid android:color="#65BDF1" />  <corners  android:topRightRadius="120dp"  android:bottomRightRadius="120dp"  android:bottomLeftRadius="120dp"  />  </shape>  --------------sender\_button\_bg.xml-------------  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item android:state\_pressed="true">  <shape>  <solid android:color="#5A69CA" />  <corners  android:topRightRadius="80dp"  android:bottomRightRadius="80dp"  android:bottomLeftRadius="80dp"  />  </shape>  </item>  <item android:state\_pressed="false">  <shape>  <solid android:color="#0A2FE8" />  <corners  android:topRightRadius="80dp"  android:bottomRightRadius="80dp"  android:bottomLeftRadius="80dp"  />  </shape>  </item>  </selector>  ------------------receiver\_shape.xml----------------  <?xml version="1.0" encoding="utf-8"?>  <shape xmlns:android="http://schemas.android.com/apk/res/android">  <solid android:color="#F16359" />  <corners  android:topLeftRadius="120dp"  android:bottomRightRadius="120dp"  android:bottomLeftRadius="120dp"  />  </shape>  -----------------receiver\_button\_bg.xml--------------  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item android:state\_pressed="true">  <shape>  <solid android:color="#F374CF" />  <corners  android:topLeftRadius="80dp"  android:bottomRightRadius="80dp"  android:bottomLeftRadius="80dp"  />  </shape>  </item>  <item android:state\_pressed="false">  <shape>  <solid android:color="#6C046A" />  <corners  android:topLeftRadius="80dp"  android:bottomRightRadius="80dp"  android:bottomLeftRadius="80dp"  />  </shape>  </item>  </selector>  ----------------edittext\_shape.xml------------  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item android:state\_focused="true">  <shape android:shape="rectangle">  <corners android:radius="30dp" />  <stroke android:width="1dp"  android:color="#202020"  android:dashWidth="1dp"  android:dashGap="2dp" />  <solid android:color="#FFFFFF" />  </shape>  </item>  <item android:state\_focused="false">  <shape android:shape="rectangle">  <corners android:radius="30dp" />  <stroke android:width="1dp"  android:color="#202020"  android:dashWidth="1dp"  android:dashGap="2dp" />  <solid android:color="#D3D0D0" />  </shape>  </item>  </selector> | </end> |
| <hitle> | MenuDemo | <chare> | 1 | <pext> | 02-01/MenuDemo  --------------MainActivity--------------  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnMenu.setOnClickListener {  popUpMenuWork()  }  setUpListView()  }  private fun setUpListView() {  val contacts = arrayOf("Name1", "Name2", "Name3", "Name4", "Name5")  val arrayAdapter = ArrayAdapter(this, android.R.layout.simple\_list\_item\_1, contacts)  binding.listOfContacts.adapter = arrayAdapter  registerForContextMenu(binding.listOfContacts)  }  override fun onCreateContextMenu(  menu: ContextMenu?,  v: View?,  menuInfo: ContextMenu.ContextMenuInfo?  ) {  super.onCreateContextMenu(menu, v, menuInfo)  menuInflater.inflate(R.menu.contact\_menu, menu)  }  override fun onContextItemSelected(item: MenuItem): Boolean {  return when(item.itemId) {  R.id.call -> {  makeToast("Call")  true  }  R.id.sms -> {  makeToast("SMS")  true  }  else -> super.onContextItemSelected(item)  }  }  override fun onCreateOptionsMenu(menu: Menu?): Boolean {  menuInflater.inflate(R.menu.pop\_up\_menu, menu)  return true  }  override fun onOptionsItemSelected(item: MenuItem): Boolean {  return when(item.itemId) {  R.id.settings -> {  makeToast("Settings")  true  }  R.id.share -> {  makeToast("Share")  true  }  R.id.save -> {  makeToast("Save")  true  }  R.id.logout -> {  makeToast("Logout")  true  }  else -> super.onOptionsItemSelected(item)  }  }  private fun makeToast(message: String): Boolean {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  return true  }  private fun popUpMenuWork() {  val popUpMenu = PopupMenu(this, binding.btnMenu)  popUpMenu.menuInflater.inflate(R.menu.option\_menu, popUpMenu.menu)  popUpMenu.setOnMenuItemClickListener {  when(it.itemId) {  R.id.coffee -> makeToast("Coffee")  R.id.tea -> makeToast("Tea")  R.id.juice -> makeToast("Juice")  R.id.water -> makeToast("Water")  }  true  }  popUpMenu.show()  }  }  -----------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <Button  android:id="@+id/btnMenu"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Hello World!"  app:layout\_constraintBottom\_toTopOf="@id/listOfContacts"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <ListView  android:id="@+id/listOfContacts"  android:layout\_width="wrap\_content"  android:layout\_height="0dp"  android:layout\_margin="20dp"  android:divider="@color/black"  android:dividerHeight="2dp"  app:layout\_constraintTop\_toBottomOf="@+id/btnMenu"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  ----------------------contact\_menu.xml------------  <?xml version="1.0" encoding="utf-8"?>  <menu xmlns:android="http://schemas.android.com/apk/res/android">  <item android:id="@+id/call"  android:title="Call"  />  <item android:id="@+id/sms"  android:title="SMS"  />  </menu>  -------------option\_menu.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <menu xmlns:android="http://schemas.android.com/apk/res/android">  <item android:id="@+id/coffee"  android:title="Coffee"  />  <item android:id="@+id/tea"  android:title="Tea"  />  <item android:id="@+id/juice"  android:title="Juice"  />  <item android:id="@+id/water"  android:title="Water"  />  </menu>  ----------------pop\_up\_menu.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/settings"  android:icon="@drawable/ic\_baseline\_settings\_24"  android:title="Settings"  app:showAsAction="always"  />  <item android:id="@+id/share"  android:icon="@drawable/ic\_baseline\_share\_24"  android:title="Share"  app:showAsAction="always"  />  <item android:id="@+id/save"  android:icon="@drawable/ic\_baseline\_save\_24"  android:title="Save"  app:showAsAction="ifRoom"  />  <item android:id="@+id/logout"  android:icon="@drawable/ic\_baseline\_logout\_24"  android:title="Logout"  app:showAsAction="ifRoom"  />  </menu> | </end> |
| <hitle> | FloatingMenuDemo | <chare> | 1 | <pext> | 02-01/FloatingMenuDemo  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  binding.btnAdd.shrink()  }  private fun initViews() {  var isFabCollapsed = false  binding.apply {  btnAdd.setOnClickListener {  if(!isFabCollapsed) {  btnAdd.extend()  btnAlarm.show()  btnCall.show()  txtAlarm.visibility = View.VISIBLE  txtCall.visibility = View.VISIBLE  isFabCollapsed = true  } else {  btnAdd.shrink()  btnAlarm.hide()  btnCall.hide()  txtAlarm.visibility = View.GONE  txtCall.visibility = View.GONE  isFabCollapsed = false  }  }  }  }  }  <?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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:padding="15dp"  tools:context=".MainActivity">  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Hello World!"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:backgroundTint="#D86868"  android:text="Action"  android:textColor="@color/white"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:icon="@drawable/ic\_baseline\_add\_circle\_24"  android:id="@+id/btnAdd"  />  <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton  android:visibility="gone"  android:id="@+id/btnAlarm"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="10dp"  android:backgroundTint="#A33535"  android:tintMode="multiply"  app:icon="@drawable/ic\_baseline\_access\_alarm\_24"  app:layout\_constraintBottom\_toTopOf="@id/btnAdd"  app:layout\_constraintEnd\_toEndOf="parent"  />  <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton  android:id="@+id/btnCall"  android:visibility="gone"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginBottom="10dp"  android:backgroundTint="#A33535"  android:tintMode="multiply"  app:icon="@drawable/ic\_baseline\_add\_ic\_call\_24"  app:layout\_constraintBottom\_toTopOf="@id/btnAlarm"  app:layout\_constraintEnd\_toEndOf="parent"  />  <TextView  android:id="@+id/txtCall"  android:visibility="gone"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginEnd="25dp"  android:layout\_marginBottom="25dp"  android:text="Make a call"  android:textSize="25sp"  android:textStyle="bold"  app:layout\_constraintTop\_toTopOf="@+id/btnCall"  app:layout\_constraintBottom\_toBottomOf="@id/btnCall"  app:layout\_constraintEnd\_toStartOf="@+id/btnCall"  />  <TextView  android:id="@+id/txtAlarm"  android:visibility="gone"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginEnd="25dp"  android:layout\_marginBottom="25dp"  android:text="Add an alam"  android:textSize="25sp"  android:textStyle="bold"  app:layout\_constraintTop\_toTopOf="@+id/btnAlarm"  app:layout\_constraintBottom\_toBottomOf="@id/btnAlarm"  app:layout\_constraintEnd\_toStartOf="@+id/btnCall"  />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | FirstNotesAppTask | <chare> | 1 | <pext> | 02-02/FirstNotesAppTask  --------------MainActivity-----------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var notes: MutableList<Note>  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  binding.btnAddNote.shrink()  }  private fun initViews() {  binding.notesList.layoutManager = GridLayoutManager(this, 2)  notes = getNotes()  binding.notesList.adapter = NoteAdapter(this, notes)  binding.btnAddNote.setOnClickListener {  openEditorDialog()  }  }  override fun onCreateContextMenu(  menu: ContextMenu?,  v: View?,  menuInfo: ContextMenu.ContextMenuInfo?  ) {  super.onCreateContextMenu(menu, v, menuInfo)  menuInflater.inflate(R.menu.context\_menu, menu)  }  override fun onContextItemSelected(item: MenuItem): Boolean {  return when(item.itemId) {  R.id.call -> {  makeToast("Call")  true  }  R.id.sms -> {  makeToast("SMS")  true  }  else -> super.onContextItemSelected(item)  }  }  private fun openEditorDialog() {  val dialogNoteEditor: EditFormBinding = EditFormBinding.inflate(layoutInflater)  val builder = AlertDialog.Builder(this).apply {  setView(dialogNoteEditor.root)  setCancelable(false)  }  val dialog = builder.create()  dialog.window?.setGravity(Gravity.CENTER)  dialogNoteEditor.apply {  btnClose.setOnClickListener { dialog.dismiss() }  btnEditDone.setOnClickListener {  val title = editTitle.text.toString()  val body = editBody.text.toString()  if(!title.isEmpty()) {  notes.add(Note(title, body))  dialog.dismiss()  }  else {  makeToast("Please enter title!")  }  }  }  dialog.show()  }  private fun makeToast(message: String): Boolean {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  return true  }  private fun getNotes(): MutableList<Note> {  val listOfNotes = mutableListOf<Note>()  listOfNotes.apply {  add(Note("Name1", "this is the description"))  add(Note("Name2", "this is the description"))  add(Note("Name3", "this is the description"))  add(Note("Name4", "this is the description"))  add(Note("Name5", "this is the description"))  add(Note("Name6", "this is the description"))  add(Note("Name7", "this is the description"))  }  return listOfNotes  }  }  ------------Note/NoteAdapter-------------  data class Note (  var title: String,  var body: String,  ): Serializable  class NoteAdapter(private val context: Context, private var noteItems: MutableList<Note>): RecyclerView.Adapter<NoteAdapter.NoteViewHolder>() {  private lateinit var binding: NoteItemBinding  override fun getItemCount() = noteItems.size  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NoteViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NoteItemBinding.inflate(layoutInflater, parent, false)  return NoteViewHolder(binding.root)  }  override fun onBindViewHolder(holder: NoteViewHolder, position: Int) {  val note = noteItems[position]  holder.bind(noteItems[position], position)  }  private fun openEditorDialog(position: Int) {  val layoutInflater = LayoutInflater.from(context)  val dialogNoteEditor: EditFormBinding = EditFormBinding.inflate(layoutInflater)  val builder = AlertDialog.Builder(context).apply {  dialogNoteEditor.btnEditDone.text = "Edit"  dialogNoteEditor.editTitle.setText(noteItems[position].title)  dialogNoteEditor.editBody.setText(noteItems[position].body)  setView(dialogNoteEditor.root)  setCancelable(false)  }  val dialog = builder.create()  dialog.window?.setGravity(Gravity.CENTER)  dialogNoteEditor.apply {  btnClose.setOnClickListener { dialog.dismiss() }  btnEditDone.setOnClickListener {  val title = editTitle.text.toString()  val body = editBody.text.toString()  if(title.isNotEmpty()) {  noteItems[position] = Note(title, body)  notifyItemChanged(position)  dialog.dismiss()  }  else {  Toast.makeText(context, "Please enter title!", Toast.LENGTH\_SHORT).show()  }  }  }  dialog.show()  }  inner class NoteViewHolder(view: View) : RecyclerView.ViewHolder(view) {  fun bind(note: Note, position: Int) {  binding.apply {  txtTitle.text = note.title  txtBody.text = note.body  btnDelete.setOnClickListener {  noteItems.removeAt(position)  notifyItemRangeRemoved(position, 1)  }  btnEdit.setOnClickListener {  openEditorDialog(position)  }  }  }  }  }  ------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/notesList"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  />  <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:backgroundTint="#D86868"  android:layout\_margin="10dp"  android:text="Action"  android:textColor="@color/white"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:icon="@drawable/ic\_baseline\_add\_circle\_outline\_24"  android:id="@+id/btnAddNote"  />  </androidx.constraintlayout.widget.ConstraintLayout>  --------------note\_item.xml---------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="150dp"  android:layout\_height="wrap\_content"  xmlns:app="http://schemas.android.com/apk/res-auto"  android:backgroundTint="#F8CE4F"  android:padding="5dp"  android:elevation="15dp"  android:layout\_margin="10dp"  app:cardCornerRadius="10dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  >  <TextView  android:id="@+id/txtTitle"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:padding="5dp"  android:textSize="18sp"  android:textColor="@color/black"  android:textStyle="bold"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  android:text="Sender name"  />  <TextView  android:id="@+id/txtBody"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="5dp"  android:padding="5dp"  android:textSize="13sp"  android:textColor="@color/black"  android:gravity="start"  android:maxLines="4"  app:layout\_constraintTop\_toBottomOf="@id/txtTitle"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintBottom\_toBottomOf="@id/btnDelete"  android:text="Note body\nNote body\nNote body\nNote body"  />  <androidx.appcompat.widget.AppCompatImageButton  android:id="@+id/btnDelete"  android:layout\_width="20dp"  android:layout\_height="20dp"  android:layout\_margin="10dp"  android:src="@drawable/baseline\_delete\_24"  app:layout\_constraintTop\_toBottomOf="@id/txtBody"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <androidx.appcompat.widget.AppCompatImageButton  android:id="@+id/btnEdit"  android:layout\_width="20dp"  android:layout\_height="20dp"  android:layout\_margin="10dp"  android:src="@drawable/baseline\_edit\_note\_24"  app:layout\_constraintTop\_toBottomOf="@id/txtBody"  app:layout\_constraintEnd\_toStartOf="@id/btnDelete"  app:layout\_constraintBottom\_toBottomOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  -------------edit\_form.xml-------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editTitle"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@id/editBody"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_marginBottom="10dp"  android:layout\_marginLeft="10dp"  android:layout\_marginRight="10dp"  android:layout\_marginTop="40dp"  android:hint="Enter title"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editBody"  android:layout\_margin="10dp"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@id/editTitle"  android:hint="Enter note"  />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnEditDone"  android:layout\_margin="10dp"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/editBody"  android:backgroundTint="@color/purple\_200"  android:text="Add"  />  <androidx.appcompat.widget.AppCompatImageButton  android:id="@+id/btnClose"  android:layout\_width="40dp"  android:layout\_height="40dp"  android:src="@drawable/baseline\_close\_24"  android:backgroundTint="@color/purple\_200"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  android:layout\_margin="2dp"  />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | NavigationDrawDemo,Toolbar,Drawer, Menu | <chare> | 1 | <pext> | 02-02/NavigationDrawDemo  ---------Change themes.xml-----------------------------------------------  Res/values/themes  <style name="Theme.Day18\_2" parent="Theme.MaterialComponents.DayNight.NoActionBar">  ----------app\_features.xml----------------------------------------------  <?xml version="1.0" encoding="utf-8"?>  <menu xmlns:android="http://schemas.android.com/apk/res/android">  <group android:checkableBehavior="single">  <item android:id="@+id/nav\_profile"  android:title="Profile"  android:icon="@drawable/ic\_baseline\_person\_24"  />  <item android:id="@+id/nav\_wallet"  android:title="Wallet"  android:icon="@drawable/ic\_baseline\_wallet\_24"  />  </group>  <item android:title="Extras"  >  <menu>  <group android:checkableBehavior="single">  <item android:id="@+id/nav\_offer"  android:title="Offer"  android:icon="@drawable/ic\_baseline\_local\_offer\_24"  />  <item android:id="@+id/nav\_settings"  android:title="Settings"  android:icon="@drawable/ic\_baseline\_settings\_24"  />  <item android:id="@+id/nav\_logout"  android:title="Logout"  android:icon="@drawable/ic\_baseline\_logout\_24"  />  </group>  </menu>  </item>  </menu>  ---------header\_layout.xml-----------------------------------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="20dp"  android:background="#F1DFDF"  xmlns:app="http://schemas.android.com/apk/res-auto">  <androidx.cardview.widget.CardView  android:id="@+id/cardImage"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  app:cardCornerRadius="30dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  >  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/senderImage"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:scaleType="centerCrop"  android:src="@drawable/actor2"  />  </androidx.cardview.widget.CardView>  <TextView  android:id="@+id/txtName"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:paddingStart="10dp"  android:paddingEnd="5dp"  android:paddingBottom="5dp"  android:gravity="center\_vertical"  android:text="Name1"  android:textSize="20dp"  android:textStyle="bold"  app:layout\_constraintStart\_toEndOf="@id/cardImage"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@id/txtEmail"  />  <TextView  android:id="@+id/txtEmail"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:paddingStart="10dp"  android:paddingEnd="5dp"  android:paddingBottom="5dp"  android:gravity="center\_vertical"  android:text="sample@gmail.com"  android:textSize="16dp"  android:textStyle="bold"  app:layout\_constraintStart\_toEndOf="@+id/cardImage"  app:layout\_constraintTop\_toBottomOf="@id/txtName"  app:layout\_constraintBottom\_toBottomOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  --------------activity\_main.xml------------------------------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.drawerlayout.widget.DrawerLayout  xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/drawerLayout"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <FrameLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  >  <androidx.appcompat.widget.Toolbar  android:id="@+id/toolbar"  android:layout\_width="match\_parent"  android:layout\_height="50dp"  android:background="@color/purple\_700"  android:gravity="top"  >  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  >  <TextView  android:id="@+id/txtToolbar"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Title of the screen"  android:textSize="25sp"  android:textColor="@color/white"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.appcompat.widget.Toolbar>  </FrameLayout>  <com.google.android.material.navigation.NavigationView  android:id="@+id/nav\_menu"  android:layout\_width="wrap\_content"  android:layout\_height="match\_parent"  android:layout\_gravity="start"  app:menu="@menu/app\_features"  app:headerLayout="@layout/header\_layout"  />  </androidx.drawerlayout.widget.DrawerLayout>  -----------MainActivity---------------------------------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  setSupportActionBar(binding.toolbar)  supportActionBar?.apply {  setDisplayHomeAsUpEnabled(true)  setHomeAsUpIndicator(R.drawable.ic\_baseline\_menu\_24)  }  binding.navMenu.setNavigationItemSelectedListener { menuItems ->  menuItems.isChecked = true  when(menuItems.itemId) {  R.id.nav\_profile -> makeToast("Profile")  R.id.nav\_offer -> makeToast("Offer")  R.id.nav\_settings -> makeToast("Settings")  R.id.nav\_wallet -> makeToast("Wallet")  R.id.nav\_logout -> makeToast("Logout")  }  true  }  }  override fun onOptionsItemSelected(item: MenuItem): Boolean {  if(item.itemId == android.R.id.home) {  if(binding.drawerLayout.isDrawerOpen(GravityCompat.START)) {  binding.drawerLayout.closeDrawer(GravityCompat.START)  } else {  binding.drawerLayout.openDrawer(GravityCompat.START)  }  }  return super.onOptionsItemSelected(item)  }  private fun makeToast(message: String): Boolean {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  return true  }  } | </end> |
| <hitle> | ProgressSnackBarDemo | <chare> | 1 | <pext> | 02-02/ProgressSnackBarDemo  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  makeSnackBar()  }  private fun makeSnackBar() {  val snackBar = Snackbar.make(this, binding.content, "hey this is snackBar", Snackbar.LENGTH\_LONG)  snackBar.apply {  setActionTextColor(Color.MAGENTA)  val snackBarView = snackBar.view  val textView =  snackBarView.findViewById<TextView>(com.google.android.material.R.id.snackbar\_text)  textView.textSize = 25f  textView.setTextColor(Color.RED)  snackBar.setAction("Retry") {  binding.root.setBackgroundColor(Color.YELLOW)  }  }  snackBar.show()  }  }  <?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"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/content"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <ProgressBar  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:indeterminate="true"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  <com.google.android.material.progressindicator.CircularProgressIndicator  android:layout\_width="100dp"  android:layout\_height="100dp"  android:indeterminate="true"  app:indicatorColor="@color/teal\_700"  app:indicatorSize="100dp"  app:trackThickness="10dp"  app:trackColor="@color/purple\_500"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <com.google.android.material.progressindicator.CircularProgressIndicator  android:layout\_width="60dp"  android:layout\_height="60dp"  android:indeterminate="true"  android:layout\_marginStart="20dp"  app:indicatorColor="@color/teal\_700"  app:indicatorSize="60dp"  app:trackThickness="10dp"  app:trackColor="@color/purple\_500"  app:indicatorDirectionCircular="counterclockwise"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  <com.google.android.material.progressindicator.LinearProgressIndicator  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  android:indeterminate="true"  android:layout\_marginTop="50dp"  app:indicatorColor="@color/teal\_200"  app:indicatorSize = "60dp"  app:trackCornerRadius="40dp"  android:progressTint="@color/purple\_200"  app:indeterminateAnimationType="disjoint"  app:indicatorDirectionLinear="leftToRight"  app:trackThickness="10dp"  app:trackColor="@color/purple\_700"  />  <com.google.android.material.progressindicator.LinearProgressIndicator  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent"  android:indeterminate="true"  android:layout\_marginTop="100dp"  app:indicatorColor="@color/teal\_200"  app:indicatorSize = "60dp"  app:trackCornerRadius="40dp"  android:progressTint="@color/purple\_200"  app:indeterminateAnimationType="disjoint"  app:indicatorDirectionLinear="rightToLeft"  app:trackThickness="10dp"  app:trackColor="@color/purple\_700"  />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | ViewPager2Demo | <chare> | 1 | <pext> | 02-03/ViewPager2Demo  ------------------ MainActivity ----------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewPagerAdapter: ViewPagerAdapter  private lateinit var tabBdinding: CustomTabLayoutBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpViewPager()  setUpTabLayout()  }  private fun setUpTabLayout() {  TabLayoutMediator(binding.tabLayout, binding.viewPager){tab, position ->  tab.text = "OBJECT ${(position+1)}"  }.attach()  binding.tabLayout.apply {  tabBdinding = CustomTabLayoutBinding.inflate(LayoutInflater.from(this@MainActivity))  tabBdinding.imgTab.setImageResource(R.drawable.baseline\_newspaper\_24)  tabBdinding.txtTabTitle.text = "News"  getTabAt(0)?.apply {  customView = tabBdinding.root  }  val tabBdinding2 = CustomTabLayoutBinding.inflate(LayoutInflater.from(this@MainActivity))  tabBdinding2.imgTab.setImageResource(R.drawable.baseline\_sports\_basketball\_24)  tabBdinding2.txtTabTitle.text = "Sports"  getTabAt(1)?.apply {  customView = tabBdinding2.root  }  val tabBdinding3 = CustomTabLayoutBinding.inflate(LayoutInflater.from(this@MainActivity))  tabBdinding3.imgTab.setImageResource(R.drawable.baseline\_trending\_up\_24)  tabBdinding3.txtTabTitle.text = "Trending"  getTabAt(2)?.apply {  customView = tabBdinding3.root  }  }  }  private fun setUpViewPager() {  binding.viewPager.adapter = ViewPagerAdapter(this, 3)  }  }  --------------ViewPagerAdapter ---------------  class ViewPagerAdapter(fragmentActivity: FragmentActivity, private val totalCount: Int):  FragmentStateAdapter(fragmentActivity) {  override fun getItemCount() = totalCount  override fun createFragment(position: Int): Fragment {  return when (position) {  0 -> NewsFragment()  1 -> SportsFragment()  2 -> TrendingFragment()  else -> NewsFragment()  }  }  }  ----------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <com.google.android.material.tabs.TabLayout  android:id="@+id/tabLayout"  android:layout\_width="match\_parent"  android:layout\_height="100dp"  android:background="@color/purple\_200"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  <androidx.viewpager2.widget.ViewPager2  android:id="@+id/viewPager"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/tabLayout"  />  </androidx.constraintlayout.widget.ConstraintLayout>  ---------------custom\_tab\_layou.xml----------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  xmlns:app="http://schemas.android.com/apk/res-auto">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgTab"  android:layout\_width="40dp"  android:layout\_height="40dp"  android:src="@drawable/ic\_launcher\_background"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  <TextView  android:id="@+id/txtTabTitle"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textColor="@color/white"  android:textSize="25dp"  android:textStyle="bold"  android:text="Hello Everyone!"  app:layout\_constraintTop\_toBottomOf="@+id/imgTab"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | Git/Whatsapp-Chatting-Sample\_Work | <chare> | 1 | <pext> | 02-03/Git/Whatsapp-Chatting-Sample\_Work  https://github.com/cheetahmail007/Whatsapp-Chatting-Sample\_Work | </end> |
| <hitle> | SQLiteDemo1 | <chare> | 1 | <pext> | 02-06/SQLiteDemo1  ---------- DatabaseHelper.kt----------------------------------------------  class DatabaseHelper(context: Context): SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {  override fun onCreate(db: SQLiteDatabase?) {  val query =  ("CREATE TABLE $TABLE\_NAME (" +  "$ID\_COL INTEGER PRIMARY KEY, " +  "$NAME\_COL TEXT, " +  "$AUTHOR\_COL TEXT, " +  "$PRICING\_COL TEXT, " +  "$RATING\_COL TEXT )")  db?.execSQL(query)  }  override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {  db?.execSQL("DROP TABLE IF EXISTS $TABLE\_NAME")  onCreate(db)  }  }  -----------BookDao.kt---------------------------------------------  class BookDao(private val databaseHelper: DatabaseHelper) {  private val db: DatabaseHelper = databaseHelper  init {  }  fun addBook(book: Book) {  val contentValue = ContentValues().apply {  book.apply {  put(Constant.ID\_COL, id)  put(Constant.NAME\_COL, name)  put(Constant.AUTHOR\_COL, author)  put(Constant.PRICING\_COL, price)  put(Constant.RATING\_COL, rating)  }  }  db.writableDatabase.apply {  insert(Constant.TABLE\_NAME, null, contentValue)  }  }  fun getAllBooks() : ArrayList<Book> {  val books = ArrayList<Book>()  val cursor: Cursor = db.readableDatabase.query(Constant.TABLE\_NAME, null, null, null, null, null, null, null)  if(cursor.count > 0) {  while (cursor.moveToNext()) {  val bookId = cursor.getInt(0) ?: 0  val name = cursor.getString(1) ?: ""  val author = cursor.getString(2) ?: ""  val price = cursor.getString(3) ?: ""  val rate = cursor.getString(4) ?: ""  val book = Book(name, author, price, rate, bookId)  books.add(book)  }  }  return books  }  fun updateBook(book: Book) {  val contentValue = ContentValues().apply {  book.apply {  put(Constant.NAME\_COL, name)  put(Constant.AUTHOR\_COL, author)  put(Constant.PRICING\_COL, price)  put(Constant.RATING\_COL, rating)  }  }  db.writableDatabase.apply {  update(Constant.TABLE\_NAME, contentValue,  "$ID\_COL = ${book.id}", null)  }  }  fun deleteBook(bookId: Int) {  db.writableDatabase.apply {  delete(TABLE\_NAME, "$ID\_COL = $bookId", null)  //close  }  }  }  ------------MainActivity--------------------------------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var databaseHelper: DatabaseHelper  private lateinit var bookDao: BookDao  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initDatabase()  initViews()  }  private fun initDatabase() {  databaseHelper = DatabaseHelper(this.applicationContext)  bookDao = BookDao(databaseHelper)  }  private fun initViews() {  binding.btnAdd.setOnClickListener {  bookDao.addBook(Book("War and Peace", "Shakespere", "$100", "5 star", 300))  }  binding.btnAllBooks.setOnClickListener {  val name = bookDao.getAllBooks()[0].name  Toast.makeText(this, name, Toast.LENGTH\_SHORT).show()  }  binding.btnUpdate.setOnClickListener {  bookDao.updateBook(Book("Testing", "testing2", "$50", "3 star", 300))  }  binding.btnDelete.setOnClickListener {  bookDao.deleteBook(300)  }  }  }  ------------- Constant.kt--------------------------------  object Constant {  const val DATABASE\_NAME = "book-db"  const val DATABASE\_VERSION = 1  const val TABLE\_NAME = "books"  const val ID\_COL = "id"  const val NAME\_COL = "name"  const val AUTHOR\_COL = "author"  const val PRICING\_COL = "pricing"  const val RATING\_COL = "rating"  }  -------Book.kt-------------------------------------------------  data class Book(  val name:String,  val author: String,  val price: String,  val rating: String,  val id: Int  ) | </end> |
| <hitle> | Git/BookingAppointmentSystem | <chare> | 1 | <pext> | 02-07/Git/ BookingAppointmentSystem  https://github.com/cheetahmail007/BookingAppointmentSystem | </end> |
| <hitle> | VolleyDemo1 | <chare> | 1 | <pext> | 02-08/VolleyDemo1  --------AndroidManifest--------  <uses-permission android:name="android.permission.INTERNET" />  ----------makeApiCall------------  private fun makeApiCallToGetRandomDog() {  binding.picture.visibility = View.VISIBLE  val requestQueue: RequestQueue = Volley.newRequestQueue(this)  val request = StringRequest(  Request.Method.GET,  API\_URL,  {  //success block  apiResponse: String ->  val typeToken = object : TypeToken<DogResponse>(){}  val gson = Gson()  try{  val response: DogResponse = gson.fromJson(apiResponse,typeToken.type)  Glide.with(this)  .load(response.url)  .error(android.R.drawable.ic\_dialog\_alert)  .placeholder(R.drawable.ic\_launcher\_background)  .into(binding.picture)  binding.picture.visibility = View.VISIBLE  binding.txtName.text = response.artist  }catch(e: Exception){  e.printStackTrace()  binding.picture.visibility = View.GONE  binding.txtName.text = "Error"  }  },  {  //error block  binding.picture.visibility = View.GONE  println(it.message.toString())  }  )  requestQueue.add(request)  }  companion object {  const val API\_URL = "https://api.catboys.com/img"  }  ------------Gradle----------  // Glide  implementation 'com.github.bumptech.glide:glide:4.13.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.13.2'  // volley for API calls  implementation 'com.android.volley:volley:1.2.1'  // gson for type conversion for data source  implementation 'com.google.code.gson:gson:2.9.0' | </end> |
| <hitle> | VolleyDemo2 Local/Post/FetchList | <chare> | 1 | <pext> | 02-09/VolleyDemo2 Local/Post/FetchList  -----------------Manifest Post/Register-----------  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnRegitser.setOnClickListener {  registerUser()  }  binding.btnFetch.setOnClickListener {  startActivity(Intent(this@MainActivity, ListActivity::class.java))  }  }  private fun registerUser() {  if(binding.editId.text.toString().isEmpty()) {  makeToast("Please enter ID1")  return  }  val newCar = Car(binding.editId.text.toString().toInt() ,  binding.editMake.text.toString(),  binding.editModel.text.toString(),  binding.editVariant.text.toString(),  binding.editColor.text.toString(),  binding.editLaunchYear.text.toString(),  binding.editFuelType.text.toString()  )  val registerData = JSONObject()  registerData.apply {  put( "id", newCar.id )  put( "make", newCar.make )  put( "model", newCar.model )  put( "variant", newCar.variant )  put( "color", newCar.color )  put( "launch\_year", newCar.launch\_year )  put( "fuel\_type", newCar.fuel\_type )  }  val requestQueue = Volley.newRequestQueue(this@MainActivity)  val request = JsonObjectRequest(  Request.Method.POST,  API\_URL,  registerData,  {  makeToast("Registered successfully!")  },  {  //error block  makeToast(it.message.toString())  }  )  val retryPolicy = DefaultRetryPolicy(2000,3,1.5f)  request.retryPolicy = retryPolicy  requestQueue.add(request)  }  private fun makeToast(message: String): Boolean {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  return true  }  companion object {  const val API\_URL = "http://10.0.2.2/api/index.php/Car"  const val CAR\_LIST\_API = "http://10.0.2.2/api/index.php/Car"  }  }  --------------ListActivity Fetch------------  class ListActivity : AppCompatActivity() {  lateinit var binding: ActivityListBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityListBinding.inflate(layoutInflater)  setContentView(binding.root)  fetchRemote()  }  fun fetchRemote() {  val requestQueue = Volley. newRequestQueue(this@ListActivity)  val listRequest = object: StringRequest(  Method.GET,  API\_URL,  {  val responseOfList: CarListReponse = Gson().fromJson(it, CarListReponse::class.java)  if(responseOfList.cars.isNotEmpty()) {  updateList(responseOfList);  }  },  {  finish()  }  ) {  }  requestQueue.add(listRequest)  }  fun updateList(response: CarListReponse) {  val listData = response.cars.map {  "${it.id} : ${it.model} - ${it.color}"  }  binding.listView.adapter = ArrayAdapter(this, android.R.layout.simple\_list\_item\_1, listData)  }  }  -------------Manifest permission/usesCleartextTraffic------------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.VolleyDemo2"  android:usesCleartextTraffic="true"  tools:targetApi="31">  <activity  android:name=".ListActivity"  android:exported="false" />  <activity  android:name=".MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  --------------Data Model------------  data class Car(  val id: Int,  val make: String,  val model: String,  val variant: String,  val color: String,  val launch\_year: String,  val fuel\_type: String  )  data class CarListReponse (  val status : Int,  val message : String,  val cars: ArrayList<Car>,  )  --------------Gradel-------------------  // Glide  implementation 'com.github.bumptech.glide:glide:4.13.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.13.2'  // volley for API calls  implementation 'com.android.volley:volley:1.2.1'  // gson for type conversion for data source  implementation 'com.google.code.gson:gson:2.9.0' | </end> |
| <hitle> | VolleyDemo3 Auth, Cancel, Priority, Header | <chare> | 1 | <pext> | 02-09/VolleyDemo3 Auth, Cancel, Priority, Header  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var requestQueue: RequestQueue  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  fetchWeatherReporter()  binding.button.setOnClickListener {  -------------------Cancel request------------  requestQueue.cancelAll(REQUEST\_TAG)  }  }  fun fetchWeatherReporter() {  requestQueue = Volley.newRequestQueue(this@MainActivity)  val request = object: StringRequest(  Method.GET,  URL,  {  val response = Gson().fromJson(it, WeatherResponse::class.java)  Toast.makeText(this, "Success", Toast.LENGTH\_SHORT).show()  },  {  Toast.makeText(this, "Failed", Toast.LENGTH\_SHORT).show()  }  ) {  --------------Authorization Header----------------  override fun getHeaders(): MutableMap<String, String> {  val map = HashMap<String, String>()  map["Authorization"] = API\_KEY  return map  }  ---------------BodyContentType---------------  override fun getBodyContentType(): String {  return "application/json; charset=utf-8"  }  ---------------Whatto on cancel---------  override fun cancel() {  super.cancel()  //After API got cancel this block got executed  //Like once canceled, we can stop updating UI.  }  ----------------Priority--------------  override fun getPriority(): Priority {  return Priority.HIGH //Priority.LOW, Priority.IMMEDIATE  }  }  val retryPolicy = DefaultRetryPolicy(20000, 3, 1.5f)  request.retryPolicy = retryPolicy  request.tag = REQUEST\_TAG  //anotherRequest.tag = REQUEST\_TAG  requestQueue.add(request)  }  companion object {  const val API\_KEY = "890027cf7c50f78f2eb20ba339b532e6"  const val URL= "https://api.openweathermap.org/data/2.5/weather?lat=44.34&lon=10.99&appid=$API\_KEY"  const val API\_KEY\_FOR\_NEWS = "hJKwiqk7NiC7myrGJftYaGC3ZnrUKOYm3Sqw3qq8nO1pNafM"  const val BASE\_URL = "https://api.currentsapi.services/v1/"  const val END\_POINT = "/search?keyword=London"  const val FILTER\_LANG = "&language=en"  const val REQUEST\_TAG = "Volley Request"  }  } | </end> |
| <hitle> | VolleyDemo4 ImageCaching Request | <chare> | 1 | <pext> | 02-10/VolleyDemo4 ImageCaching Request  \*\*\*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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  -------------NetworkImageView in xml--------------  <com.android.volley.toolbox.NetworkImageView  android:id="@+id/image"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toTopOf="@id/btnCustom"  />  <Button  android:id="@+id/btnBasic"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toStartOf="@id/btnCustom"  android:text="Basic"  />  <Button  android:id="@+id/btnCustom"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Custom"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toStartOf="@+id/btnError"  app:layout\_constraintStart\_toEndOf="@id/btnBasic" />  <Button  android:id="@+id/btnError"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Error"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/btnCustom" />  </androidx.constraintlayout.widget.ConstraintLayout>  ----------------ImageCachingVolleyHandler-------------  class ImageCachingVolleyHandler(context: Context) {  val imageLoader: ImageLoader  private var requestQueue: RequestQueue  init {  val cache = DiskBasedCache(  context.cacheDir,  10 \* 1024 \* 1024 //10mb  )  val network = BasicNetwork(HurlStack())  requestQueue = RequestQueue(cache, network).also { it.start() }  imageLoader = ImageLoader(  requestQueue,  object : ImageLoader.ImageCache {  private val \_cache = LruCache<String, Bitmap>(15)  override fun getBitmap(url: String): Bitmap? {  return \_cache[url]  }  override fun putBitmap(url: String?, bitmap: Bitmap?) {  \_cache.put(url, bitmap)  }  }  )  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  lateinit var imageLoader: ImageLoader  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  initLoaders()  }  -------------init imageLoader-------------  private fun initLoaders() {  imageLoader = ImageCachingVolleyHandler(this).imageLoader  }  private fun initViews() {  val url1 = "https://howdyev.github.io/static/service\_content\_image\_3-9131dc87559217dc96be4fffb9986787.jpg"  val url2 = "https://howdyev.github.io/static/service\_content\_image\_2-009264fe94068ddd8385aef32422a675.jpg"  val url3 = "https://errorImage"  binding.btnBasic.setOnClickListener {  fetchImageBasicVersion(url1, binding.image, imageLoader)  }  binding.btnCustom.setOnClickListener {  fetchImageCustomVersion(url2, binding.image, imageLoader,  R.drawable.ic\_launcher\_background,  R.drawable.baseline\_report\_gmailerrorred\_24  )  }  binding.btnError.setOnClickListener {  fetchImageCustomVersion(url3, binding.image, imageLoader,  R.drawable.ic\_launcher\_background,  R.drawable.baseline\_report\_gmailerrorred\_24  )  }  }  -----------fetch and set networkImageView using imageLoader------------  private fun fetchImageBasicVersion(  url: String,  networkImageView: NetworkImageView,  imageLoader: ImageLoader  ) = networkImageView.setImageUrl(url, imageLoader)  private fun fetchImageCustomVersion(  url: String,  networkImageView: NetworkImageView,  imageLoader: ImageLoader,  @DrawableRes placeholder: Int,  @DrawableRes errorImage: Int  ) {  imageLoader.get(  url,  ImageLoader.getImageListener(  binding.image,  placeholder,  errorImage  )  )  networkImageView.setImageUrl(url, imageLoader)  }  }  \*\*\*AndroidManifest\*\*\*  <uses-permission android:name="android.permission.INTERNET" />  \*\*\*Gradle\*\*\*  // Glide  implementation 'com.github.bumptech.glide:glide:4.13.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.13.2'  // volley for API calls  implementation 'com.android.volley:volley:1.2.1'  // gson for type conversion for data source  implementation 'com.google.code.gson:gson:2.9.0' | </end> |
| <hitle> | MVPDemo1 Volley | <chare> | 1 | <pext> | 02-10/MVPDemo1 Volley  ------------Implement model layer-------------  \*\*\*model/remote/\*\*\*  object Constant {  const val API\_URL ="https://dog.ceo/api/breeds/image/random"  }  data class DogResponse(  val message: String,  val status: String  )  class VolleyHandler(private val context: Context) {  fun makeApiCallToGetRandomDog(callback: OperationalCallback) {  val requestQueue: RequestQueue = Volley.newRequestQueue(context)  val request = StringRequest(  Request.Method.GET,  API\_URL,  {  val apiResponse = Gson().fromJson(it, DogResponse::class.java)  callback.onSuccess(apiResponse)  },  {  callback.onFailure(it.toString())  }  )  requestQueue.add(request)  }  }  interface OperationalCallback {  fun onSuccess(dogResponse: DogResponse)  fun onFailure(message: String)  }  -------------Create View interface/Presenter Interface---------  \*\*\*presenter/\*\*\*  interface DogMVP {  interface DogPresenter {  fun getDog()  }  interface DogView {  fun setResult(dogResponse: DogResponse)  fun onLoad(isLoading: Boolean)  fun showError(message: String)  }  }  ------------Implement Presenter------------  \*\*\*presenter/\*\*\*  class DogPresenter (  private val volleyHandler: VolleyHandler,  private val dogView: DogMVP.DogView  ): DogMVP.DogPresenter {  override fun getDog() {  dogView.onLoad(true)  volleyHandler.makeApiCallToGetRandomDog(object: OperationalCallback {  override fun onSuccess(dogResponse: DogResponse) {  dogView.onLoad(false)  dogView.setResult(dogResponse)  }  override fun onFailure(message: String) {  dogView.onLoad(false)  dogView.showError(message)  }  })  }  }  ------------Implement View Layer-------------  \*\*\*view/\*\*\*  class MainActivity : AppCompatActivity(), DogMVP.DogView {  lateinit var binding: ActivityMainBinding  lateinit var presenter: DogPresenter  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  ----------Init Presenter in the View----------  presenter = DogPresenter(VolleyHandler(this), this)  binding.btnGetRandomDog.setOnClickListener {  -------call presenter from UI---------------  presenter.getDog()  }  }  ----------provide body to all MVP view methods-----------  override fun onLoad(isLoading: Boolean) {  if(isLoading) {  binding.loader.visibility = View.VISIBLE  } else {  binding.loader.visibility = View.GONE  }  }  override fun setResult(dogResponse: DogResponse) {  Glide.with(this)  .load(dogResponse.message)  .error(android.R.drawable.ic\_dialog\_alert)  .placeholder(R.drawable.ic\_launcher\_background)  .into(binding.imageOfDog)  }  override fun showError(message: String) {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  }  }  \*\*\*AndroidManifest\*\*\*  <uses-permission android:name="android.permission.INTERNET" />  \*\*\*Gradle\*\*\*  // Glide  implementation 'com.github.bumptech.glide:glide:4.13.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.13.2'  // volley for API calls  implementation 'com.android.volley:volley:1.2.1'  // gson for type conversion for data source  implementation 'com.google.code.gson:gson:2.9.0' | </end> |
| <hitle> | Git/Weather-Widget-Demo | <chare> | 1 | <pext> | 02-10/Git/Weather-Widget-Demo  https://github.com/cheetahmail007/Weather-Widget-Demo | </end> |
| <hitle> | RetrofitDemo1 | <chare> | 1 | <pext> | 02-14/RetrofitDemo1  --------------Constant---------------------  object Constant {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  ------------------ApiService------------------  interface ApiService {  @GET(END\_POINT)  fun getRandomDog(): Call<DogResponse>  }  ------------------DogResponse---------------  data class DogResponse(  val message: String,  val status: String  )  ------------------RetrofitBuilder---------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .build()  }  return retrofit  }  }  -----------------MainActivity-------------------  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  lateinit var apiService: ApiService  lateinit var retrofit: Retrofit  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  binding.button.setOnClickListener {  makeAPICall()  }  }  private fun makeAPICall() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  apiService.getRandomDog().enqueue(object :Callback<DogResponse> {  override fun onResponse(call: Call<DogResponse>, response: Response<DogResponse>) {  if(response.isSuccessful) {  Picasso.get().load(response.body()?.message).into(binding.imageView)  }  }  override fun onFailure(call: Call<DogResponse>, t: Throwable) {  Log.i("tag", t.message.toString())  }  })  }  }  -------------activity\_main.xml-----------------------  imageView, button | </end> |
| <hitle> | Howto Retrofit Walkthrough | <chare> | 1 | <pext> | 02-14/Howto Retrofit Walkthrough  --------------Create ApiService interface using Retrofit Annotations------------  interface ApiService {  //In this API call, we are fetching a post on the basis of post id  @GET("posts/{postId}")  fun getPosts(  @Path("postId") postId: String,  ): Call<PostsResponse>  //In this API call, we are fetching all the posts available  @GET(END\_POINT)  fun getAllPosts(): Call<PostAllData>  //In this API call, we are adding a new post by using field  @FormUrlEncoded  @POST(END\_POINT)  fun postPosts(  @Field("title") title: String,  @Field("body") body: String,  @Field("userId") userId: String  ): Call<PostsResponse>  //In this API call, we are adding a new post by using body  @POST(END\_POINT)  fun postPosts(@Body postRequest: PostRequest): Call<PostsResponse>  //In this API call, we are update the post details  @FormUrlEncoded  @PUT("posts/{postId}")  fun updatePosts(  @Path("postId") postId: String,  @Field("id") id: String,  @Field("title") title: String,  @Field("body") body: String,  @Field("userId") userId: String  ): Call<PostsResponse>  @FormUrlEncoded  @PATCH("posts/{postId}")  fun updatePostTitle(  @Path("postId") id: String, @Field("title") title: String  ): Call<PostsResponse>  //In this API call we are deleting a post on the basis of post Id  //which received as parameter  @DELETE("posts/{id}")  fun deletePost(@Path("id") id: String): Call<Unit>  }  -----------make Retrofit builder-----------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  -----------make apiService instance----------  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  -----------use apiService interface methods----------  private fun getAPost() {  apiService.getPosts("1")  .enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  Log.i("tag", response.message())  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  }  private fun getALlPosts() {  apiService.getAllPosts()  .enqueue(object : Callback<PostAllData> {  override fun onResponse(  call: Call<PostAllData>,  response: Response<PostAllData>  ) {  Log.i("tag", response.message())  }  override fun onFailure(call: Call<PostAllData>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  }  private fun deletePost() {  apiService.deletePost("1")  .enqueue(object : Callback<Unit> {  override fun onResponse(  call: Call<Unit>,  response: Response<Unit>  ) {  Log.i("tag", response.message())  }  override fun onFailure(call: Call<Unit>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  }  private fun updatePostTitle() {  apiService.updatePostTitle("1", "London")  .enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  Log.i("tag", response.body()?.title!!)  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  }  private fun updatePosts() {  apiService.updatePosts("1", "1", "London", "good city", "1")  .enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  Log.i("tag", response.body()?.title!!)  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  }  private fun postComment() {  apiService.postPosts("London", "good city", "1")  .enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  Log.i("tag", response.body()?.title!!)  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  }  private fun postPosts() {  apiService.postPosts(PostRequest("London", "good city", "1"))  .enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  Log.i("tag", response.body()?.title!!)  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  // binding.loader.visibility = View.GONE  Log.i("tag", t.message.toString())  }  })  } | </end> |
| <hitle> | FragmentPassDataDemo | <chare> | 1 | <pext> | 02-15/FragmentPassDataDemo  ------1. Activity to Fragment through args---------  -------prepare args in Fragment to receive data-------  private const val ARG\_PARAM1 = "param1"  class BlankFragment : Fragment() {  private var param1: String? = null  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  arguments?.let {  param1 = it.getString(ARG\_PARAM1)  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  val view = inflater.inflate(R.layout.fragment\_blank, container, false)  ---------process the data received through args-----------  view.findViewById<TextView>(R.id.textView).text = param1  return view  }  companion object {  @JvmStatic  fun newInstance(param1: String) =  BlankFragment().apply {  arguments = Bundle().apply {  putString(ARG\_PARAM1, param1)  }  }  }  }  \*\*\*MainActivity\*\*\*  private fun initViews() {  findViewById<Button>(R.id.btnSubmit).setOnClickListener {  var name = findViewById<EditText>(R.id.editName).text.toString()  // var fragmentManager = supportFragmentManager  // var fragmentTransaction = fragmentManager.beginTransaction()  // fragmentTransaction.add(R.id.fragmentContainer, BlankFragment()).commit()  -----------Activity send data to Fragment through args----------  supportFragmentManager  .beginTransaction()  .add(R.id.fragmentContainer, BlankFragment.newInstance(name))  .commit()  }  }  \*\*\*activity\_main.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout 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"  android:orientation="vertical"  tools:context=".MainActivity">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical"  >  <EditText  android:id="@+id/editName"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:hint="Enter name"  />  <Button  android:id="@+id/btnSubmit"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="Submit"  />  </LinearLayout>  <FrameLayout  android:id="@+id/fragmentContainer"  android:layout\_weight="1"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  />  </LinearLayout>  \*\*\*fragment\_blank.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"  android:background="@color/purple\_200"  tools:context=".BlankFragment">  <TextView  android:id="@+id/textView"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:textSize="30sp"  android:text="@string/hello\_blank\_fragment" />  </FrameLayout>  --------2. Fragment send data to Activity through interface-------  \*\*\*BlankFragment2\*\*\*  class BlankFragment2 : Fragment() {  ----------define FragmentListener interface in Frag------  interface OnFragmentListener {  fun onClick(name: String)  }  var listener: OnFragmentListener? = null  ----------save parentActivity instance-------------  override fun onAttach(context: Context) {  super.onAttach(context)  listener = context as MainActivity2  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  // Inflate the layout for this fragment  val view = inflater.inflate(R.layout.fragment\_blank2, container, false)  init(view)  return view  }  private fun init(view: View) {  view.findViewById<Button>(R.id.btnSubmit).setOnClickListener {  var name = view.findViewById<EditText>(R.id.editName).text.toString()  ---------send data to Activity through interface-------------  listener?.onClick(name)  }  }  }  \*\*\*MainActivity2\*\*\*  --------Implement FragmentListener interface in activity-----------  class MainActivity2 : AppCompatActivity(), BlankFragment2.OnFragmentListener {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main2)  init()  }  private fun init() {  supportFragmentManager  .beginTransaction()  .add(R.id.fragment\_container2, BlankFragment2())  .commit()  }  ----------Implement FragmentListener callback-----------  override fun onClick(name: String) {  findViewById<TextView>(R.id.txtName).text = name  }  }  \*\*\*activity\_main2.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout 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"  android:orientation="vertical"  android:weightSum="2"  tools:context=".MainActivity2">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  android:orientation="vertical"  >  <TextView  android:id="@+id/txtName"  android:text="Hello"  android:textSize="45sp"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  />  </LinearLayout>  <FrameLayout  android:id="@+id/fragment\_container2"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  />  </LinearLayout>  \*\*\*blank\_fragment2.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:background="@color/teal\_200"  android:padding="10dp"  android:orientation="vertical"  tools:context=".BlankFragment2">  <EditText  android:id="@+id/editName"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:hint="Enter name"  />  <Button  android:id="@+id/btnSubmit"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="Submit Data" />  </LinearLayout> | </end> |
| <hitle> | PassData\_FragmentA\_FragmentB | <chare> | 1 | <pext> | 02-15/PassData\_FragmentA\_FragmentB  \*\*\*FragmentA\*\*\*  class OneFragment : Fragment() {  lateinit var binding: FragmentOneBinding  --------Define FragmentALinsener interface  interface OnFragmentInteraction {  fun onClick(name: String)  }  var listener: OnFragmentInteraction? = null  --------Save parent activity in FragmentA--------  override fun onAttach(context: Context) {  super.onAttach(context)  listener = context as MainActivity  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  // Inflate the layout for this fragment  binding = FragmentOneBinding.inflate(layoutInflater, container, false)  initViews()  return binding.root  }  private fun initViews() {  binding.btnSend.setOnClickListener {  val name = binding.editName.text.toString()  -----------send data to activity through listener callback-------  listener?.onClick(name)  }  }  }  \*\*\*FragmentB\*\*\*  ---------prepare args in FragmentB-----------  private const val ARG\_PARAM1 = "param1"  class TwoFragment : Fragment() {  // TODO: Rename and change types of parameters  private var param1: String? = null  lateinit var binding: FragmentTwoBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  arguments?.let {  param1 = it.getString(ARG\_PARAM1)  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  // Inflate the layout for this fragment  binding = FragmentTwoBinding.inflate(layoutInflater, container, false)  initViews()  return binding.root  }  private fun initViews() {  -----------show data received through args------------  binding.txtName.text = param1  }  companion object {  @JvmStatic  fun newInstance(param1: String) =  TwoFragment().apply {  arguments = Bundle().apply {  putString(ARG\_PARAM1, param1)  }  }  }  }  \*\*\*MainActivity\*\*\*  -------Implement FragmentAListener in parent activity-------  class MainActivity : AppCompatActivity(), OneFragment.OnFragmentInteraction {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  initViews()  }  private fun initViews() {  supportFragmentManager  .beginTransaction()  .add(R.id.fragmentContainer1, OneFragment())  .commit()  }  ---------------send data to FragmentB through args---------  override fun onClick(name: String) {  supportFragmentManager  .beginTransaction()  .add(R.id.fragmentContainer2, TwoFragment.newInstance(name))  .commit()  }  }  \*\*\*activity\_main.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout 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"  android:orientation="vertical"  android:weightSum="2"  tools:context=".MainActivity">  <FrameLayout  android:id="@+id/fragmentContainer1"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_weight="1"  />  <FrameLayout  android:id="@+id/fragmentContainer2"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_weight="1"  />  </LinearLayout>  \*\*\*fragment\_one.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="@color/teal\_200"  tools:context=".OneFragment">  <EditText  android:id="@+id/editName"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:hint="Enter Name"  />  <Button  android:id="@+id/btnSend"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:text="Send Data"  />  </LinearLayout>  \*\*\*fragment\_two.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"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@color/purple\_500"  tools:context=".TwoFragment">  <TextView  android:id="@+id/txtName"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_centerInParent="true"  android:textSize="40sp"  android:text="Hello" />  </RelativeLayout> | </end> |
| <hitle> | PassDataFragmentsOpenWebsite | <chare> | 1 | <pext> | 02-15/PassDataFragmentsOpenWebsite  --------define communicator interface------------  interface WebsiteCommunicator {  fun setAddress(address:String)  }  \*\*\* RecycleViewFragment\*\*\*  --------setup data source fragment to send----------  class RecycleViewFragment : Fragment() {  private lateinit var binding: FragmentRecycleViewFragementBinding  private lateinit var websiteAdapter: WebsiteAdapter  private val website = ArrayList<Website>()  private lateinit var communicator: WebsiteCommunicator  override fun onAttach(context: Context) {  super.onAttach(context)  ----------save parent activity as communicator-------------  communicator = context as WebsiteCommunicator  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  binding = FragmentRecycleViewFragementBinding.inflate(inflater, container, false)  initView()  return binding.root  }  private fun initView(){  website.apply {  add(Website("https://www.google.com"))  add(Website("https://www.facebook.com"))  add(Website("https://www.yahoo.com"))  add(Website("https://www.mavs.com"))  add(Website("https://www.bilibili.com"))  add(Website("https://www.icloud.com"))  add(Website("https://www.adorama.com"))  add(Website("https://www.microcenter.com"))  add(Website("https://www.kotlinlang.org"))  }  websiteAdapter = WebsiteAdapter(communicator, website)  val layoutManager = LinearLayoutManager(context, RecyclerView.VERTICAL, false)  binding.ryvWebsite.layoutManager = layoutManager  binding.ryvWebsite.adapter = websiteAdapter  }  }  \*\*\*Data and Adapter for RecyclerView\*\*\*  data class Website(val address: String){  }  class WebsiteAdapter (private val communicator: WebsiteCommunicator, private val website:ArrayList<Website>):RecyclerView.Adapter<WebsiteAdapter.WebsiteViewHolder>(){  private lateinit var bindingWebsite: ItemViewBinding  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): WebsiteViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  bindingWebsite = ItemViewBinding.inflate(layoutInflater, parent, false)  return WebsiteViewHolder(bindingWebsite.root)  }  override fun onBindViewHolder(holder: WebsiteViewHolder, position: Int) {  holder.bind(website[position])  }  override fun getItemCount()= website.size  inner class WebsiteViewHolder(view: View) :RecyclerView.ViewHolder(view) {  fun bind(website: Website){  bindingWebsite.txtWebsite.text = website.address  bindingWebsite.txtWebsite.setOnClickListener {  ---------------send data through communicator-------------  communicator.setAddress(website.address)  }  }  }  }  \*\*\*MainActivity\*\*\*  ------------Implement Communicator in parent Activity--------------  class MainActivity : AppCompatActivity(), WebsiteCommunicator {  private lateinit var websiteFragment:WebViewFragment  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  addFragments()  }  private fun addFragments(){  supportFragmentManager  .beginTransaction()  .replace(R.id.recycleFragment, RecycleViewFragment())  .commit()  websiteFragment = WebViewFragment()  supportFragmentManager  .beginTransaction()  .replace(R.id.websiteFragment, websiteFragment)  .commit()  }  ----------override method where receive data----------------  override fun setAddress(address:String){  ----------send data to target fragment----------------  websiteFragment.addressReceiver(address)  Toast.makeText(this, address, Toast.LENGTH\_SHORT).show()  }  }  \*\*\*WebViewFragment\*\*\*  class WebViewFragment : Fragment() {  private lateinit var websitePage: WebView  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  return inflater.inflate(R.layout.fragment\_web\_view\_fragement, container, false)  }  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  websitePage = view.findViewById(R.id.webPage)  }  -----------final method where receive data from parent activity-----------  fun addressReceiver(address:String){  websitePage.apply {  webViewClient = WebViewClient()  websitePage.loadUrl(address)  settings.javaScriptEnabled = true  settings.setSupportZoom(true)  }  }  }  \*\*\*activity\_main.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout 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"  android:orientation="vertical"  android:weightSum="2"  tools:context=".MainActivity">  <FrameLayout  android:id="@+id/websiteFragment"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:layout\_weight="1"  />  <FrameLayout  android:id="@+id/recycleFragment"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:layout\_weight="1"/>  </LinearLayout>  \*\*\*fragment\_recycle\_view\_fragment.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"  xmlns:app="http://schemas.android.com/apk/res-auto"  tools:context=".RecycleViewFragment"  android:background="#F9AAAD">  <androidx.recyclerview.widget.RecyclerView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/ryvWebsite"  tools:itemCount="3"  tools:listitem="@layout/item\_view"/>  </LinearLayout>  <?xml version="1.0" encoding="utf-8"?>  \*\*\*fragment\_web\_view\_fragment.xml\*\*\*  <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"  android:background="@color/black"  tools:context=".WebViewFragment">  <WebView  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:id="@+id/webPage"  />  </FrameLayout>  \*\*\*item\_view.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:background="@color/teal\_200"  android:orientation="vertical">  <TextView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/txtWebsite"  android:text="Email Address"  android:gravity="center"  android:textStyle="bold"  android:textColorHint="@color/black"  android:textSize="30sp"  android:clickable="true"  android:focusable="true"/>  </LinearLayout> | </end> |
| <hitle> | RetrofitDemo2 LoggingInterceptor | <chare> | 1 | <pext> | 02-15/RetrofitDemo2 LoggingInterceptor  \*\*\*ApiService\*\*\*  interface ApiService {  @GET(END\_POINT)  fun getWeather(  @Query("q") city: String,  @Query("appid") appId: String  ): Call<WeatherResponse>  ------------replace Query by QueryMap-------------  @GET(END\_POINT)  fun getWeatherUsingQueryMap(  @QueryMap params: HashMap<String, String>  ): Call<WeatherResponse>  -----------Path control ----------  @GET("weather/{serverId}/filter")  fun getWeatherUsingPathAndQueryMap(  @Path("serverId") id:Int,  @Query("q") city: String,  @Query("appid") appId: String  ): Call<WeatherResponse>  }  object Constant {  const val BASE\_URL = "https://openweathermap.org/data/2.5/"  const val END\_POINT = "weather"  //const val API\_KEY = "3f9bb0b98f7d19d210e8fb480fc1a8a2"  const val API\_KEY = "f454f29fc2f260a06f2475cb2e377299"  }  data class WeatherResponse(  val id: String,  val main: String,  )  \*\*\*RetrofitBuilder\*\*\*  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  ---------- HttpLoggingInterceptor--------------  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  ------------OkHttpClient--------------  val client = OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  -----------retrofit with client/interceptor---------  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  -----------call normal Query api----------  private fun fetchWeatherData() {  apiService.getWeather(  "London",  API\_KEY  ).enqueue(object : Callback<WeatherResponse> {  override fun onResponse(  call: Call<WeatherResponse>,  response: Response<WeatherResponse>  ) {  binding.txtData.text = response.body()?.main?.toString()  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<WeatherResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  -----------call QueryMap api----------  private fun fetchWeatherDataUsingQueryMap() {  val params = HashMap<String, String>()  params["q"] = "London"  params["appid"] = API\_KEY  apiService.getWeatherUsingQueryMap(  params  ).enqueue(object : Callback<WeatherResponse> {  override fun onResponse(  call: Call<WeatherResponse>,  response: Response<WeatherResponse>  ) {  binding.txtData.text = response.body()?.main?.toString()  }  override fun onFailure(call: Call<WeatherResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  -----------call QueryMap/Path api----------  private fun fetchWeatherDetaUsingPathAndQueryMap() {  apiService.getWeatherUsingPathAndQueryMap(  1020,  "Paris",  API\_KEY  ).enqueue(object: Callback<WeatherResponse> {  override fun onResponse(  call: Call<WeatherResponse>,  response: Response<WeatherResponse>  ) {  binding.txtData.text = response.body()?.main?.toString()  }  override fun onFailure(call: Call<WeatherResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  } | </end> |
| <hitle> | RetrofitDemo3 Post Comment | <chare> | 1 | <pext> | 02-15/RetrofitDemo3 Post Comment  \*\*\*ApiService\*\*\*  interface ApiService {  -----------post endpoint ----------  @FormUrlEncoded  @POST(END\_POINT)  fun postComment(  @Field("title") title: String,  @Field("body") body: String,  @Field("userId") userId: String  ): Call<CommentResponse>  }  object Constant {  const val BASE\_URL = "https://jsonplaceholder.typicode.com/"  const val END\_POINT = "posts"  }  data class Comment (  val title: String,  val body: String,  val userId: String,  )  data class CommentResponse(  val body: String,  val id: Int,  val title: String,  val userId: Int  )  \*\*\*RetrofitBuilder\*\*\*  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  lateinit var apiService: ApiService  lateinit var retrofit: Retrofit  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  binding.btnCallApi.setOnClickListener {  postComment()  binding.loader.visibility = View.VISIBLE  }  }  --------------call post comment api---------------  private fun postComment() {  apiService.postComment(  "London",  "good city",  "1"  ).enqueue(object : Callback<CommentResponse> {  override fun onResponse(  call: Call<CommentResponse>,  response: Response<CommentResponse>  ) {  binding.txtData.text = response.body()?.title  }  override fun onFailure(call: Call<CommentResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  } | </end> |
| <hitle> | NewsApp Retrofit AuthInterceptor | <chare> | 1 | <pext> | 02-16/NewsApp Retrofit AuthInterceptor  interface ApiService {  @GET(END\_POINT)  fun getLatestNews(): Call<NewsResponse>  }  object Constant {  const val BASE\_URL = "https://api.currentsapi.services/v1/"  const val END\_POINT = "latest-news"  const val AUTHORIZATION = "Authorization"  const val TOKEN = "ehTUs\_L7VNOevxSsW301L3Y6KhOmJ573Grs-VKu--uPjKPZF"  }  data class New(  val author: String,  val category: List<String>,  val description: String,  val id: String,  val image: String,  val language: String,  val published: String,  val title: String,  val url: String  )  data class NewsResponse(  val news: List<New>,  val status: String  )  ------------define AuthInterceptor-----------  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  currentRequest.addHeader(AUTHORIZATION, TOKEN)  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  }  \*\*\*RetrofitBuilder\*\*\*  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  -------------add AuthInterceptor when build retrofit-----------  .addInterceptor(AuthInterceptor())  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  lateinit var apiService: ApiService  lateinit var retrofit: Retrofit  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  getLatestNews()  }  private fun getLatestNews() {  apiService.getLatestNews()  .enqueue(object: Callback<NewsResponse> {  override fun onResponse(  call: Call<NewsResponse>,  response: Response<NewsResponse>  ) {  binding.txtMessage.text = response.body()?.status!!  }  override fun onFailure(call: Call<NewsResponse>, t: Throwable) {  binding.txtMessage.text = t.message.toString()  }  })  }  }  \*\*\*Reference code  ---------all Interceptors-----------  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  --------------RetryInterceptor-------------  class RetryInterceptor(private val retryAttempts: Int) : Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  for (i in 1..retryAttempts) {  try {  return chain.proceed(chain.request())  } catch (e: SocketTimeoutException) {  e.printStackTrace()  }  }  throw RuntimeException("failed to compile the request")  }  }  -------------APIKeyInterceptor--------------  class APIKeyInterceptor : Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentUrl = chain.request().url  val newUrl = currentUrl.newBuilder().addQueryParameter("api\_key", "api\_key").build()  val currentRequest = chain.request().newBuilder()  val newRequest = currentRequest.url(newUrl).build()  return chain.proceed(newRequest)  }  }  ------------------- CacheInterceptor---------------  class CacheInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val originalResponse = chain.proceed(chain.request())  return if (Utils.isNetworkAvailable(applicationContext)) {  val maxAge = 60  originalResponse.newBuilder()  .addHeader("Cache-control", "public, max-age = $maxAge")  .build()  } else {  val maxStale = 60 \* 60 \* 24 \* 28 // 4 weeks  originalResponse.newBuilder()  .addHeader("Cache-control", "public, only-if-cache max-age = $maxStale")  .build()  }  }  }  -------------Header via Interceptor(AuthInterceptor)---------  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  currentRequest.addHeader(AUTHORIZATION, TOKEN)  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  } | </end> |
| <hitle> | PostApiRetrofitDemo All Endpoint types | <chare> | 1 | <pext> | 02-16/PostApiRetrofitDemo All Endpoint types  interface ApiService {  ----------all endpoint types-----------  @FormUrlEncoded  @POST(END\_POINT)  fun postPosts(  @Field("title") title: String,  @Field("body") body: String,  @Field("userId") userId: String  ): Call<PostsResponse>  @GET("posts/{postId}")  fun getPosts(  @Path("postId") postId: String,  ): Call<PostsResponse>  @GET(END\_POINT)  fun getAllPosts(): Call<PostAllData>  @Headers("Content-type: application/json")  @POST(END\_POINT)  fun addCourse(@Body postRequest: PostRequest): Call<PostsResponse>  @FormUrlEncoded  @PUT("posts/{postId}")  fun updatePosts(  @Path("postId") postId: String,  @Field("id") id: String,  @Field("title") title: String,  @Field("body") body: String,  @Field("userId") userId: String  ): Call<PostsResponse>  @FormUrlEncoded  @PATCH("posts/{postId}")  fun updatePostTitle(  @Path("postId") postId: String,  @Field("title") title: String,  ): Call<PostsResponse>  @PATCH("posts/{postId}")  fun deletePost(  @Path("postId") postId: String  ): Call<Unit>  }  object Constant {  const val BASE\_URL = "https://jsonplaceholder.typicode.com/"  const val END\_POINT = "posts"  }  data class AllPosts(  val listOfPosts: List<PostsResponse>  )  class PostAllData : ArrayList<PostAllDataItem>()  data class PostAllDataItem(  val body: String,  val id: Int,  val title: String,  val userId: Int  )  data class PostRequest(  val body: String,  val id: Int,  val title: String,  val userId: Int  )  data class PostsResponse(  val id: Int,  val title: String,  val body: String,  val userId: Int  )  --------RetrofitBuilder------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  ------------MainActivity------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  lateinit var apiService: ApiService  lateinit var retrofit: Retrofit  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  binding.btnCallApi.setOnClickListener {  postPosts()  updatePosts()  updatePostTitle()  deletePost()  getAPost()  getAllPosts()  binding.loader.visibility = View.VISIBLE  }  }  private fun getAllPosts() {  apiService.getAllPosts().enqueue(object : Callback<PostAllData> {  override fun onResponse(  call: Call<PostAllData>,  response: Response<PostAllData>  ) {  binding.txtData.text = response.body()?.toString()  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<PostAllData>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  private fun getAPost() {  apiService.getPosts(  "1",  ).enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  binding.txtData.text = response.body()?.title  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  private fun deletePost() {  apiService.deletePost(  "1",  ).enqueue(object : Callback<Unit> {  override fun onResponse(  call: Call<Unit>,  response: Response<Unit>  ) {  binding.txtData.text = response.message().toString()  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<Unit>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  private fun updatePostTitle() {  apiService.updatePostTitle(  "1",  "London",  ).enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  binding.txtData.text = response.body()?.title  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  private fun updatePosts() {  apiService.updatePosts(  "1",  "1",  "London",  "good city",  "1"  ).enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  binding.txtData.text = response.body()?.title  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  private fun postPosts() {  apiService.postPosts(  "London",  "good city",  "1"  ).enqueue(object : Callback<PostsResponse> {  override fun onResponse(  call: Call<PostsResponse>,  response: Response<PostsResponse>  ) {  binding.txtData.text = response.body()?.title  Toast.makeText(this@MainActivity, "Call Api Success!", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<PostsResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Call Api Failure", Toast.LENGTH\_SHORT).show()  }  })  }  } | </end> |
| <hitle> | RetrofitDemo6 ImageUpload | <chare> | 1 | <pext> | 02-17/RetrofitDemo6 ImageUpload  -------------ApiService--------------  interface ApiService {  @Multipart  @POST("upload\_image.php")  fun uploadPhoto(  @Part city: MultipartBody.Part,  @Part fileName: MultipartBody.Part,  @Part photo: MultipartBody.Part  ): Call<UploadResponse>  @Multipart  @POST  fun uploadPhotoUsingMultiPartMap(  @PartMap params: HashMap<String, MultipartBody.Part>  ): Call<UploadResponse>  @Multipart  @POST("uploads/binary")  fun uploadTry(  @Part accountId: MultipartBody.Part,  @Part filePath: MultipartBody.Part,  @Part fileUrl: MultipartBody.Part  ): Call<UploadResponse>  }  object Constants {  const val BASE\_URL = "https://psmobitech.com/fileupload/"  const val BASE\_URL2 = "https://api.upload.io/v2/accounts/12a1xyW/"  }  data class UploadResponse(  @SerializedName("status") val status: Int,  @SerializedName("message") val message: String  )  ------------AuthInterceptor-------------  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  // currentRequest.addHeader("Authorization", "Bearer public\_12a1xyW9SwYkiazq2wzWyLyuwXTT")  currentRequest.addHeader("Authorization", "Bearer public\_FW25bEpFAw7ssyEQt13fsS1ZCvyF")  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  }  -------------Retrofit by ApiClient---------  object ApiClient {  private lateinit var myRetrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::myRetrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor()  loggingInterceptor.level = HttpLoggingInterceptor.Level.BODY  val client = OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .addInterceptor(AuthInterceptor())  .callTimeout(60, TimeUnit.SECONDS)  .build()  myRetrofit = Retrofit.Builder()  .baseUrl(Constants.BASE\_URL2)  .client(client)  .addConverterFactory(GsonConverterFactory.create())  .build()  }  return myRetrofit  }  val apiService: ApiService by lazy {  getRetrofit().create(ApiService::class.java)  }  }  \*\*\*MainActivity\*\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private val SELECT\_PHOTO = 100  private var selectImageUrl: Uri? = null  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpEvents()  }  private fun setUpEvents() {  binding.apply {  btnSelectPhoto.setOnClickListener {  selectPhoto()  }  btnUploadPhoto.setOnClickListener {  // uploadPhoto()  uploadPhoto2()  }  }  }  private fun selectPhoto() {  -------------select photo from gallery-------  val pickPhotoIntent = Intent(ACTION\_PICK)  pickPhotoIntent.type = MediaStore.Images.Media.CONTENT\_TYPE  startActivityForResult(pickPhotoIntent, SELECT\_PHOTO)  }  override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  super.onActivityResult(requestCode, resultCode, data)  ------------process selected photo--------  if (resultCode == RESULT\_OK && requestCode == SELECT\_PHOTO) {  selectImageUrl = data?.data  binding.imgPhoto.setImageURI(selectImageUrl)  }  }  private fun uploadPhoto() {  val city = binding.city.text.toString()  val cityPart = MultipartBody.Part.createFormData("city", city)  val photoTitle = binding.title.text.toString()  val fileNamePart = MultipartBody.Part.createFormData("file\_name", photoTitle)  if (selectImageUrl == null) {  Toast.makeText(this, "Please select Photo", Toast.LENGTH\_SHORT).show()  return  }  selectImageUrl?.let {  val file = File(it.path)  val inputStream = contentResolver.openInputStream(it)  val totalSize = inputStream?.available() ?: 0  val buffer = ByteArray(totalSize)  inputStream?.read()  val body: RequestBody = RequestBody.create("image/\*".toMediaTypeOrNull(), buffer)  val photo = MultipartBody.Part.createFormData("photo", file.name, body)  val progress = ProgressDialog(this)  progress.setMessage("Uploading image...")  progress.setCancelable(false)  val call: Call<UploadResponse> =  ApiClient.apiService.uploadPhoto(cityPart, fileNamePart, photo)  call.enqueue(object : Callback<UploadResponse> {  override fun onResponse(  call: Call<UploadResponse>,  response: Response<UploadResponse>  ) {  progress.dismiss()  Toast.makeText(this@MainActivity, "Success", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<UploadResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Failure", Toast.LENGTH\_SHORT).show()  }  })  }  }  --------------upload photo----------  private fun uploadPhoto2() {  val cityPart = MultipartBody.Part.createFormData("accountId", "12a1xyW")  val fileNamePart = MultipartBody.Part.createFormData("filePath", "/uploads/file.png")  if (selectImageUrl == null) {  Toast.makeText(this, "Please select Photo", Toast.LENGTH\_SHORT).show()  return  }  selectImageUrl?.let {  val file = File(it.path)  val inputStream = contentResolver.openInputStream(it)  val totalSize = inputStream?.available() ?: 0  val buffer = ByteArray(totalSize)  inputStream?.read()  val body: RequestBody = RequestBody.create("image/\*".toMediaTypeOrNull(), buffer)  val photo = MultipartBody.Part.createFormData("fileUrl", file.name, body)  val progress = ProgressDialog(this)  progress.setMessage("Uploading image...")  progress.setCancelable(false)  val call: Call<UploadResponse> =  ApiClient.apiService.uploadTry(cityPart, fileNamePart, photo)  call.enqueue(object : Callback<UploadResponse> {  override fun onResponse(  call: Call<UploadResponse>,  response: Response<UploadResponse>  ) {  progress.dismiss()  Toast.makeText(this@MainActivity, "Success", Toast.LENGTH\_SHORT).show()  }  override fun onFailure(call: Call<UploadResponse>, t: Throwable) {  Toast.makeText(this@MainActivity, "Failure", Toast.LENGTH\_SHORT).show()  }  })  }  }  }  \*\*\*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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:layout\_margin="15dp"  android:background="#F6F2F2"  tools:context=".MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/city"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="@string/enter\_city"  android:textColor="@color/purple\_700"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/title"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="@string/enter\_title"  android:textColor="@color/purple\_700"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintTop\_toBottomOf="@+id/city" />  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/img\_photo"  android:layout\_width="0dp"  android:layout\_height="0dp"  app:layout\_constraintBottom\_toTopOf="@+id/btn\_select\_photo"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/title"  app:srcCompat="@drawable/ic\_launcher\_background" />  <Button  android:id="@+id/btn\_select\_photo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/select\_photo"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent" />  <Button  android:id="@+id/btn\_upload\_photo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/upload\_photo"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout>  \*\*\*AndroidManifest\*\*\*  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />  <uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />  \*\*\*Gradle\*\*\*  //Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:4.9.3'  //glide  implementation 'com.github.bumptech.glide:glide:4.13.0'  annotationProcessor 'com.github.bumptech.glide:compiler:4.13.0' | </end> |
| <hitle> | RoomDatabaseDemo1 | <chare> | 1 | <pext> | 02-17/RoomDatabaseDemo1  -----------create Table/Data Entity--------  \*\*\*database/User\*\*\*  @Entity(tableName = "MyUser")  data class User (  @PrimaryKey(autoGenerate = true) val id: Long,  @ColumnInfo(name = "name") val name: String,  @ColumnInfo(name = "email") val email: String,  @ColumnInfo(name = "age") val age: Int? = 0  )  -----------create Dao-----------  \*\*\*database/Dao\*\*\*  @Dao  interface UserDao {  @Insert  fun insertUser(user: User)  @Query("SELECT \* FROM MyUser")  fun getAllUsers(): List<User>  @Delete  fun deleteUser(user: User)  @Update  fun updateUser(user: User)  }  --------create Database------------  \*\*\*database/AppDatabase\*\*\*  @Database(entities = [User::class], version = 1, exportSchema = false)  abstract class AppDatabase: RoomDatabase() {  abstract fun userDao(): UserDao  companion object {  private var appDatabase: AppDatabase? = null  fun getDatabaseInstance(context: Context): AppDatabase {  if(appDatabase == null) {  appDatabase = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  ---------Here, database name-----------  "UserDB"  )  .allowMainThreadQueries()  .build()  }  return appDatabase as AppDatabase  }  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var database: AppDatabase  private lateinit var userDao: UserDao  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initDatabase()  initViews()  }  ---------Init Room Database-------  private fun initDatabase() {  database = AppDatabase.getDatabaseInstance(applicationContext)  userDao = database.userDao()  }  ---------operate DB by Room Dao--------  private fun initViews() {  binding.btnInsert.setOnClickListener {  userDao.insertUser(User(0, "Thomas", "test@gmail.com", 33))  }  binding.btnFetch.setOnClickListener {  val user = userDao.getAllUsers()[0].name  Toast.makeText(this@MainActivity, user, Toast.LENGTH\_SHORT).show()  }  binding.btnUpdate.setOnClickListener {  userDao.updateUser(User(1, "Hello", "update@gmail.com"))  }  binding.btnDelete.setOnClickListener {  userDao.deleteUser(User(1, "Hello", "update@gmail.com", 0))  }  }  }  \*\*\*Gradle\*\*\*  Plugins: id 'kotlin-kapt'  def room\_version = "2.5.0"  // To use Kotlin annotation processing tool (kapt)  kapt "androidx.room:room-compiler:$room\_version"  // optional - RxJava2 support for Room  implementation "androidx.room:room-rxjava2:$room\_version"  // optional - RxJava3 support for Room  implementation "androidx.room:room-rxjava3:$room\_version"  // optional - Guava support for Room, including Optional and ListenableFuture  implementation "androidx.room:room-guava:$room\_version"  // optional - Test helpers  testImplementation "androidx.room:room-testing:$room\_version"  // optional - Paging 3 Integration  implementation "androidx.room:room-paging:$room\_version" | </end> |
| <hitle> | Git/NotesAndTodoApp | <chare> | 1 | <pext> | 02-17/Git/NotesAndTodoApp  https://github.com/cheetahmail007/Notes-and-Todo-app | </end> |
| <hitle> | RoomDatabaseDemo1 Embedded | <chare> | 1 | <pext> | 02-21/RoomDatabaseDemo1 Embedded  -----------create Table/Data Entity--------  @Entity(tableName = "MyUser")  data class User (  @PrimaryKey(autoGenerate = true) val id: Long,  -----------embed object field---------  @Embedded val name: Name,  @ColumnInfo(name = "email") val email: String,  @ColumnInfo(name = "age") val age: Int? = 0,  @Embedded val address: Address? = null  )  data class Address(  val houseNumber: String?,  val street: String?,  val area: String?,  val city: String,  val state: String,  val country: String,  val zipCode: String  )  data class Name (  val firstName: String,  val middleName: String,  val lastName: String  )  -----------Create Dao----------  @Dao  interface UserDao {  @Insert  fun insertUser(user: User)  @Query("SELECT \* FROM MyUser")  fun getAllUsers(): List<User>  @Delete  fun deleteUser(user: User)  @Update  fun updateUser(user: User)  }  ------------Create Database-------------  @Database(entities = [User::class], version = 1, exportSchema = false)  abstract class AppDatabase: RoomDatabase() {  abstract fun userDao(): UserDao  companion object {  private var appDatabase: AppDatabase? = null  fun getDatabaseInstance(context: Context): AppDatabase {  if(appDatabase == null) {  appDatabase = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  ------------Here’s database name---------  "UserDB"  )  .allowMainThreadQueries()  .build()  }  return appDatabase as AppDatabase  }  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var database: AppDatabase  private lateinit var userDao: UserDao  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initDatabase()  initViews()  }  ------------init database and get dao object------------  private fun initDatabase() {  database = AppDatabase.getDatabaseInstance(applicationContext)  userDao = database.userDao()  }  -----------perform database operations------------  private fun initViews() {  binding.btnInsert.setOnClickListener {  val homeAddress = Address(  houseNumber = "323",  street = "Twitter",  area = "west estern",  city = "London",  state = "Georgia",  country = "US",  zipCode = "30294"  )  userDao.insertUser(  User(  0,  name = Name("Thomas", "Santos", "Jones"),  "test@gmail.com",  33, address = homeAddress  )  )  }  binding.btnFetch.setOnClickListener {  val user = userDao.getAllUsers()[0].name.middleName  Toast.makeText(this@MainActivity, user, Toast.LENGTH\_SHORT).show()  }  binding.btnUpdate.setOnClickListener {  userDao.updateUser(User(1, name = Name("Hello", "my", "world"), "update@gmail.com"))  }  binding.btnDelete.setOnClickListener {  userDao.deleteUser(User(1, Name("Hello", "my", "world"), "update@gmail.com", 0))  }  }  }  \*\*\*Gradle\*\*\*  Plugins: id 'kotlin-kapt'  def room\_version = "2.5.0"  // To use Kotlin annotation processing tool (kapt)  kapt "androidx.room:room-compiler:$room\_version"  // optional - RxJava2 support for Room  implementation "androidx.room:room-rxjava2:$room\_version"  // optional - RxJava3 support for Room  implementation "androidx.room:room-rxjava3:$room\_version"  // optional - Guava support for Room, including Optional and ListenableFuture  implementation "androidx.room:room-guava:$room\_version"  // optional - Test helpers  testImplementation "androidx.room:room-testing:$room\_version"  // optional - Paging 3 Integration  implementation "androidx.room:room-paging:$room\_version" | </end> |
| <hitle> | RoomDatabaseDemo1 Migration | <chare> | 1 | <pext> | 02-21/RoomDatabaseDemo1 Migration  @Entity(tableName = "MyUser")  data class User (  @PrimaryKey(autoGenerate = true) val id: Long,  @Embedded val name: Name,  @ColumnInfo(name = "email") val email: String,  @ColumnInfo(name = "age") val age: Int? = 0,  @Embedded val address: Address? = null,  ----------add columns to the table------------  @ColumnInfo(name = "nickName") val nickName: String? = null,  @ColumnInfo(name = "phone") val phone: Int? = 0,  )  ----------create migration method values----------------  val MIGRATION\_1\_2 = object : Migration(1, 2) {  override fun migrate(database: SupportSQLiteDatabase) {  database.execSQL("ALTER TABLE MyUser ADD COLUMN nickName Text")  }  }  val MIGRATION\_2\_3 = object : Migration(2, 3) {  override fun migrate(database: SupportSQLiteDatabase) {  database.execSQL("ALTER TABLE MyUser ADD COLUMN phone Int")  }  }  -----------------specify new version number----------------  @Database(entities = [User::class], version = 3, exportSchema = false)  abstract class AppDatabase: RoomDatabase() {  abstract fun userDao(): UserDao  companion object {  private var appDatabase: AppDatabase? = null  fun getDatabaseInstance(context: Context): AppDatabase {  if(appDatabase == null) {  appDatabase = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "UserDB"  )  .allowMainThreadQueries()  --------------add migration method-------  .addMigrations(MIGRATION\_2\_3)  .build()  }  return appDatabase as AppDatabase  }  }  } | </end> |
| <hitle> | RoomDBTypeConverter | <chare> | 1 | <pext> | 02-21/RoomDBTypeConverter  -------Product table and PriceChange  @Entity(tableName = "product\_table")  data class Product (  @PrimaryKey val name: String,  var initialPrice: BigDecimal,  val price: BigDecimal  )  data class PriceChange (  val name: String,  val initialPrice: BigDecimal,  val price: BigDecimal,  val change: BigDecimal  )  -----------ProductDao--------  @Dao  interface ProductDao {  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun insertProduct(product: Product)  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun insertProducts(products: List<Product>)  @Query("Select \* FROM product\_table")  fun getAllProducts(): List<Product>  ----------SQL middle operation from product to pricechange  @Query("Select \*, price-initialPrice as change FROM product\_table")  fun getAllPriceChange(): List<PriceChange>  }  -------Room Type Converter--------  class Converter {  companion object {  @TypeConverter  @JvmStatic  fun fromBigDecimal(value: BigDecimal): String = value.toString()  @TypeConverter  @JvmStatic  fun toBigDecimal(value: String): BigDecimal = value.toBigDecimal()  }  }  ---------AppDatabase using TypeConverter-------  @Database(entities = [Product::class], version = 1, exportSchema = false)  @TypeConverters(Converter::class)  abstract class AppDatabase: RoomDatabase() {  abstract fun productDao(): ProductDao  companion object {  private var appDatabase: AppDatabase? = null  fun getDatabaseInstance(context: Context): AppDatabase {  if (appDatabase == null) {  appDatabase = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "ProductDB"  )  .allowMainThreadQueries()  .build()  }  return appDatabase as AppDatabase  }  }  }  --------MainActivity---------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var database: AppDatabase  private lateinit var productDao: ProductDao  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpDB()  insertData()  binding.apply {  binding.btnAdd.setOnClickListener { insertData() }  binding.btnFetch.setOnClickListener { fetchData() }  }  }  private fun setUpDB() {  database = AppDatabase.getDatabaseInstance(this.applicationContext)  productDao = database.productDao()  }  private fun insertData() {  val list = mutableListOf<Product>().apply {  add(  Product(  "Bitcoin1",  "0.000002323232323232".toBigDecimal(),  "0.000008474383834".toBigDecimal()  ))  add(Product(  "Bitcoin2",  "0.00000345345345".toBigDecimal(),  "0.00000948373748".toBigDecimal()  ))  add(Product(  "Bitcoin3",  "0.00000546138766".toBigDecimal(),  "0.000001654689123".toBigDecimal()  ))  add(Product(  "Bitcoin4",  "0.00000879424932468".toBigDecimal(),  "0.00000216534894684".toBigDecimal()  ))  add(Product(  "Bitcoin5",  "0.000002464564646".toBigDecimal(),  "0.000002164538435".toBigDecimal()  ))  }  productDao.insertProducts(list)  }  ------allPriceChange is generated from Product durin SQL-------  private fun fetchData() {  val pPriceChange = database.productDao().getAllPriceChange()  val first = pPriceChange[0]  val second = pPriceChange[1]  val third = pPriceChange[2]  val fourth = pPriceChange[3]  binding.apply {  firstRowName.text = first.name  firstRowInitPrice.text = first.initialPrice.toString()  firstRowPrice.text = first.change.toString()  secondRowName.text = second.name  secondRowInitPrice.text = second.initialPrice.toString()  secondRowPrice.text = second.change.toString()  thirdRowName.text = third.name  thirdRowInitPrice.text = third.initialPrice.toString()  thirdRowPrice.text = third.change.toString()  fourthRowName.text = fourth.name  fourthRowInitPrice.text = fourth.initialPrice.toString()  fourthRowPrice.text = fourth.change.toString()  }  }  } | </end> |
| <hitle> | MultipleTablesInRoomDB | <chare> | 1 | <pext> | 02-21/ MultipleTablesInRoomDB  -------parent is Customer table which will be associated with child-------  @Entity  data class Customer (  @PrimaryKey var customer\_id: Int,  @ColumnInfo(name = "customer\_name") var customerName: String,  @ColumnInfo(name = "customer\_is\_prime") var customerIsPrime: String,  @ColumnInfo(name = "customer\_mobile") var customerMobile: String,  @ColumnInfo(name = "customer\_email") var customerEmail: String,  )  -------child is Bill table which has parent fields connected--------  @Entity(  ------------customer\_id is foreign key----------  foreignKeys = [ForeignKey(  entity = Customer::class,  parentColumns = arrayOf("customer\_id"),  childColumns = arrayOf("bill\_id")  )], indices = [Index(value = ["bill\_id"])]  )  data class Bill(  @PrimaryKey(autoGenerate = true) var bill\_id: Int = 0,  @ColumnInfo(name = "amount") var amount:Int,  @ColumnInfo(name = "customer\_id") var customerId: Int,  )  -----------------CustomerDao---------  @Dao  interface CustomerDao {  @Insert  fun insertCustomer(customer: Customer)  @Insert  fun insertAllCustomers(customers: List<Customer>)  @Delete  fun deleteCustomer(customer: Customer)  @Update  fun updateCustomer(customer: Customer)  @Query("SELECT \* FROM Customer")  fun getAllCustomers(): List<Customer>  ------------we can use parameter in SQL like customer\_id----------  @Query("SELECT \* FROM Customer WHERE customer\_id IN (:customer\_id)")  fun getAllCustomers(customer\_id: Int): List<Customer>  }  ----------BillDao-----------  @Dao  interface BillDao {  @Insert  fun insertBill(bill: Bill)  @Insert  fun insertAllBills(bills: List<Bill>)  @Delete  fun deleteBill(bill: Bill)  @Update  fun updateBill(bill: Bill)  @Query("SELECT \* FROM Bill")  fun getAllBills(): List<Bill>  ----------------query from bill associating with Customer table--------  @Query("SELECT \* FROM Bill WHERE customer\_id IN (:customer\_id)")  fun getCustomerById(customer\_id: Int): List<Bill>  } | </end> |
| <hitle> | AnimationDemo1 | <chare> | 1 | <pext> | 02-21/AnimationDemo1  --------create res/anim/ animation resources-------  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android">  <rotate android:fromDegrees="0"  android:toDegrees="360"  android:pivotX="50%"  android:pivotY="50%"  android:duration="5000"  />  </set>  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android">  <translate android:toYDelta="0%"  android:fromYDelta="-300"  android:duration="20000" />  </set>  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android">  <rotate android:fromDegrees="0"  android:toDegrees="360"  android:pivotX="50%"  android:pivotY="50%"  android:duration="5000"  />  <translate android:toYDelta="0%"  android:fromYDelta="-300"  android:duration="20000" />  </set>  -------------apply the animation to the view--------  val animTest = AnimationUtils.loadAnimation(this, R.anim.rotate\_translate)  binding.imgSun.startAnimation(animTest) | </end> |
| <hitle> | ViewModel vs Normal | <chare> | 1 | <pext> | 02-23/MVVMDemo1  ----------------------Model------------------------  class Counter {  var counter: Int = 0  }  ----------------------ViewModel-----------------  class TestViewModel: ViewModel() {  val liveDataCounter2 = MutableLiveData<Counter>()  val model = Counter()  fun addCounter2() {  model.counter++  liveDataCounter2.po nmmmnm stValue(model)  }  }  -----------------------MainActivity------------------  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  lateinit var testViewModel: TestViewModel  var counter1 = Counter()  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  initViewModel()  setUpObservers()  }  private fun setUpObservers() {  testViewModel.liveDataCounter2.observe(this) {  binding.txtCounter2.text = it.counter.toString()  }  }  private fun initViewModel() {  testViewModel = ViewModelProvider(this)[TestViewModel::class.java]  }  private fun initViews() {  binding.btnCounter1.setOnClickListener {  counter1.counter++  binding.txtCounter1.text = counter1.counter.toString()  }  binding.btnCounter2.setOnClickListener {  testViewModel.addCounter2()  }  }  }  \*\*\*activity\_main.xml\*\*\*  txtCounter1, txtCounter2, btnCounter1, btnCounter2 | </end> |
| <hitle> | MVVMDemo2 DogApi Retrofit | <chare> | 1 | <pext> | 02-23/MVVMDemo2 DogApi walkthrough  ------------------model layer, prepare data, retrofit related components-------------------  object Constant {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  interface ApiService {  @GET(END\_POINT)  fun getRandomDog(): Call<DogResponse>  }  data class DogResponse(  val message: String,  val status: String  )  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .build()  }  return retrofit  }  }  ---------implement view model layer--------  -----------create custom viewmodel class extending ViewModel-------------------  class DogViewModel: ViewModel() {  --------create observable variable for view------------  val dogResponse = MutableLiveData<DogResponse>()  val error = MutableLiveData<String>()  private lateinit var apiService: ApiService  private lateinit var retrofit: Retrofit  fun getDogResponse() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  apiService.getRandomDog().enqueue(object :Callback<DogResponse>{  override fun onResponse(call: Call<DogResponse>, response: Response<DogResponse>) {  if(response.isSuccessful) {  ----------update observable value and it is automatically reflected in the view------  dogResponse.value = response.body()  }  }  override fun onFailure(call: Call<DogResponse>, t: Throwable) {  error.value = t.message  }  })  }  }  ---------------------MainActivity--------------------  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  lateinit var dogViewModel: DogViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  initViewModel()  setUpObservers()  }  ------------init viewmodel--------------  private fun initViewModel() {  dogViewModel = ViewModelProvider(this)[DogViewModel::class.java]  }  ----------setup observer for viewmodel’s observable value---------  private fun setUpObservers() {  dogViewModel.dogResponse.observe(this) {  Picasso.get().load(it.message).into(binding.imageView)  }  dogViewModel.error.observe(this) {  Toast.makeText(this, it, Toast.LENGTH\_SHORT).show()  }  }  private fun initViews() {  binding.button.setOnClickListener {  dogViewModel.getDogResponse()  }  }  }  \*\*\*activity\_main.xml\*\*\*  imageView, button | </end> |
| <hitle> | MVVMDemo3 SharedViewModel fragments openWebView from url list | <chare> | 1 | <pext> | 02-23/MVVMDemo3 SharedViewModel fragments openWebView from url list  -------------Model--------------------  data class Url (  val urlValue: String  )  --------------SharedViewModel------------  class SharedViewModel: ViewModel() {  val urlLiveData = MutableLiveData<Url>()  fun sendData(url: Url) {  urlLiveData.value = url  }  }  --------------UrlFragment--------------------  class UrlFragment : Fragment() {  lateinit var binding: FragmentUrlBinding  lateinit var viewModel: SharedViewModel  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  // Inflate the layout for this fragment  binding = FragmentUrlBinding.inflate(inflater, container, false)  return binding.root  }  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  viewModel = ViewModelProvider(requireActivity())[SharedViewModel::class.java]  binding.txtGoogle.setOnClickListener {  callViewModel(binding.txtGoogle.text.toString())  }  binding.txtFacebook.setOnClickListener {  callViewModel(binding.txtFacebook.text.toString())  }  binding.txtGithub.setOnClickListener {  callViewModel(binding.txtGithub.text.toString())  }  }  private fun callViewModel(url: String) {  viewModel.sendData(Url(url))  }  }  \*\*\*xml\*\*\*txtGoogle, txtFacebook, txtGithub  ----------------------WebViewFragment----------------------  class WebViewFragment : Fragment() {  lateinit var binding: FragmentWebViewBinding  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  // Inflate the layout for this fragment  binding = FragmentWebViewBinding.inflate(inflater, container, false)  return binding.root  }  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  val viewModel = ViewModelProvider(requireActivity())[SharedViewModel::class.java]  viewModel.urlLiveData.observe(viewLifecycleOwner) {  binding.webView.apply {  settings.javaScriptEnabled = true  webViewClient = WebViewClient()  loadUrl(it.urlValue)  }  }  }  }  \*\*\*xml\*\*\*WebView-webView  ----------------------MainActivity--------------------------------------------  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initFragments()  }  private fun initFragments() {  supportFragmentManager.beginTransaction()  .add(R.id.fragment\_container\_url, UrlFragment())  .add(R.id.fragment\_container\_web\_view, WebViewFragment())  .commit()  }  }  \*\*\*Xml\*\*\*FrameLayout-fragment\_container\_web\_view  FrameLayout-fragment\_container\_url | </end> |
| <hitle> | NewsApp repository retrofit/room/viewmodel/mvvm | <chare> | 1 | <pext> | 02-24/MVVM NewsApp  model/local/AppDatabase, NewsDao  model/remote/data/News, NewsResponse  model/remote/ApiService, AuthIntercepter, Constant, RetrofitBuilder  model/repository/IRepository, LocalRepository, RemoteRepository, Repository  view/MainActivity, NewsRvAdapter  viewmodel/NewsViewModel, NewsViewModelFactory  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'kotlin-kapt'  id 'kotlin-parcelize'  }  android {  namespace 'com.example.newsapp'  compileSdk 33  defaultConfig {  applicationId "com.example.newsapp"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures {  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  //Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  //Convertor factory by Gson  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso  implementation 'com.squareup.picasso:picasso:2.71828'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  }  ----------------News------------------------  @Entity(tableName = "News")  data class News(  @ColumnInfo(name = "author")  @SerializedName("author")  val author: String,  @ColumnInfo(name = "description")  @SerializedName("description")  val description: String,  @PrimaryKey(autoGenerate = false)  @ColumnInfo(name = "id")  @SerializedName("id")  val id: String,  @ColumnInfo(name = "image")  @SerializedName("image")  val image: String,  @ColumnInfo(name = "language")  @SerializedName("language")  val language: String,  @ColumnInfo(name = "published")  @SerializedName("published")  val published: String,  @ColumnInfo(name = "title")  @SerializedName("title")  val title: String,  @ColumnInfo(name = "url")  @SerializedName("url")  val url: String  )  ---------------NewsResponse------------------------  data class NewsResponse(  @SerializedName("news")  val news: List<News>,  @SerializedName("status")  val status: String  )  ----------------ApiService------------------------------  interface ApiService {  @GET(END\_POINT)  fun getLatestNews(): Call<NewsResponse>  }  ----------------AuthIntercepter----------------------  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  currentRequest.addHeader(AUTHORIZATION, TOKEN)  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  }  -------------------Constant-------------------------  object Constant {  const val BASE\_URL = "https://api.currentsapi.services/v1/"  const val END\_POINT = "latest-news"  const val AUTHORIZATION = "Authorization"  const val TOKEN = "ehTUs\_L7VNOevxSsW301L3Y6KhOmJ573Grs-VKu--uPjKPZF"  }  -------------------RetrofitBuilder---------------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(AuthInterceptor())  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  -------------------AppDatabase----------------------------  @Database(entities = [News::class], version = 1, exportSchema = false)  abstract class AppDatabase : RoomDatabase() {  abstract fun getNewsDao(): NewsDao  companion object {  private var INSTANCE: AppDatabase? = null  fun getInstance(context: Context): AppDatabase {  if (INSTANCE == null) {  INSTANCE = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "newsDB"  ).allowMainThreadQueries().build()  }  return INSTANCE as AppDatabase  }  }  }  -----------------NewsDao----------------  @Dao  interface NewsDao {  @Insert(onConflict = OnConflictStrategy.IGNORE)  fun saveNews(news: List<News>): List<Long>  @Query("SELECT \* FROM News")  fun getNews(): LiveData<List<News>>  }  ------------------IRepository-----------------  interface IRepository {  fun getLatestNews(): LiveData<List<News>>  fun updateLatestNews()  val isProcessing: MutableLiveData<Boolean>  }  ----------------------LocalRepository-------------  class LocalRepository(private val appDatabase: AppDatabase) {  fun getLatestNews() = appDatabase.getNewsDao().getNews()  fun saveNews(news: List<News>) = appDatabase.getNewsDao().saveNews(news)  }  ----------------------RemoteRepository--------------  class RemoteRepository(private val apiService: ApiService) {  fun loadLatestNews() = apiService.getLatestNews()  }  ---------------------Repository-------------------------  class Repository(val localRepository: LocalRepository, private val remoteRepository: RemoteRepository): IRepository {  override val isProcessing = MutableLiveData<Boolean>()  override fun getLatestNews(): LiveData<List<News>> {  updateLatestNews()  return localRepository.getLatestNews()  }  override fun updateLatestNews() {  val call: Call<NewsResponse> = remoteRepository.loadLatestNews()  call.enqueue(object :Callback<NewsResponse>{  override fun onResponse(call: Call<NewsResponse>, response: Response<NewsResponse>) {  isProcessing.postValue(false)  if(!response.isSuccessful) {  return  }  val newsResponse: NewsResponse = response.body() ?: return  if(newsResponse.status == "ok") {  localRepository.saveNews(newsResponse.news)  }  }  override fun onFailure(call: Call<NewsResponse>, t: Throwable) {  isProcessing.postValue(false)  t.printStackTrace()  }  })  }  }  -------------------NewsViewModel-----------------------  class NewsViewModel(application: Application, val repository: IRepository): AndroidViewModel(application) {  val latestNews: LiveData<List<News>> = repository.getLatestNews()  val isProcessing: LiveData<Boolean> = repository.isProcessing  fun refreshNews() {  repository.updateLatestNews()  }  }  -----------------NewViewModelFactory---------------------  fun <T: ViewModel> T.createFactory(): ViewModelProvider.Factory {  val viewModel = this  return object: ViewModelProvider.Factory {  @Suppress("UNCHECKED\_CAST")  override fun <T : ViewModel> create(modelClass: Class<T>): T = viewModel as T  }  }  -----------------NewsRvAdapter-------------------  class NewsRvAdapter(private val context: Context, private val newsList: List<News>) :  RecyclerView.Adapter<NewsRvAdapter.NewsViewHolder>() {  private lateinit var binding: NewsItemBinding  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NewsViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NewsItemBinding.inflate(layoutInflater, parent, false)  return NewsViewHolder(binding.root)  }  override fun onBindViewHolder(holder: NewsViewHolder, position: Int) {  holder.apply {  val item = newsList[position]  item.apply {  newsTitle.text = title  //newsCategory.text = category?.get(0) ?: ""  newsAuthor.text = author  val url = newsList[position].image  Picasso.get().load(url).into(newsImg)  }  itemView.setOnClickListener {  /\* val intent = Intent(context, NewsDetailsActivity::class.java)  intent.putExtra("news",item)  context.startActivity(intent)\*/  }  }  }  override fun getItemCount(): Int {  return newsList.size  }  inner class NewsViewHolder(view: View) : RecyclerView.ViewHolder(view) {  val newsTitle = binding.txtTitle  val newsCategory = binding.txtCategory  val newsImg = binding.imgNews  val newsAuthor = binding.txtAuthor  }  }  -------------------------MainActivity-------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: NewsViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViewModel()  setUpObserver()  }  private fun setUpObserver() {  viewModel.isProcessing.observe(this) {  if(it) {  binding.progressBar.visibility = View.VISIBLE  } else {  binding.progressBar.visibility = View.GONE  }  }  viewModel.latestNews.observe(this) {  binding.rvNews.layoutManager = LinearLayoutManager(this)  binding.rvNews.adapter = NewsRvAdapter(this, it)  }  }  private fun initViewModel() {  val remoteRepository = RemoteRepository(RetrofitBuilder.getRetrofit().create(ApiService::class.java))  val localRepository = LocalRepository(AppDatabase.getInstance(this.applicationContext))  val repository = Repository(localRepository, remoteRepository)  val factory = NewsViewModel(application, repository).createFactory()  viewModel = ViewModelProvider(this, factory)[NewsViewModel::class.java]  }  }  \*\*\*bg.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item android:state\_pressed="false">  <shape android:shape="rectangle">  <corners android:radius="200dp" />  <solid android:color="@color/primary\_color" />  </shape>  </item>  <item android:state\_pressed="true">  <shape android:shape="rectangle">  <corners android:radius="200dp" />  <solid android:color="#251818" />  </shape>  </item>  </selector>  \*\*\*search\_bg.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <selector xmlns:android="http://schemas.android.com/apk/res/android">  <item android:state\_focused="false">  <shape android:shape="rectangle">  <corners android:radius="15dp" />  <solid android:color="#B8AEAE" />  </shape>  </item>  <item android:state\_focused="true">  <shape android:shape="rectangle">  <corners android:radius="15dp" />  <solid android:color="#B8AEAE" />  <stroke android:width="2dp" android:color="@color/white" android:dashWidth="1dp" android:dashGap="1dp" />  </shape>  </item>  </selector>  --------------------news\_item.xml--------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView 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="100dp"  android:layout\_margin="8dp"  android:elevation="15dp"  app:cardBackgroundColor="#E7E9EC"  app:cardCornerRadius="20dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgNews"  android:layout\_width="100dp"  android:layout\_height="100dp"  android:layout\_marginEnd="10dp"  android:scaleType="centerCrop"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtCategory"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="15dp"  android:background="@color/black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:textColor="@color/white"  android:textSize="16sp"  android:textStyle="italic"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toTopOf="parent"  tools:text="politics" />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="5dp"  android:maxLines="2"  android:textColor="@color/black"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtCategory"  tools:text="amazon" />  <TextView  android:id="@+id/txtAuthor"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="10dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  tools:text="Charles Ryan" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  \*\*\*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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="#EFE8E8"  android:padding="5dp"  tools:context=".view.MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/edtSearch"  android:layout\_width="0dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  android:layout\_marginEnd="100dp"  android:background="@drawable/search\_bg"  android:hint="@string/search\_here"  android:padding="10dp"  android:textColor="@color/black"  android:textColorHint="@color/primary\_color"  android:textSize="22sp"  app:layout\_constraintEnd\_toStartOf="@id/frame"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <FrameLayout  android:id="@+id/frame"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="5dp"  app:layout\_constraintBottom\_toBottomOf="@+id/edtSearch"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSearch"  android:layout\_width="wrap\_content"  android:layout\_height="50dp"  android:layout\_gravity="center"  android:background="@drawable/search\_bg" />  <androidx.appcompat.widget.AppCompatImageView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:elevation="5dp"  android:src="@drawable/ic\_baseline\_search\_24" />  </FrameLayout>  <ProgressBar  android:id="@+id/progressBar"  style="?android:attr/progressBarStyleLarge"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:max="3"  android:progress="100"  android:visibility="gone"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintLeft\_toLeftOf="parent"  app:layout\_constraintRight\_toRightOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  tools:visibility="visible" />  <HorizontalScrollView  android:id="@+id/svLayout"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="8dp"  android:layout\_marginTop="8dp"  android:scrollbars="horizontal"  android:visibility="gone"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/edtSearch">  <LinearLayout  android:id="@+id/lnrLayout"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="8dp"  android:orientation="horizontal"  app:layout\_constraintTop\_toBottomOf="@+id/edtSearch">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnGeneral"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:background="@drawable/bg"  android:paddingEnd="10dp"  android:text="General"  android:clickable="true"  android:focusable="true"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnPolitics"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Politics"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnEntertainment"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Entertainment"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnFood"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Food"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnAcademic"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Academic"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSports"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Sports"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBusiness"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Business"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnRegional"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Regional"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  </LinearLayout>  </HorizontalScrollView>  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:elevation="10dp"  android:padding="10dp"  app:cardBackgroundColor="#FFFFFF"  app:cardCornerRadius="30dp"  app:layout\_constraintTop\_toBottomOf="@id/svLayout">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvNews"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp"  app:layout\_constraintTop\_toBottomOf="@+id/svLayout" />  </androidx.cardview.widget.CardView>  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | Git/QuickNews | <chare> | 1 | <pext> | 02-24/ Git/QuickNews  https://github.com/cheetahmail007/QuickNews | </end> |
| <hitle> | NewsAppDataBinding | <chare> | 1 | <pext> | 02-27/NewsAppDataBinding  -----------------------Gradle.module-----------------------  android {    buildFeatures {  viewBinding true  dataBinding true  }  }  dependencies {  //Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  //Convertor factory by Gson  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso  implementation 'com.squareup.picasso:picasso:2.71828'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  }  -----------------------generate data binding from news\_item.xml-----------------------  On the root view we can use a menu to generate data binding layout template.  <?xml version="1.0" encoding="utf-8"?>  <layout 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">  <data>  <variable  name="news"  type="com.example.newsapp.model.remote.data.News" />  </data>  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="100dp"  android:layout\_margin="8dp"  android:elevation="15dp"  app:cardBackgroundColor="#E7E9EC"  app:cardCornerRadius="20dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgNews"  android:layout\_width="100dp"  android:layout\_height="100dp"  android:layout\_marginEnd="10dp"  android:scaleType="centerCrop"  remoteSourceImage = '@{news.image}'  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtCategory"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="15dp"  android:background="@color/black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:textColor="@color/white"  android:textSize="16sp"  android:textStyle="italic"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toTopOf="parent"  tools:text="politics" />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="5dp"  android:maxLines="2"  android:textColor="@color/black"  android:textSize="20sp"  android:textStyle="bold"  android:text='@{news.title}'  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtCategory"  tools:text="" />  <TextView  android:id="@+id/txtAuthor"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="10dp"  android:text='@{news.author}'  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  tools:text="" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  </layout>  -------------bind layout to view class NewsRvAdapter---------  \*\*\*view/NewsRvAdapter\*\*\*  class NewsRvAdapter(private val newsList: List<News>) :  RecyclerView.Adapter<NewsViewHolder>() {  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NewsViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  ----------generate binding instance by DataBindingUtil.inflate()-------  val binding: NewsItemBinding = DataBindingUtil.inflate(layoutInflater, R.layout.news\_item, parent, false)  return NewsViewHolder(binding)  }  override fun onBindViewHolder(holder: NewsViewHolder, position: Int) {  holder.setData(newsList[position])  }  override fun getItemCount() = newsList.size  }  \*\*\*view/NewsViewHolder\*\*\*  class NewsViewHolder(private val binding: NewsItemBinding): RecyclerView.ViewHolder(binding.root) {  fun setData(news: News) {  -------------set data to binding instance------  binding.news = news  }  }  --------create BindingAdapters-------  class CommonBindingAdapters {  companion object {  @JvmStatic  @BindingAdapter("remoteSourceImage")  ----------“remoteSourceImage” is responsible for attribute name in layout xml---------  fun loadImageFromServer(imageView: ImageView, url: String) = with(url) {  ----------The first parameter is component type------------  if(!isNullOrEmpty() && contains("https://")) {  Picasso.get()  .load(this)  .error(R.drawable.baseline\_error\_24)  .placeholder(R.drawable.ic\_launcher\_background)  .into(imageView)  }  }  }  }  --------prepare data models------------  \*\*\*model/remote/News\*\*\*  @Entity(tableName = "News")  data class News(  @ColumnInfo(name = "author")  @SerializedName("author")  val author: String,  @ColumnInfo(name = "description")  @SerializedName("description")  val description: String,  @PrimaryKey(autoGenerate = false)  @ColumnInfo(name = "id")  @SerializedName("id")  val id: String,  @ColumnInfo(name = "image")  @SerializedName("image")  val image: String,  @ColumnInfo(name = "language")  @SerializedName("language")  val language: String,  @ColumnInfo(name = "published")  @SerializedName("published")  val published: String,  @ColumnInfo(name = "title")  @SerializedName("title")  val title: String,  @ColumnInfo(name = "url")  @SerializedName("url")  val url: String  )  \*\*\*model/data/NewsResponse\*\*\*\*  data class NewsResponse(  @SerializedName("news")  val news: List<News>,  @SerializedName("status")  val status: String  )  ------------Room DB related components-------------------------  \*\*\*model/local/AppDatabase\*\*\*  @Database(entities = [News::class], version = 1, exportSchema = false)  abstract class AppDatabase : RoomDatabase() {  abstract fun getNewsDao(): NewsDao  companion object {  private var INSTANCE: AppDatabase? = null  fun getInstance(context: Context): AppDatabase {  if (INSTANCE == null) {  INSTANCE = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "newsDB"  ).allowMainThreadQueries().build()  }  return INSTANCE as AppDatabase  }  }  }  \*\*\*model/local/NewsDao\*\*\*  @Dao  interface NewsDao {  @Insert(onConflict = OnConflictStrategy.IGNORE)  fun saveNews(news: List<News>): List<Long>  @Query("SELECT \* FROM News")  fun getNews(): LiveData<List<News>>  }  -------------Retrofit related components-------------  \*\*\*model/remote/ApiService\*\*\*  interface ApiService {  @GET(END\_POINT)  fun getLatestNews(): Call<NewsResponse>  }  \*\*\* model/remote/AuthInterceptor\*\*\*  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  currentRequest.addHeader(AUTHORIZATION, TOKEN)  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  }  \*\*\* model/remote/Constant\*\*\*  object Constant {  const val BASE\_URL = "https://api.currentsapi.services/v1/"  const val END\_POINT = "latest-news"  const val AUTHORIZATION = "Authorization"  const val TOKEN = "ehTUs\_L7VNOevxSsW301L3Y6KhOmJ573Grs-VKu--uPjKPZF"  }  \*\*\*model/remote/RetrofitBuilder\*\*\*  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(AuthInterceptor())  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  ------------------Repository related components-----------  \*\*\*model/repository/IRepository\*\*\*  interface IRepository {  fun getLatestNews(): LiveData<List<News>>  fun updateLatestNews()  val isProcessing: MutableLiveData<Boolean>  }  \*\*\* model/repository/LocalRepository\*\*\*  class LocalRepository(private val appDatabase: AppDatabase) {  fun getLatestNews() = appDatabase.getNewsDao().getNews()  fun saveNews(news: List<News>) = appDatabase.getNewsDao().saveNews(news)  }  \*\*\* model/repository/RemoteRepository  class RemoteRepository(private val apiService: ApiService) {  fun loadLatestNews() = apiService.getLatestNews()  }  \*\*\* model/repository/Repository\*\*\*  class Repository(val localRepository: LocalRepository, private val remoteRepository: RemoteRepository): IRepository {  override val isProcessing = MutableLiveData<Boolean>()  override fun getLatestNews(): LiveData<List<News>> {  updateLatestNews()  return localRepository.getLatestNews()  }  override fun updateLatestNews() {  val call: Call<NewsResponse> = remoteRepository.loadLatestNews()  call.enqueue(object :Callback<NewsResponse>{  override fun onResponse(call: Call<NewsResponse>, response: Response<NewsResponse>) {  isProcessing.postValue(false)  if(!response.isSuccessful) {  return  }  val newsResponse: NewsResponse = response.body() ?: return  if(newsResponse.status == "ok") {  localRepository.saveNews(newsResponse.news)  }  }  override fun onFailure(call: Call<NewsResponse>, t: Throwable) {  isProcessing.postValue(false)  t.printStackTrace()  }  })  }  }  --------------NewsViewModel-----------------  \*\*\*viewmodel/NewsViewModel\*\*\*  class NewsViewModel(application: Application, val repository: IRepository): AndroidViewModel(application) {  val latestNews: LiveData<List<News>> = repository.getLatestNews()  val isProcessing: LiveData<Boolean> = repository.isProcessing  fun refreshNews() {  repository.updateLatestNews()  }  }  \*\*\*viewmodel/NewsViewModelFactory\*\*\*  fun <T: ViewModel> T.createFactory(): ViewModelProvider.Factory {  val viewModel = this  return object: ViewModelProvider.Factory {  @Suppress("UNCHECKED\_CAST")  override fun <T : ViewModel> create(modelClass: Class<T>): T = viewModel as T  }  }  -----------MainActivity-----------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: NewsViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViewModel()  setUpObserver()  }  private fun setUpObserver() {  viewModel.isProcessing.observe(this) {  if(it) {  binding.progressBar.visibility = View.VISIBLE  } else {  binding.progressBar.visibility = View.GONE  }  }  viewModel.latestNews.observe(this) {  binding.rvNews.layoutManager = LinearLayoutManager(this)  binding.rvNews.adapter = NewsRvAdapter(it)  }  }  private fun initViewModel() {  val remoteRepository = RemoteRepository(RetrofitBuilder.getRetrofit().create(ApiService::class.java))  val localRepository = LocalRepository(AppDatabase.getInstance(this.applicationContext))  val repository = Repository(localRepository, remoteRepository)  val factory = NewsViewModel(application, repository).createFactory()  viewModel = ViewModelProvider(this, factory)[NewsViewModel::class.java]  }  }  \*\*\*ativity\_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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="#EFE8E8"  android:padding="5dp"  tools:context=".view.MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/edtSearch"  android:layout\_width="0dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  android:layout\_marginEnd="100dp"  android:background="@drawable/search\_bg"  android:hint="@string/search\_here"  android:padding="10dp"  android:textColor="@color/black"  android:textColorHint="@color/primary\_color"  android:textSize="22sp"  app:layout\_constraintEnd\_toStartOf="@id/frame"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <FrameLayout  android:id="@+id/frame"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="5dp"  app:layout\_constraintBottom\_toBottomOf="@+id/edtSearch"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSearch"  android:layout\_width="wrap\_content"  android:layout\_height="50dp"  android:layout\_gravity="center"  android:background="@drawable/search\_bg" />  <androidx.appcompat.widget.AppCompatImageView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:elevation="5dp"  android:src="@drawable/ic\_baseline\_search\_24" />  </FrameLayout>  <ProgressBar  android:id="@+id/progressBar"  style="?android:attr/progressBarStyleLarge"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:max="3"  android:progress="100"  android:visibility="gone"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintLeft\_toLeftOf="parent"  app:layout\_constraintRight\_toRightOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  tools:visibility="visible" />  <HorizontalScrollView  android:id="@+id/svLayout"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="8dp"  android:layout\_marginTop="8dp"  android:scrollbars="horizontal"  android:visibility="gone"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/edtSearch">  <LinearLayout  android:id="@+id/lnrLayout"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="8dp"  android:orientation="horizontal"  app:layout\_constraintTop\_toBottomOf="@+id/edtSearch">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnGeneral"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:background="@drawable/bg"  android:paddingEnd="10dp"  android:text="General"  android:clickable="true"  android:focusable="true"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnPolitics"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Politics"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnEntertainment"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Entertainment"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnFood"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Food"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnAcademic"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Academic"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSports"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Sports"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBusiness"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Business"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnRegional"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Regional"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  </LinearLayout>  </HorizontalScrollView>  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:elevation="10dp"  android:padding="10dp"  app:cardBackgroundColor="#FFFFFF"  app:cardCornerRadius="30dp"  app:layout\_constraintTop\_toBottomOf="@id/svLayout">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvNews"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp"  app:layout\_constraintTop\_toBottomOf="@+id/svLayout" />  </androidx.cardview.widget.CardView>  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | TMDB-MOVIE-APP | <chare> | 1 | <pext> | 02-27/Git/TMDB-MOVIE-APP  https://github.com/cheetahmail007/TMDB-MOVIE-APP | </end> |
| <hitle> | MVVMTwoWayDataBinding | <chare> | 1 | <pext> | 02-27/MVVMTwoWayDataBinding  class MainViewModel: ViewModel() {  val firstNumber = MutableLiveData<String>()  val secondNumber = MutableLiveData<String>()  var result = MutableLiveData<Float>()  fun add() {  var a = 0.0f  firstNumber.value?.let{  if(it.isNotEmpty()) {  a = it.toFloat()  }  }  var b = 0.0f  secondNumber.value?.let {  if(it.isNotEmpty()) {  b = it.toFloat()  }  }  result.value = a + b  }  -------------float to String BindingAdapter---------  companion object {  @JvmStatic  @BindingAdapter("floatString")  fun floatToString(txtView: TextView, num: Float) {  txtView.text = num.toString()  }  }  }  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: MainViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  -------- data binding, important to setContentView by DataBindingUtil----------  binding = DataBindingUtil.setContentView<ActivityMainBinding?>(this, R.layout.activity\_main)  .apply {  -----------------two way databinding, important to assign lifecycleOwner to binding instance-----------  lifecycleOwner = this@MainActivity  }  viewModel = ViewModelProvider(this)[MainViewModel::class.java]  binding.viewModel = viewModel  setUpObservers()  }  private fun setUpObservers() {  viewModel.firstNumber.observe(this) {  viewModel.add()  }  viewModel.secondNumber.observe(this) {  viewModel.add()  }  }  }  -----------need to wrap the root view with layout tag in xml------------  <?xml version="1.0" encoding="utf-8"?>  <layout 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">  -----------------define data with <data> tag-------------  <data>  <variable  name="viewModel"  type="com.example.mvvmtwowaydatabinding.MainViewModel" />  </data>  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  tools:context=".MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/firstNumber"  ---------------two way binding expression-----------  android:text='@={viewModel.firstNumber}'  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp" />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/secondNumber"  android:text='@={viewModel.secondNumber}'  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp" />  <TextView  android:layout\_margin="10dp"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="16sp"  android:text="Sum of above"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtResult"  android:layout\_margin="10dp"  android:textSize="24sp"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  ---------------binding adapter in layout xml-----------  floatString='@{viewModel.result}'  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  </LinearLayout>  </layout> | </end> |
| <hitle> | FirebaseAuthDemo1-Email | <chare> | 1 | <pext> | 02-28/FirebaseAuthDemo1-Email  ----------Connnect project to Firebase-----------  Tools/Firebase/Authentication/Authentication using a custom authentication system (kotlin)  ----------Create FirebaseApp in Application------------  class MyApp:Application() {  override fun onCreate() {  super.onCreate()  FirebaseApp.initializeApp(this)  }  }  ----------Need to mention App class in Manifest------  <uses-permission android:name="android.permission.INTERNET" />  <application  android:name=".MyApp"  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  lateinit var firebaseAuth: FirebaseAuth  lateinit var firebaseUser: FirebaseUser  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initFirebase()  initViews()  }  ----------get instance of FirebaseAuth-------  private fun initFirebase() {  firebaseAuth = FirebaseAuth.getInstance()  if(firebaseAuth.currentUser != null) {  firebaseUser = firebaseAuth.currentUser!!  }  }  private fun initViews() {  binding.btnRegister.setOnClickListener {  registerUser()  }  binding.loginBtn.setOnClickListener {  loginUser()  }  }  private fun loginUser() {  val email = binding.usernameEdit.text.toString()  val password = binding.passwordEdit.text.toString()  ---------login/signIn by signInWithEmailAndPassword()-------  firebaseAuth.signInWithEmailAndPassword(email, password)  .addOnCompleteListener { task->  if(task.isSuccessful) {  showToast("User logged in successfully")  } else {  showToast("User logged in failed!!")  }  }  }  ---------register/signup by createUserWithEmailAndPassword()---------  private fun registerUser() {  val email = binding.usernameEdit.text.toString()  val password = binding.passwordEdit.text.toString()  firebaseAuth.createUserWithEmailAndPassword(email, password)  .addOnCompleteListener { task->  if(task.isSuccessful) {  showToast("User registered successfully")  if (firebaseAuth.currentUser != null) {  firebaseUser = firebaseAuth.currentUser!!  }  sendVerificationEmailLink()  } else {  showToast("User registration failed!!")  }  }  }  ----------on successful signup, send verification email link----------  private fun sendVerificationEmailLink() {  if(this::firebaseUser.isInitialized) {  firebaseUser.sendEmailVerification().addOnCompleteListener { task ->  if(task.isSuccessful) {  showToast("User verification email sent!!")  } else {  showToast("User verification email not sent!!")  }  }  }  }  private fun showToast(message: String) {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  }  }  \*\*\*activity\_main.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout 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"  xmlns:app="http://schemas.android.com/apk/res-auto">  <TextView  android:id="@+id/loginView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  android:layout\_marginStart="40dp"  android:layout\_marginTop="40dp"  android:layout\_marginEnd="40dp"  android:layout\_marginBottom="40dp"  android:text="Login"  android:textColor="@color/black"  android:textSize="40sp"  android:textStyle="bold"/>  <EditText  android:id="@+id/usernameEdit"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@+id/loginView"  android:layout\_below="@id/loginView"  android:layout\_marginLeft="60dp"  android:layout\_marginRight="60dp"  android:hint="Username"/>  <EditText  android:id="@+id/passwordEdit"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@id/usernameEdit"  android:layout\_marginLeft="60dp"  android:layout\_marginRight="60dp"  android:layout\_marginTop="20dp"  android:hint="Password"/>  <TextView  android:id="@+id/forgotPass"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@+id/passwordEdit"  app:layout\_constraintEnd\_toEndOf="@id/passwordEdit"  android:layout\_marginTop="10dp"  android:text="Forgot Password ?"  android:textColor="#FF0000"  android:textStyle="bold"/>  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@id/passwordEdit"  app:layout\_constraintStart\_toStartOf="@id/passwordEdit"  android:layout\_marginTop="10dp"  android:text="Remember me"  android:textColor="#676767"  android:textStyle="bold"/>  <Button  android:id="@+id/loginBtn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/forgotPass"  android:layout\_marginTop="30dp"  android:text="Login" />  <Button  android:id="@+id/btnRegister"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@id/loginBtn"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/forgotPass"  android:layout\_marginTop="30dp"  android:text="Register" />  <TextView  android:id="@+id/loginCenterView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toBottomOf="@id/loginBtn"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_marginTop="50dp"  android:text="Login"  android:textColor="@color/black"  android:textSize="25sp"  android:textStyle="bold" />  <TextView  android:id="@+id/text"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="5dp"  app:layout\_constraintTop\_toBottomOf="@id/loginCenterView"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  android:layout\_centerHorizontal="true"  android:text="With your social media account"  android:textColor="@color/black"  android:textSize="15sp"  android:textStyle="bold" />  <Button  android:id="@+id/facebookBtn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/text"  android:layout\_marginTop="30dp"  android:text="FACEBOOK" />  <Button  android:id="@+id/googleBtn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toEndOf="@id/facebookBtn"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/text"  android:layout\_marginTop="30dp"  android:text="Google+" />  <Button  android:id="@+id/twitterBtn"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintEnd\_toStartOf="@id/facebookBtn"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/text"  android:layout\_marginTop="30dp"  android:text="Twitter" />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | FirebaseAuthDemo1- Phone | <chare> | 1 | <pext> | 02-28/FirebaseAuthDemo1- Phone  Enable Phone service on Firebase  \*\*\*PhoneNumberActivity\*\*\*  class PhoneNumberActivity : AppCompatActivity() {  private lateinit var binding: ActivityPhoneNumberBinding  private lateinit var firebaseAuth: FirebaseAuth  private lateinit var firebaseUser: FirebaseUser  private lateinit var callback: PhoneAuthProvider.OnVerificationStateChangedCallbacks  private lateinit var resendToken: PhoneAuthProvider.ForceResendingToken  private lateinit var storedVerificationId: String  private var mobileNumber: String = ""  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityPhoneNumberBinding.inflate(layoutInflater)  setContentView(binding.root)  setUp()  sendOtpProcess()  }  private fun setUp() {  ----------- FirebaseAuth.getInstance()-----------  firebaseAuth = FirebaseAuth.getInstance()  ------------ PhoneAuthProvider.OnVerificationStateChangedCallbacks()--------  callback = object : PhoneAuthProvider.OnVerificationStateChangedCallbacks() {  override fun onVerificationCompleted(p0: PhoneAuthCredential) {  Log.i("tag", "onVerificationCompleted")  }  override fun onVerificationFailed(p0: FirebaseException) {  Log.i("tag", "onVerificationFailed")  }  -------------onCodeSent---------------  override fun onCodeSent(  verificationId: String,  token: PhoneAuthProvider.ForceResendingToken  ) {  super.onCodeSent(verificationId, token)  Log.i("tag", "onCodeSent")  storedVerificationId = verificationId  resendToken = token  showToast("OTP sent to your mobile, please enter")  binding.apply {  edPhone.text?.clear()  btnRegister.text = getString(R.string.register)  edPhone.hint = getString(R.string.enter\_otp)  }  }  }  }  private fun sendOtpProcess() {  binding.btnRegister.setOnClickListener {  if (binding.btnRegister.text == getString(R.string.register)) {  loginWithMobileNumber()  } else {  sendOtp()  }  }  }  private fun loginWithMobileNumber() {  val otp = binding.edPhone.text.toString()  if (otp.isNotEmpty()) {  val credentials: PhoneAuthCredential =  PhoneAuthProvider.getCredential(storedVerificationId, otp)  signInWithCredentials(credentials)  } else {  showToast("Enter valid OTP")  }  }  private fun signInWithCredentials(credentials: PhoneAuthCredential) {  --------------signInWithCredential() with complete listener--------------------  firebaseAuth.signInWithCredential(credentials)  .addOnCompleteListener { task ->  if (task.isSuccessful) {  showToast("Success login")  } else {  showToast("Failed login")  }  }  }  private fun sendOtp() {  mobileNumber = binding.edPhone.text.toString()  if (mobileNumber.isNotEmpty()) {  mobileNumber = "+86$mobileNumber"  sendVerificationCode()  } else {  showToast("Invalid mobile number")  }  }  private fun sendVerificationCode() {  -------------Build options for verifyPhoneNumber()--------  val options = PhoneAuthOptions.newBuilder(firebaseAuth)  .setPhoneNumber(mobileNumber)  .setTimeout(60L, TimeUnit.SECONDS)  .setActivity(this)  .setCallbacks(callback)  .build()  ----------------PhoneAuthProvider.verifyPhoneNumber()------------  PhoneAuthProvider.verifyPhoneNumber(options)  }  private fun showToast(message: String) {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  }  }  \*\*\*activity\_phone\_number.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="450dp"  android:layout\_gravity="center"  android:layout\_margin="10dp"  android:background="#F3FBFF"  android:padding="10dp">  <TextView  android:id="@+id/txtLogin"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:fontFamily="cursive"  android:gravity="center"  android:text="@string/register\_here\_using\_mobile"  android:textColor="@color/black"  android:textSize="32sp"  android:textStyle="bold"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/edPhone"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:fontFamily="monospace"  android:hint="@string/enter\_mobile"  android:textColor="#130000"  android:textColorHint="#670404"  android:textSize="23sp"  app:layout\_constraintTop\_toBottomOf="@+id/circularProgressBar" />  <com.google.android.material.progressindicator.CircularProgressIndicator  android:id="@+id/circularProgressBar"  android:layout\_width="120dp"  android:layout\_height="120dp"  android:layout\_marginTop="25dp"  android:indeterminate="true"  android:visibility="gone"  app:indicatorColor="@color/green"  app:indicatorDirectionCircular="clockwise"  app:indicatorSize="120dp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:trackColor="@color/red"  app:trackThickness="20dp" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnRegister"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:background="#D61D1D"  android:fontFamily="monospace"  android:text="@string/send\_otp"  android:textColor="#FFFFFF"  android:textSize="23sp"  app:layout\_constraintTop\_toBottomOf="@+id/edPhone" />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | FirebaseAuthDemo1-Gmail | <chare> | 1 | <pext> | 02-28/FirebaseAuthDemo1-Gmail  Enable google signin service on Firebase  \*\*\*ProfileActivity\*\*\*  class ProfileActivity : AppCompatActivity() {  private lateinit var binding: ActivityProfileBinding  private lateinit var googleSignInClient: GoogleSignInClient  private lateinit var firebaseAuth: FirebaseAuth  val REQ\_CODE = 100  --------------needto copy clientID-----------------  val clientId = "723884935107-rgjn083n9fqjhk5uqb1n3130hfvuc8db.apps.googleusercontent.com"  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityProfileBinding.inflate(layoutInflater)  setContentView(binding.root)  setUp()  binding.googleSignInButton.setOnClickListener {  signInUsingGoogle()  }  }  private fun setUp() {  firebaseAuth = FirebaseAuth.getInstance()  ---------build GoogleSignInOptions-------------  val googleSignInOptions = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT\_SIGN\_IN)  .requestIdToken(clientId)  .requestEmail()  .build()  ---------get GoogleSignIn Client----------  googleSignInClient = GoogleSignIn.getClient(this, googleSignInOptions)  }  private fun signInUsingGoogle() {  ----------start googleSignInClient.signInIntent----------  val intent = googleSignInClient.signInIntent  startActivityForResult(intent, REQ\_CODE)  }  override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  super.onActivityResult(requestCode, resultCode, data)  if (requestCode == REQ\_CODE) {  ---------------success callback process data---------  val task: Task<GoogleSignInAccount> = GoogleSignIn.getSignedInAccountFromIntent(data)  handleTask(task)  }  }  private fun handleTask(task: Task<GoogleSignInAccount>) {  try {  val account = task.getResult(ApiException::class.java)  account?.let {  updateUI(account)  }  } catch (e: Exception) {  e.printStackTrace()  }  }  private fun updateUI(account: GoogleSignInAccount) {  val credentials = GoogleAuthProvider.getCredential(account.idToken, null)  ------------------signin with google info-------------  firebaseAuth.signInWithCredential(credentials)  .addOnCompleteListener { task ->  --------------on success, use account Information----------  if (task.isSuccessful) {  binding.apply {  account.apply {  Glide  .with(this@ProfileActivity)  .load(photoUrl)  .into(profilePicture)  txtName.text = displayName  txtEmail.text = email  }  }  } else {  Toast.makeText(this, "Error", Toast.LENGTH\_SHORT).show()  }  }  }  }  <?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"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/nav\_root"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:layout\_marginTop="50dp"  android:background="#FF3D00"  android:padding="10dp">  <TextView  android:id="@+id/txtName"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:fontFamily="casual"  android:textColor="@color/white"  android:textSize="26sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/profilePicture"  tools:text="Abhishek Pathak" />  <TextView  android:id="@+id/txtEmail"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:fontFamily="casual"  android:textColor="@color/white"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/txtName"  tools:text="abhishek.pathak@000gmail.com" />  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/profilePicture"  android:layout\_width="90dp"  android:layout\_height="90dp"  android:layout\_margin="10dp"  android:cropToPadding="true"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:srcCompat="@drawable/ic\_launcher\_background" />  <com.google.android.gms.common.SignInButton  android:id="@+id/googleSignInButton"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | Git/Whatsapp-Chatting-Firebase-RealtimeDB | <chare> | 1 | <pext> | 03-01/Git/Whatsapp-Chatting-Firebase-RealtimeDB  https://github.com/cheetahmail007/Whatsapp-Chatting-Sample\_Work | </end> |
| <hitle> | FirebasePushNotification Local/Simple Push | <chare> | 1 | <pext> | 03-02/FirebasePushNotification Local/Simple Push  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private val channelId = "channelId"  private lateinit var notificationManager: NotificationManager  private lateinit var notificationBuilder: Notification.Builder  val PERMISSION\_REQUEST\_CODE = 112  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  --------------request notification permission-------------  if (Build.VERSION.SDK\_INT > 32) {  if (!shouldShowRequestPermissionRationale("112")) {  getNotificationPermission()  }  }  binding.btnSimpleNotify.setOnClickListener {  makeSimpleNotification()  }  }  -------------create NotificationChannel------------  private fun createNotificationChannel() {  -------------get notification manager by getSystemService(NOTI)--------  notificationManager = getSystemService(Context.NOTIFICATION\_SERVICE) as NotificationManager  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  val channel = NotificationChannel(  channelId,  "desc",  NotificationManager.IMPORTANCE\_HIGH  ).also {  it.enableLights(true)  it.enableVibration(true)  }  -----------create NotificationChannel by notificationManager--------------  notificationManager.createNotificationChannel(channel)  }  }  fun makeSimpleNotification() {  createNotificationChannel()  ---------------create notificationBuilder with title, icon, etc---------------  notificationBuilder = if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  Notification.Builder(this, channelId)  .setContentTitle("testing")  .setContentText("content")  .setSmallIcon(R.drawable.ic\_launcher\_background)  ----------------set pendingIntent with IMMITABLE flag--------  .setContentIntent(setPendingIntent(this))  } else {  Notification.Builder(this)  .setContentTitle("testing")  .setContentText("content")  .setSmallIcon(R.drawable.ic\_launcher\_background)  .setContentIntent(setPendingIntent(this))  }  --------------------send notification by notificationManager-------------  notificationManager.notify(1, notificationBuilder.build())  }  private fun setPendingIntent(context: Context): PendingIntent? {  val intent = Intent(context, MainActivity::class.java)  return PendingIntent.getActivity(  this,  0,  intent,  PendingIntent.FLAG\_IMMUTABLE  )  }  private fun getNotificationPermission() {  try {  if (Build.VERSION.SDK\_INT > 32) {  ActivityCompat.requestPermissions(  this, arrayOf<String>(Manifest.permission.POST\_NOTIFICATIONS),  PERMISSION\_REQUEST\_CODE  )  }  } catch (e: Exception) {  e.printStackTrace()  }  }  override fun onRequestPermissionsResult(  requestCode: Int,  permissions: Array<out String>,  grantResults: IntArray  ) {  super.onRequestPermissionsResult(requestCode, permissions, grantResults)  when (requestCode) {  PERMISSION\_REQUEST\_CODE -> {  if (grantResults.isNotEmpty() &&  grantResults[0] == PackageManager.PERMISSION\_GRANTED  ) {  } else {  }  return  }  }  }  }  --------Manifest POST\_NOTIFICATION >Api33-------  <uses-permission android:name="android.permission.POST\_NOTIFICATIONS" /> | </end> |
| <hitle> | FirebasePushNotification FCM BigPicture | <chare> | 1 | <pext> | 03-02/ FirebasePushNotification FCM BigPicture  --------prepare FirebaseApp-----------  class MyApp: Application() {  override fun onCreate() {  super.onCreate()  FirebaseApp.initializeApp(this)  }  }  \*\*\*Manifest\*\*\*  <uses-permission android:name="android.permission.POST\_NOTIFICATIONS" />  <application  android:name=".MyApp" >  <service  android:name=".MyFirebaseMessagingService"  android:exported="true">  <intent-filter>  <action android:name="com.google.firebase.INSTANCE\_ID\_EVENT" />  <action android:name="com.google.firebase.MESSAGING\_EVENT" />  </intent-filter>  </service>  </application>  --------customize FirebaseMessagingService------------  class MyFirebaseMessagingService: FirebaseMessagingService() {  private val channelId = "channelId"  private lateinit var notificationManager: NotificationManager  private lateinit var notificationBuilder: Notification.Builder  ---------------create notification channel--------------  private fun getNotificationChannel() {  notificationManager = getSystemService(Context.NOTIFICATION\_SERVICE) as NotificationManager  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  val channel = NotificationChannel(  channelId,  "anything",  NotificationManager.IMPORTANCE\_HIGH  ).also {  it.enableLights(true)  it.enableVibration(true)  }  notificationManager.createNotificationChannel(channel)  }  }  ----------------prepare PendingIntent will be used in notification builder----------  private fun setPendingIntent(context: Context): PendingIntent? {  val intent = Intent(context, MainActivity::class.java)  return PendingIntent.getActivity(  this,  0,  intent,  PendingIntent.FLAG\_IMMUTABLE  )  }  ---------------prepare bitmap from image url--------------------  private fun getBitmapFromURL(strURL: String?): Bitmap? {  return try {  val url = URL(strURL)  val connection: HttpURLConnection = url.openConnection() as HttpURLConnection  connection.doInput = true  connection.connect()  val input: InputStream = connection.inputStream  BitmapFactory.decodeStream(input)  } catch (e: IOException) {  e.printStackTrace()  null  }  }  -------------onMessageReceived() is important-----------  override fun onMessageReceived(message: RemoteMessage) {  super.onMessageReceived(message)  val notification = message.notification  makeBigPictureNotification(notification)  }  ---------------make BigPictureNofication and notify----------------  private fun makeBigPictureNotification(message: RemoteMessage.Notification?) {  getNotificationChannel()  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  val style = NotificationCompat.BigPictureStyle()  val bitmap = getBitmapFromURL(message?.imageUrl.toString())  style.bigPicture(bitmap).build()  val notificationBuilder = NotificationCompat.Builder(this, channelId)  .setContentTitle(message?.title)  .setContentText(message?.body)  .setSmallIcon(R.drawable.ic\_launcher\_background)  .setStyle(style)  .setContentIntent(setPendingIntent(this))  notificationManager.notify(1, notificationBuilder.build())  }  }  ------------make SimpleNotification and nofity-----------------  private fun makeSimpleNotification(message: RemoteMessage) {  getNotificationChannel()  notificationBuilder = if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  Notification.Builder(this, channelId)  .setContentTitle("testing")  .setContentText("content")  .setSmallIcon(R.drawable.ic\_launcher\_background)  .setContentIntent(setPendingIntent(this))  } else {  Notification.Builder(this)  .setContentTitle("testing")  .setContentText("content")  .setSmallIcon(R.drawable.ic\_launcher\_background)  .setContentIntent(setPendingIntent(this))  }  notificationManager.notify(1, notificationBuilder.build())  }  }  \*\*\*FCMActivity\*\*\*  class FCMActivity : AppCompatActivity() {  private lateinit var binding: ActivityFcmactivity2Binding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityFcmactivity2Binding.inflate(layoutInflater)  setContentView(binding.root)  registerToFCM()  }  private fun registerToFCM() {  -----------FirebaseMessaging token lisener-----------  FirebaseMessaging.getInstance().token.addOnCompleteListener { task ->  if (task.isSuccessful) {  Log.i("pmh", "${task.result.toString()}")  }  }  }  } | </end> |
| <hitle> | BroadcastReceiverDemo | <chare> | 1 | <pext> | 03-03/BroadcastReceiverDemo  -----------register custom broadcastreceivers in Manifest---------  <application>  <receiver android:name=".AirPlaneModeDetector" />  <receiver android:name=".MainActivity$MyLocalBroadcastReceiver" />  </application>  ------------implement using BroadcastReceiver class--------  class AirPlaneModeDetector: BroadcastReceiver() {  ------------onReceive is where to work-----------  override fun onReceive(context: Context?, intent: Intent) {  val isAirPlaneModeEnabled = intent.getBooleanExtra("state", false)  if (isAirPlaneModeEnabled) {  showToast( context,"Airplane mode is enabled")  } else {  showToast( context, "Airplane mode is disabled")  }  }  private fun showToast(context: Context?, message: String) {  Toast.makeText(context, message, Toast.LENGTH\_SHORT).show()  }  }  \*\*\*MainActivity\*\*\*\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var airPlaneModeDetector: AirPlaneModeDetector  private lateinit var localBroadcastReceiver: MyLocalBroadcastReceiver  private lateinit var localBroadcastManager: LocalBroadcastManager  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  globalBroadcast()  localBroadcast()  }  private fun localBroadcast() {  binding.btnPlayGame.setOnClickListener {  ------------------register localBroadcastReceiver-------  ---------------registerReceiver using BroadCastManager----------  localBroadcastManager = LocalBroadcastManager.getInstance(this)  localBroadcastReceiver = MyLocalBroadcastReceiver()  localBroadcastManager.registerReceiver(localBroadcastReceiver, IntentFilter("PlayGame"))  val localIntent = Intent("PlayGame")  .putExtra("data", "Mobo legend")  localBroadcastManager.sendBroadcast(localIntent)  }  }  ----------------register globalBroadcastReceiver----------  private fun globalBroadcast() {  airPlaneModeDetector = AirPlaneModeDetector()  IntentFilter(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED).also {  ------------Activity.registerRecever to current Activity-----------------  registerReceiver(airPlaneModeDetector, it)  }  }  private fun initViews() {  airPlaneModeDetector = AirPlaneModeDetector()  IntentFilter(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED).also {  registerReceiver(airPlaneModeDetector, it)  }  }  override fun onStop() {  super.onStop()  --------------important to unregisterReceiver onStop--------  unregisterReceiver(airPlaneModeDetector)  unregisterReceiver(localBroadcastReceiver)  }  private fun showToast(message: String) {  Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()  }  ------------howto implement LocalBroadcastReceiver---------------  inner class MyLocalBroadcastReceiver(): BroadcastReceiver() {  override fun onReceive(context: Context?, intent: Intent) {  when (intent.action) {  "PlayGame" -> {  val result = intent.getStringExtra("data")  showToast("result is $result")  }  else -> {  showToast("Action not found")  }  }  }  }  } | </end> |
| <hitle> | ContentProviderDemo | <chare> | 1 | <pext> | 03-03/ContentProviderDemo  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.READ\_CONTACTS" />  <application>  ------------register provider in Manifest----------------  <provider  android:authorities="com.example.contentproviderdemo"  android:name=".MyContentProvider"  android:exported="false" />  </application>  </manifest>  \*\*\*\*MainActivity\*\*\*\*\*\*\*\*\*\*\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnFetch.setOnClickListener {  loadContacts()  }  }  private fun loadContacts() {  val builder: StringBuilder  if(Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M &&  checkSelfPermission(Manifest.permission.READ\_CONTACTS) != PackageManager.PERMISSION\_GRANTED) {  requestPermissions(  arrayOf(Manifest.permission.READ\_CONTACTS),  READ\_CONTACTS\_REQUEST\_CODE  )  } else {  builder = getContacts()  binding.txtContacts.text = builder.toString()  }  }  @SuppressLint("Range")  private fun getContacts(): StringBuilder {  val builder = StringBuilder()  --------query contacts using contentResolver-----------  val cursor = contentResolver.query(ContactsContract.Contacts.CONTENT\_URI, null, null, null, null)  cursor?.let {  if (cursor.count > 0) {  while (cursor.moveToNext()) {  val id = cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.\_ID))  val name = cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY\_NAME))  val phoneNumber = cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.HAS\_PHONE\_NUMBER)).toInt()  if (phoneNumber>0) {  val cursorPhone = contentResolver.query(  ContactsContract.CommonDataKinds.Phone.CONTENT\_URI,  null,  ContactsContract.CommonDataKinds.Phone.CONTACT\_ID + " =?",  arrayOf(id),  null  )  cursorPhone?.let {  if (cursorPhone.count>0) {  while (cursorPhone.moveToNext()) {  val phoneNumber = cursorPhone.getString(  cursorPhone.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER)  )  builder.append("Contacts: $name")  .append("Phone number is $phoneNumber")  .append("\n")  Log.i("pmh", "name is $name")  }  }  cursorPhone.close()  }  }  }  }  cursor.close()  }  return builder  }  companion object {  const val READ\_CONTACTS\_REQUEST\_CODE = 100  }  }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ------------ContentProvider functions-------------  class MyContentProvider: ContentProvider() {  override fun onCreate(): Boolean {  TODO("Not yet implemented")  }  override fun query(  uri: Uri,  projection: Array<out String>?,  selection: String?,  selectionArgs: Array<out String>?,  sortOrder: String?  ): Cursor? {  TODO("Not yet implemented")  }  override fun getType(uri: Uri): String? {  TODO("Not yet implemented")  }  override fun insert(uri: Uri, values: ContentValues?): Uri? {  TODO("Not yet implemented")  }  override fun delete(uri: Uri, selection: String?, selectionArgs: Array<out String>?): Int {  TODO("Not yet implemented")  }  override fun update(  uri: Uri,  values: ContentValues?,  selection: String?,  selectionArgs: Array<out String>?  ): Int {  TODO("Not yet implemented")  }  } | </end> |
| <hitle> | ForegroundServiceDemo | <chare> | 1 | <pext> | 03-06/ForegroundServiceDemo  <service android:name=".ForegroundService" />  </application>  <uses-permission android:name="android.permission.FOREGROUND\_SERVICE" />  <uses-permission android:name="android.permission.POST\_NOTIFICATIONS" />  <uses-permission android:name="android.permission.WAKE\_LOCK" />  ---------extend Service-------------  class ForegroundService:Service() {  override fun onBind(intent: Intent?): IBinder? {  return null  }  ---------------override onStartCommand where run task/work--------  override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {  createNotificationChannel()  --------------to show notification, we need to use pendingIntent----------  val pendingIntent = PendingIntent.getActivity(  this,  0,  Intent(this, MainActivity::class.java),  PendingIntent.FLAG\_IMMUTABLE or PendingIntent.FLAG\_UPDATE\_CURRENT  )  val notification = NotificationCompat.Builder(this,"ChannelId" )  .setContentTitle("Foreground service...")  .setContentText("Service is running!!")  .setContentIntent(pendingIntent)  .build()  ------------- make service run in the foreground-----------  startForeground(1,notification)  return START\_NOT\_STICKY  }  private fun createNotificationChannel() {  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  val serviceChannel = NotificationChannel(  "ChannelId", "ForeGround service",  NotificationManager.IMPORTANCE\_HIGH  )  val manager = getSystemService(NotificationManager::class.java)  manager.createNotificationChannel(serviceChannel)  }  }  }  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  getNotificationPermission()  binding.start.setOnClickListener {  ---------------create intent targeting service---------  val intent = Intent(this, ForegroundService::class.java)  intent.putExtra("IntentExtra","Stared. enjoy")  ---------------startForegroundService with intent---------  ContextCompat.startForegroundService(this, intent)  }  binding.stop.setOnClickListener {  --------------to stop foreground service we need intent----------  val intent = Intent(this, ForegroundService::class.java)  stopService(intent)  }  }  override fun onRequestPermissionsResult(  requestCode: Int,  permissions: Array<String?>,  grantResults: IntArray  ) {  super.onRequestPermissionsResult(requestCode, permissions, grantResults)  when (requestCode) {  100 -> {  // If request is cancelled, the result arrays are empty.  if (grantResults.isNotEmpty() &&  grantResults[0] == PackageManager.PERMISSION\_GRANTED  ) {  // allow  } else {  //deny  }  return  }  }  }  private fun getNotificationPermission() {  try {  if (Build.VERSION.SDK\_INT > 32) {  ActivityCompat.requestPermissions(  this, arrayOf<String>(Manifest.permission.POST\_NOTIFICATIONS),  100  )  }  } catch (e: Exception) {  e.printStackTrace()  }  }  } | </end> |
| <hitle> | IntentServiceDemo | <chare> | 1 | <pext> | 03-06/IntentServiceDemo  <service android:name=".DemoIntentService" />  ------------extend IntentService-------------  class DemoIntentService : IntentService("DemoIntentService") {  ------------onHandleIntent where define work/task-----------  override fun onHandleIntent(intent: Intent?) {  intent?.let {  for (i in 1..35){  Log.i("pmh", "value is $i data is ${intent.getStringExtra("data")}")  Thread.sleep(500)  Log.i("pmh","${Thread.currentThread().name}")  }  }  Log.i("tag", "Doing some background work")  }  --------can also override onDestroy()----------  override fun onDestroy() {  super.onDestroy()  Log.i("pmh", "intent Service finishing")  Log.i("pmh","${Thread.currentThread().name}")  }  }  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.root.setOnClickListener {  -------------create intent targeting IntentService---------  Log.i("pmh","${Thread.currentThread().name}")  val intent = Intent(this, DemoIntentService::class.java)  intent.putExtra("data", "Hey i am intent service")  -----------startService with intent--------------  startService(intent)  }  }  } | </end> |
| <hitle> | JobIntentServiceDemo | <chare> | 1 | <pext> | 03-06/JobIntentServiceDemo  <service  android:name=".JobIntentServiceDemo"  android:exported="true"  android:permission="android.permission.BIND\_JOB\_SERVICE" />  -----------extend JobIntentService----------  class JobIntentServiceDemo : JobIntentService() {  ------------onHandleIntent where define work/task-----------  override fun onHandleWork(intent: Intent) {  intent.let {  val maxCount = intent.getIntExtra("data", -1)  for (i in 0 until maxCount) {  try {  Thread.sleep(2000)  Log.i("pmh", Thread.currentThread().name.toString())  Log.i("pmh", "value is $i}")  } catch (e: Exception) {  e.printStackTrace()  }  }  }  }  override fun onDestroy() {  super.onDestroy()  Log.i("pmh", "intent Service finishing")  Log.i("pmh", Thread.currentThread().name.toString())  }  companion object {  ---------job id is needed to take care---------  const val JOB\_ID = 1  }  }  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.root.setOnClickListener {  ------------create intent targeting JobIntentService--------  val intent = Intent(this, JobIntentServiceDemo::class.java)  intent.putExtra("data", 10)  ----------enqueueWork with intent and predefined job id-----------  JobIntentService.enqueueWork(this, JobIntentServiceDemo::class.java, JOB\_ID, intent)  }  }  } | </end> |
| <hitle> | JobSchedulerServiceDemo | <chare> | 1 | <pext> | 03-06/JobSchedulerServiceDemo  <service  android:name=".MyJobScheduler"  android:permission="android.permission.BIND\_JOB\_SERVICE" />  ------------extend JobService-----------  class MyJobScheduler :JobService(){  --------onStartJob where define task/work----------  override fun onStartJob(params: JobParameters?): Boolean {  for (i in 1..35){  Log.i("pmh", "value is $i")  Thread.sleep(500)  Log.i("pmh", Thread.currentThread().name.toString())  }  Log.i("pmh", "Job finished!!")  Toast.makeText(applicationContext,"Job finished!!",Toast.LENGTH\_LONG ).show()  return false  }  --------------handle service stop onStopJob------------  override fun onStopJob(params: JobParameters?): Boolean {  Toast.makeText(applicationContext,"Job Cancelled!!",Toast.LENGTH\_LONG ).show()  return false  }  ------------handle onLowMemory------------  override fun onLowMemory() {  super.onLowMemory()  Log.i("pmh", "Job running on Low Memory")  }  }  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var jobScheduler: JobScheduler  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun initViews() {  binding.start.setOnClickListener {  --------------get system job scheduler service instance----------  jobScheduler = getSystemService(JOB\_SCHEDULER\_SERVICE) as JobScheduler  -----------need componentname for jobscheduler-----------  val component = ComponentName(this, MyJobScheduler::class.java)  ------------get jobinfo with component name--------  val jobInfo = JobInfo.Builder(1, component)  .setMinimumLatency(3000)  .build()  -------------finally schedule with this job info---------  jobScheduler.schedule(jobInfo)  }  binding.stop.setOnClickListener {  if (this::jobScheduler.isInitialized){  // jobScheduler.cancel(1)  ------------we can cancel jobScheduler----------  jobScheduler.cancelAll()  }  }  }  } | </end> |
| <hitle> | MusicService MediaPlayer | <chare> | 1 | <pext> | 03-06/MusicBoundService MediaPlayer  <service android:name=".MusicService" />  ------------extend Service---------  class MusicService : Service() {  ---------customize Binder if needed-----------  inner class MediaController : Binder() {  fun play() {  mediaPlayer.start()  }  fun stop() {  mediaPlayer.stop()  }  fun pause() {  if (mediaPlayer.isPlaying) {  mediaPlayer.pause()  }  }  }  private lateinit var mediaPlayer: MediaPlayer  private lateinit var mediaController: MediaController  ----------prepare Binder onCreate of Service-------  override fun onCreate() {  super.onCreate()  mediaPlayer = MediaPlayer()  mediaPlayer.apply {  setDataSource(MUSIC\_2)  prepareAsync()  setOnPreparedListener { it.start() }  }  }  -----------onBind we will return custom binder----------  override fun onBind(intent: Intent?): IBinder {  if (!this::mediaController.isInitialized) {  mediaController = MediaController()  }  return mediaController  }  -----------another important is onUnbind-------------  override fun onUnbind(Intent intent) {  return super.onUnbind(intent);  }  ----------if we have heavy operatioins we need to do something in onDestroy---------  override fun onDestroy() {  super.onDestroy()  mediaPlayer.stop()  }  }  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var mediaController: MusicService.MediaController  -------------implement ServiceConnection interface---------  private val serviceConnection = object : ServiceConnection {  override fun onServiceConnected(name: ComponentName?, service: IBinder?) {  -----------we may retain Binder instance to control service---------  mediaController = service as MusicService.MediaController  }  override fun onServiceDisconnected(name: ComponentName?) {  }  }  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  --------------create intent targeting our custom Service-----  val intent = Intent(this, MusicService::class.java)  -------------bindService using intent------------  bindService(intent, serviceConnection, BIND\_AUTO\_CREATE)  binding.textViewMusic.isSelected = true  binding.start.setOnClickListener {  mediaController.play()  }  binding.stop.setOnClickListener {  mediaController.stop()  }  binding.pause.setOnClickListener {  mediaController.pause()  }  }  override fun onDestroy() {  super.onDestroy()  mediaController.stop()  unbindService(serviceConnection)  }  }  object Constant {  const val MUSIC\_1 = "https://www.soundhelix.com/examples/mp3/SoundHelix-Song-6.mp3"  const val MUSIC\_2 = "https://dl.espressif.com/dl/audio/ff-16b-2c-44100hz.mp3"  } | </end> |
| <hitle> | GoogleMap Fused Find Current Location Api | <chare> | 1 | <pext> | 03-07/GoogleMapDemo Custom Marker  --------------------------Gradle--------------------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'com.google.android.libraries.mapsplatform.secrets-gradle-plugin'  }  android {  namespace 'com.example.googlemapdemo'  compileSdk 33  defaultConfig {  applicationId "com.example.googlemapdemo"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures {  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'com.google.android.gms:play-services-maps:18.1.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  implementation 'com.google.android.gms:play-services-location:21.0.1'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  }  ---------------------permission and google service key Manifest-------------  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />  <uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />  <uses-permission android:name="android.permission.ACCESS\_BACKGROUND\_LOCATION" />  <application>  <meta-data  android:name="com.google.android.geo.API\_KEY"  android:value="AIzaSyC\_1VVH9j0zQCeFYf6QEIEM3Oc7LsJA8xM" />  </application>  --------------------------ui/FindMyCurrentLocation--------------------------  @RequiresApi(Build.VERSION\_CODES.M)  class FindMyCurrentLocationActivity : AppCompatActivity() {  private lateinit var binding: ActivityFindMyCurrentLocationBinding  ------------get fusedLocationProviderClient by lazy  private val fusedLocationProviderClient: FusedLocationProviderClient by lazy {  LocationServices.getFusedLocationProviderClient(applicationContext)  }  ----------create cancellationTokenSource-------------  private var cancellationTokenSource = CancellationTokenSource()  private val snackBar by lazy {  Snackbar.make(binding.container, getString(R.string.get\_location), Snackbar.LENGTH\_LONG)  .setAction(getString(R.string.ok)) {  requestPermissions(  arrayOf(Manifest.permission.ACCESS\_FINE\_LOCATION),  REQ\_CODE  )  }  }  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityFindMyCurrentLocationBinding.inflate(layoutInflater)  setContentView(binding.root)  requestForPermission()  }  -------request to allow permissions------------  private fun requestForPermission() {  val permission = applicationContext.hasPermission(Manifest.permission.ACCESS\_FINE\_LOCATION)  if (permission) {  requestForCurrentLocation()  } else {  requirePermissionWithRationale(  Manifest.permission.ACCESS\_FINE\_LOCATION,  REQ\_CODE,  snackBar  )  }  }  --------------request current location by fusedLocationProviderClient()----------------  @SuppressLint("MissingPermission")  private fun requestForCurrentLocation() {  if (applicationContext.hasPermission(Manifest.permission.ACCESS\_FINE\_LOCATION)) {  val currentTask: Task<Location> = fusedLocationProviderClient.getCurrentLocation(  PRIORITY\_HIGH\_ACCURACY,  cancellationTokenSource.token  )  currentTask.addOnCompleteListener { task: Task<Location> ->  if (task.isSuccessful) {  task.result?.let {  val intent = Intent(this, MapsActivity::class.java)  intent.putExtra("lat", it.latitude)  intent.putExtra("lng", it.longitude)  startActivity(intent)  }  } else {  Log.i("tag", "Bad fetch")  }  }  }  }  override fun onRequestPermissionsResult(  requestCode: Int,  permissions: Array<out String>,  grantResults: IntArray  ) {  if (requestCode == REQ\_CODE) {  when {  grantResults.isEmpty() -> Log.i("tag", "user cancelled")  grantResults[0] == PackageManager.PERMISSION\_GRANTED ->  Snackbar.make(  binding.container,  getString(R.string.approved),  Snackbar.LENGTH\_LONG  ).show()  }  } else {  Snackbar.make(  binding.container,  getString(R.string.pemission\_denied),  Snackbar.LENGTH\_LONG  )  .setAction(getString(R.string.settings)) {  val intent = Intent()  intent.action = Settings.ACTION\_APPLICATION\_DETAILS\_SETTINGS  intent.data = Uri.fromParts("package", "com.example.googlemapdemo", null)  intent.flags = Intent.FLAG\_ACTIVITY\_NEW\_TASK  startActivity(intent)  }.show()  }  super.onRequestPermissionsResult(requestCode, permissions, grantResults)  }  companion object {  private const val REQ\_CODE = 100  }  }  --------------------------ui/MapsActivity--------------------------  class MapsActivity : AppCompatActivity(), OnMapReadyCallback {  private lateinit var map: GoogleMap  private lateinit var binding: ActivityMapsBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMapsBinding.inflate(layoutInflater)  setContentView(binding.root)  // Obtain the SupportMapFragment and get notified when the map is ready to be used.  val mapFragment = supportFragmentManager  .findFragmentById(R.id.map) as SupportMapFragment  mapFragment.getMapAsync(this)  }  override fun onMapReady(googleMap: GoogleMap) {  map = googleMap  // Add a marker in Sydney and move the camera  val myLocation = LatLng(  intent.getDoubleExtra("lat", 0.0), intent.getDoubleExtra("lng", 0.0)  )  val sydney = LatLng(-34.0, 151.0)  val london = LatLng(51.5, -0.12)  val home = LatLng(51.515968, -0.237460)  val office = LatLng(51.525990, -0.087710)  val covidAffected = LatLng(51.460930, -0.116020)  val place1 = LatLng(51.550331, -0.292560)  val place2 = LatLng(51.566080, -0.220110)  val place3 = LatLng(51.513350, -0.304210)  map.apply {  mapType = GoogleMap.MAP\_TYPE\_TERRAIN  addMarker(MarkerOptions().position(sydney).title("Marker in Sydney"))  addMarker(MarkerOptions().position(london).title("Marker in London"))  addMarker(MarkerOptions().position(place1).title("Marker in Wembley"))  addMarker(MarkerOptions().position(place2).title("Marker in Cricklewood"))  addMarker(MarkerOptions().position(place3).title("Marker in ealing"))  addMarker(  MarkerOptions().position(home).title("Home")  .icon(bitmapFromVector(R.drawable.baseline\_home\_24))  )  addMarker(  MarkerOptions().position(myLocation).title("Marker in current location")  .icon(bitmapFromVector(R.drawable.baseline\_add\_location\_24))  )  addMarker(  MarkerOptions().position(office).title("Office")  .icon(bitmapFromVector(R.drawable.baseline\_business\_center\_24))  )  addPolyline(  PolylineOptions().add(home).add(office).color(Color.CYAN).geodesic(true)  .visible(true).width(10.0f)  )  addCircle(  CircleOptions().radius(1000.00).center(covidAffected).fillColor(Color.RED)  .strokeWidth(2.0f).strokeColor(Color.BLACK)  )  addPolyline(  PolylineOptions().add(place1).add(place2).add(place3).add(place1)  .color(Color.MAGENTA).geodesic(true)  .visible(true).width(15.0f)  )  moveCamera(CameraUpdateFactory.newLatLngZoom(myLocation, 11.0f))  }  }  private fun bitmapFromVector(vectorResId: Int): BitmapDescriptor {  val vectorDrawable = ContextCompat.getDrawable(this, vectorResId)!!.apply {  setBounds(0, 0, intrinsicWidth, intrinsicHeight)  }  val bitmap = Bitmap.createBitmap(  vectorDrawable.intrinsicWidth,  vectorDrawable.intrinsicHeight,  Bitmap.Config.ARGB\_8888  )  val canvas = Canvas(bitmap)  vectorDrawable.draw(canvas)  return BitmapDescriptorFactory.fromBitmap(bitmap)  }  }  --------------------------util/Helper extentions--------------------------  /\*  If your app targets Android 10 (API level 29) or higher  \*/  fun Context.hasPermission(permission: String): Boolean {  if (permission == Manifest.permission.ACCESS\_BACKGROUND\_LOCATION  && Build.VERSION.SDK\_INT > Build.VERSION\_CODES.Q  ) {  return true  }  return ActivityCompat.checkSelfPermission(this, permission) == PackageManager.PERMISSION\_GRANTED  }  fun Activity.requirePermissionWithRationale(  permission: String,  requestCode: Int,  snackbar: Snackbar  ) {  val provideRationale = shouldShowRequestPermissionRationale(permission)  if (provideRationale) {  snackbar.show()  } else {  requestPermissions(arrayOf(permission), requestCode)  }  }  ------------------AndroidManifest------------------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />  <uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />  <uses-permission android:name="android.permission.ACCESS\_BACKGROUND\_LOCATION" />  <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.GoogleMapDemo"  android:usesCleartextTraffic="true"  tools:targetApi="31">  <activity  android:name=".ui.FindMyCurrentLocationActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  <meta-data  android:name="com.google.android.geo.API\_KEY"  android:value="AIzaSyC\_1VVH9j0zQCeFYf6QEIEM3Oc7LsJA8xM" />  <activity  android:name=".ui.MapsActivity"  android:label="@string/title\_activity\_maps"  />  </application>  </manifest> | </end> |
| <hitle> | Google Fused Api current location | <chare> | 1 | <pext> | 03-08/Google Fused Api current location  --------------------------Gradle--------------------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'com.google.android.libraries.mapsplatform.secrets-gradle-plugin'  }  android {  namespace 'com.example.googlemapdemo'  compileSdk 33  defaultConfig {  applicationId "com.example.googlemapdemo"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures {  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'com.google.android.gms:play-services-maps:18.1.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  implementation 'com.google.android.gms:play-services-location:21.0.1'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  }  ---------------------permission and google service key Manifest-------------  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />  <uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />  <uses-permission android:name="android.permission.ACCESS\_BACKGROUND\_LOCATION" />  <application>  <meta-data  android:name="com.google.android.geo.API\_KEY"  android:value="AIzaSyC\_1VVH9j0zQCeFYf6QEIEM3Oc7LsJA8xM" />  </application>  \*\*\*MainActivity\*\*\*  class MapsActivity : AppCompatActivity(), OnMapReadyCallback {  private lateinit var mMap: GoogleMap  private lateinit var binding: ActivityMapsBinding  --------fusedLocationProviderClient by LocationServies------------  private val fusedLocationProviderClient: FusedLocationProviderClient by lazy {  LocationServices.getFusedLocationProviderClient(applicationContext)  }  private var cancellationTokenSource = CancellationTokenSource()  private lateinit var locationCallback: LocationCallback  private lateinit var mapFragment: SupportMapFragment  private var currentLocation: Location? = null  private val snackBar by lazy {  Snackbar.make(binding.root, "Location", Snackbar.LENGTH\_LONG)  .setAction("OK") {  requestPermissions(  arrayOf(Manifest.permission.ACCESS\_FINE\_LOCATION),  REQ\_CODE  )  }  }  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMapsBinding.inflate(layoutInflater)  setContentView(binding.root)  // Obtain the SupportMapFragment and get notified when the map is ready to be used.  mapFragment = supportFragmentManager  .findFragmentById(R.id.map) as SupportMapFragment  mapFragment.getMapAsync(this)  requestForPermission()  registerForCallback()  }  private fun registerForCallback() {  ------------locationCallback when location changed-------  locationCallback = object : LocationCallback() {  override fun onLocationResult(locationResult: LocationResult) {  for (location in locationResult.locations) {  currentLocation = location  mapFragment.getMapAsync(this@MapsActivity)  break  }  }  }  }  ----------onMapReady is called by mapFragment.getMapAsync(this)------  override fun onMapReady(googleMap: GoogleMap) {  mMap = googleMap  var curLatLng = LatLng(-34.0, 151.0)  // Add a marker in Sydney and move the camera  currentLocation?.let {  curLatLng = LatLng(it.latitude, it.longitude)  }  mMap.addMarker(MarkerOptions().position(curLatLng).title("My current location"))  mMap.moveCamera(CameraUpdateFactory.newLatLng(curLatLng))  }  private fun requestForPermission() {  val permission = appHasPermission(Manifest.permission.ACCESS\_FINE\_LOCATION)  if (permission) {  requestForCurrentLocation()  } else {  val provideRationale = shouldShowRequestPermissionRationale(Manifest.permission.ACCESS\_FINE\_LOCATION)  if (provideRationale) {  snackBar.show()  } else {  requestPermissions(arrayOf(Manifest.permission.ACCESS\_FINE\_LOCATION), REQ\_CODE)  }  }  }  @SuppressLint("MissingPermission")  private fun requestForCurrentLocation() {  -----------request permission to use Fused Api-------  if (appHasPermission(Manifest.permission.ACCESS\_FINE\_LOCATION)) {  val currentTask: Task<Location> = fusedLocationProviderClient.getCurrentLocation(  PRIORITY\_HIGH\_ACCURACY,  cancellationTokenSource.token  )  currentTask.addOnCompleteListener { task: Task<Location> ->  if (task.isSuccessful) {  task.result?.let {  currentLocation = it  mapFragment.getMapAsync(this)  }  } else {  Log.i("tag", "Bad fetch")  }  }  }  }  -----------onRequestPermissionsResult after permission request------  override fun onRequestPermissionsResult(  requestCode: Int,  permissions: Array<out String>,  grantResults: IntArray  ) {  if (requestCode == REQ\_CODE) {  when {  grantResults.isEmpty() -> Log.i("tag", "user cancelled")  grantResults[0] == PackageManager.PERMISSION\_GRANTED ->  Snackbar.make(  binding.root,  "Approved",  Snackbar.LENGTH\_LONG  ).show()  }  } else {  Snackbar.make(  binding.root,  "Denied",  Snackbar.LENGTH\_LONG  )  .setAction("Settings") {  val intent = Intent()  intent.action = Settings.ACTION\_APPLICATION\_DETAILS\_SETTINGS  intent.data = Uri.fromParts("package", "com.example.fusedapidemo", null)  intent.flags = Intent.FLAG\_ACTIVITY\_NEW\_TASK  startActivity(intent)  }.show()  }  super.onRequestPermissionsResult(requestCode, permissions, grantResults)  }  private fun appHasPermission(permission: String): Boolean {  if (permission == Manifest.permission.ACCESS\_BACKGROUND\_LOCATION  && Build.VERSION.SDK\_INT > Build.VERSION\_CODES.Q  ) {  return true  }  return ActivityCompat.checkSelfPermission(this, permission) == PackageManager.PERMISSION\_GRANTED  }  companion object {  private const val REQ\_CODE = 100  }  } | </end> |
| <hitle> | RxJavaDemo1 DogApi MVVM | <chare> | 1 | <pext> | 03-09/RxJavaDemo1  ------------Constants----------------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  -----------ApiService--------------------  interface ApiService {  @GET(END\_POINT)  fun getRandomDog(): Single<DogResponse>  }  ------------DogResponse-----------------  data class DogResponse(  val message: String,  val status: String  )  -------------RetrofitBuilder--------------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if(!this::retrofit.isInitialized) {  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .addCallAdapterFactory(RxJava2CallAdapterFactory.create())  .build()  }  return retrofit  }  }  --------------DogViewModel---------------  class DogViewModel: ViewModel() {  val dogResponse = MutableLiveData<DogResponse>()  val error = MutableLiveData<String>()  private lateinit var retrofit: Retrofit  private lateinit var apiService: ApiService  fun getRandomDog() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  val result = apiService.getRandomDog()  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .subscribe({  dogResponse.postValue(it)  }, {  error.postValue(it.message)  })  }  }  -----------------MainActivity--------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: DogViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpviewModel()  }  private fun setUpviewModel() {  viewModel = ViewModelProvider(this)[DogViewModel::class.java]  viewModel.dogResponse.observe(this) {  Picasso.get()  .load(it.message)  .into(binding.imageView)  }  viewModel.error.observe(this) {  Toast.makeText(this, it.toString(), Toast.LENGTH\_SHORT).show()  }  binding.button.setOnClickListener {  viewModel.getRandomDog()  }  }  }  -----------------activity\_main.xml--------------------------  imageView, button | </end> |
| <hitle> | RxJavaDemo2 All Observables | <chare> | 1 | <pext> | 03-09/RxJavaDemo2  Observable.just("Apple", "Mango", "Orange")  .subscribe({  next -> println("Latest item is $next")  }, {  error -> println("Error is $error")  }, {  println("Task got completed")  })  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Latest item is Apple  Latest item is Mango  Latest item is Orange  Task got completed  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Single.just("This block is containting all result")  .subscribe({  result -> println("Task got completed and result is $result")  }, {  error -> println("Error is $error")  })  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Task got completed and result is This block is containting all result  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.just(200, 300, 400, 500, 11, 22, 44, 55, 888)  .filter { item -> item < 100 }  .subscribe { result -> println("Selected items are $result") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Selected items are 11  Selected items are 22  Selected items are 44  Selected items are 55  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.zip(  Observable.just("Android", "IOS", "Xamrin", "Flutter"),  Observable.just("Lolipop", "Kitkat", "Oreo", "Pie"),  BiFunction<String, String, String>{ OS, Version -> "$OS are $Version"}  ).subscribe { res -> println("$res") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Android are Lolipop  IOS are Kitkat  Xamrin are Oreo  Flutter are Pie  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  val source1 = Observable.just("Japan", "Hongkong", "USA")  val source2 = Observable.just("Japanese", "Chinese", "English")  val source3 = Observable.just("Tokyo", "Kowloon", "Washington DC")  Observable.concat(source1, source2, source3)  .subscribe { result -> println("$result")}  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Japan  Hongkong  USA  Japanese  Chinese  English  Tokyo  Kowloon  Washington DC  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.merge(  Observable.fromArray(listOf("Hello", "world")),  Observable.fromArray(listOf("I love", "RxJava"))  ).subscribe { result -> println("$result") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  [Hello, world]  [I love, RxJava]  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Maybe.just("Example of Maybe Observable")  .subscribe(  { result -> println("result is $result") },  { error -> println("error is $error") },  { println("Completed the task") }  )  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  result is Example of Maybe Observable  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Completable.create { completableEmitter ->  completableEmitter.onComplete()  completableEmitter.onError(Exception("something wrong!!"))  }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Flowable.just("Example of Flowable Observable")  .subscribe(  { result -> println("result is $result") },  { error -> println("error is $error") },  { println("Completed the task") }  )  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  result is Example of Flowable Observable  Completed the task  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.just("Titanic", "Conjuring", "Cars")  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .map { it -> "$it is old movie" }  .subscribe(  { item -> Log.i("tag","$item") }, // This is onNext Block  { error -> Log.i("tag","Error is $error") }, // This is onError Block  { Log.i("tag","Task got completed") } // This is onComplete Block  )  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/tag: Titanic is old movie  I/tag: Conjuring is old movie  I/tag: Cars is old movie  I/tag: Task got completed  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.just(1, 2, 3, 4)  .flatMap { item -> Observable.just(item + 2) }  .subscribeOn(Schedulers.io())  .subscribe { values -> Log.i("tag","$values") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/tag: 3  I/tag: 4  I/tag: 5  I/tag: 6  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.merge(  Observable.interval(250, TimeUnit.MILLISECONDS).map { i -> "JAVA" },  Observable.interval(250, TimeUnit.MILLISECONDS).map { i -> "Kotlin" })  .take(10)  .subscribe { result -> Log.i("tag","$result") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/tag: JAVA  I/tag: Kotlin  I/tag: JAVA  I/tag: Kotlin  I/tag: JAVA  I/tag: Kotlin  I/tag: JAVA  I/tag: Kotlin  I/tag: JAVA  I/tag: Kotlin  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.just("Swimming,", "Playing", "Boating", "Fishing")  .take(2)  .subscribe { result -> Log.i("tag", "Your main 2 hobbies are $result") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/tag: Your main 2 hobbies are Swimming,  I/tag: Your main 2 hobbies are Playing  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Log.i("tag","Which team wins the repeated world cup in Football")  Observable.just("France", "Argentina", "England", "USA")  .repeat(2)  .subscribe { result -> Log.i("tag", "$result") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/tag: Which team wins the repeated world cup in Football  I/tag: France  I/tag: Argentina  I/tag: England  I/tag: USA  I/tag: France  I/tag: Argentina  I/tag: England  I/tag: USA  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.range(1, 50)  .buffer(8)  .subscribe { println("Result: $it") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/System.out: Result: [1, 2, 3, 4, 5, 6, 7, 8]  I/System.out: Result: [9, 10, 11, 12, 13, 14, 15, 16]  I/System.out: Result: [17, 18, 19, 20, 21, 22, 23, 24]  I/System.out: Result: [25, 26, 27, 28, 29, 30, 31, 32]  I/System.out: Result: [33, 34, 35, 36, 37, 38, 39, 40]  I/System.out: Result: [41, 42, 43, 44, 45, 46, 47, 48]  I/System.out: Result: [49, 50]  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Observable.range(1, 50)  .buffer(5) // range from initial to max [1,2,3,4,5] = creating a set of total buffer size  .take(4) // out of total result, get up to the count into the take input  .subscribe { println("Result: $it") }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/System.out: Result: [1, 2, 3, 4, 5]  I/System.out: Result: [6, 7, 8, 9, 10]  I/System.out: Result: [11, 12, 13, 14, 15]  I/System.out: Result: [16, 17, 18, 19, 20]  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* | </end> |
| <hitle> | Cold Observable RxJavaDemo3 | <chare> | 1 | <pext> | 03-09/RxJavaDemo3  val source: Observable<String> = Observable.just(  "Spider man",  "John wick",  "Ant man",  "Titanic",  "FF7",  "James bond series 007",  "Jumangi"  )  source.map { it }  .subscribe { println("Luan is $it") }  Thread.sleep(3000)  source.map { it }  .subscribe { println("Alex is $it") }  Thread.sleep(3000)  source.map { it }  .subscribe { println("Thomas is $it") }  Thread.sleep(3000)  source.map { it }  .subscribe { println("Josh is $it") }  Thread.sleep(3000)  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Luan is Spider man  Luan is John wick  Luan is Ant man  Luan is Titanic  Luan is FF7  Luan is James bond series 007  Luan is Jumangi  (After 3 seconds…)  Alex is Spider man  Alex is John wick  Alex is Ant man  Alex is Titanic  Alex is FF7  Alex is James bond series 007  Alex is Jumangi  (After 3 seconds…)  Thomas is Spider man  Thomas is John wick  Thomas is Ant man  Thomas is Titanic  Thomas is FF7  Thomas is James bond series 007  Thomas is Jumangi  (After 3 seconds…)  Josh is Spider man  Josh is John wick  Josh is Ant man  Josh is Titanic  Josh is FF7  Josh is James bond series 007  Josh is Jumangi | </end> |
| <hitle> | Hot Observable RxJavaDemo3 | <chare> | 1 | <pext> | 03-09/RxJavaDemo3  val myObservable = Observable.interval(1, TimeUnit.SECONDS)  val hotObservable = myObservable.publish().refCount()  val subscription1 = hotObservable  .doOnSubscribe { Log.i("tag", "Observer 1 subscribed") }  .doFinally { Log.i("tag", "Observer 1 unsubscribed") }  .subscribe { Log.i("tag", "Observer 1 is $it") }  Thread.sleep(3000)  val subscription2 = hotObservable  .doOnSubscribe { Log.i("tag", "Observer 2 subscribed") }  .doFinally { Log.i("tag", "Observer 2 unsubscribed") }  .subscribe { Log.i("tag", "Observer 2 is $it") }  Thread.sleep(3000)  subscription1.dispose()  val subscription3 = hotObservable  .doOnSubscribe { Log.i("tag", "Observer 3 subscribed") }  .doFinally { Log.i("tag", "Observer 3 unsubscribed") }  .subscribe { Log.i("tag", "Observer 3 is $it") }  Thread.sleep(3000)  subscription2.dispose()  subscription3.dispose()  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  I/tag: Observer 1 subscribed  I/tag: Observer 1 is 0  I/tag: Observer 1 is 1  I/tag: Observer 1 is 2  I/tag: Observer 2 subscribed  I/tag: Observer 1 is 3  I/tag: Observer 2 is 3  I/tag: Observer 1 is 4  I/tag: Observer 2 is 4  I/tag: Observer 1 is 5  I/tag: Observer 2 is 5  I/tag: Observer 1 unsubscribed  I/tag: Observer 3 subscribed  I/tag: Observer 2 is 6  I/tag: Observer 3 is 6  I/tag: Observer 2 is 7  I/tag: Observer 3 is 7  I/tag: Observer 2 unsubscribed  I/tag: Observer 3 unsubscribed | </end> |
| <hitle> | TMDBRxJavaSearch Debounce | <chare> | 1 | <pext> | 03-10/TMDBRxJavaSearch Debounce  -----------Gradle-----------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'kotlin-kapt'  id 'kotlin-parcelize'  }  android {  namespace 'com.example.tmdbmovieapp'  compileSdk 33  defaultConfig {  applicationId "com.example.tmdbmovieapp"  minSdk 26  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures{  viewBinding true  dataBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  //Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  //Convertor factory by Gson  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso  implementation 'com.squareup.picasso:picasso:2.71828'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  implementation "io.reactivex.rxjava2:rxjava:2.2.7"  implementation "io.reactivex.rxjava2:rxandroid:2.1.1"  //RxJava2 with Retrofit  implementation "com.squareup.retrofit2:adapter-rxjava2:2.9.0"  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso for Image Loading  implementation 'com.squareup.picasso:picasso:2.71828'  }  -----------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".view.activity.MainActivity">  <androidx.fragment.app.FragmentContainerView  android:id="@+id/dashboardFragment"  android:name="com.example.tmdbmovieapp.view.fragment.DashboardFragment"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  tools:context=".view.fragment.DashboardFragment" />  </androidx.constraintlayout.widget.ConstraintLayout>  -----------component\_moviedetail.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  </androidx.cardview.widget.CardView>  -----------fragment\_dashboard.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <layout 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"  tools:context=".view.fragment.DashboardFragment">  <data>  <variable  name="dashboardFragment"  type="com.example.tmdbmovieapp.view.fragment.DashboardFragment" />  </data>  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  >  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recyclerViewSearchResult"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:visibility="gone"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <LinearLayout  android:id="@+id/layoutTabs"  android:layout\_width="wrap\_content"  android:layout\_height="match\_parent"  android:orientation="vertical"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <com.google.android.material.tabs.TabLayout  android:id="@+id/tabLayout"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content" />  <androidx.viewpager2.widget.ViewPager2  android:id="@+id/dashboardViewPager"  fragment="@{dashboardFragment}"  setupWithTab="@{tabLayout}"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_weight="1" />  </LinearLayout>  </androidx.constraintlayout.widget.ConstraintLayout>  </layout>  -----------fragment\_latest.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <layout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <data>  <variable  name="vm"  type="com.example.tmdbmovieapp.viewmodel.MovieListViewModel" />  </data>  <FrameLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="#283593"  tools:context=".view.fragment.LatestFragment">  <TextView  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:textColor="@color/white"  android:textStyle="italic"  android:textSize="40sp"  android:text='@{vm.latestMovie.title}'  tools:text="N/A"/>  </FrameLayout>  </layout>  -----------fragment\_popular.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=".view.fragment.PopularFragment">  </FrameLayout>  -----------fragment\_trending.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <layout 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">  <FrameLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".view.fragment.TrendingFragment">  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:elevation="10dp"  android:padding="10dp"  app:cardBackgroundColor="#FFFFFF"  app:cardCornerRadius="30dp">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvUpMovies"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp" />  </androidx.cardview.widget.CardView>  </FrameLayout>  </layout>  -----------fragment\_upcoming.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <layout 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">  <FrameLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".view.fragment.UpcomingFragment">  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:elevation="10dp"  android:padding="10dp"  app:cardBackgroundColor="#FFFFFF"  app:cardCornerRadius="30dp">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvUpMovies"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp" />  </androidx.cardview.widget.CardView>  </FrameLayout>  </layout>  -----------upcoming\_movie\_item.xml-----------  <?xml version="1.0" encoding="utf-8"?>  <layout 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">  <data>  <variable  name="movie"  type="com.example.tmdbmovieapp.model.local.data.Movie" />  </data>  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="8dp"  android:elevation="15dp"  app:cardBackgroundColor="#E7E9EC"  app:cardCornerRadius="20dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgNews"  android:layout\_width="match\_parent"  android:layout\_height="230dp"  android:scaleType="centerCrop"  remoteSourceImage='@{movie.posterPath}'  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toTopOf="@+id/txtTitle" />  <TextView  android:id="@+id/txtRating"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="15dp"  android:background="@drawable/baseline\_circle\_24"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:textColor="@color/white"  android:textSize="16sp"  android:textStyle="italic"  android:text='@{Double.toString(movie.voteAverage)}'  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintBottom\_toTopOf="@+id/txtTitle"  tools:text="" />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="5dp"  android:maxLines="2"  android:text='@{movie.title}'  android:textColor="@color/black"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/imgNews"  tools:text="" />  <TextView  android:id="@+id/txtDate"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="10dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  app:layout\_constraintBottom\_toBottomOf="parent"  tools:text='@{movie.releaseDate}' />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  </layout>  -----------menu/main.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/action\_search"  android:icon="@drawable/baseline\_search\_24"  android:title="Search"  app:actionViewClass="android.widget.SearchView"  app:showAsAction="always|collapseActionView"  />  </menu>  -----------model/local/data/-----------  @Entity(tableName = Constant.TABLE\_MOVIE)  data class Movie(  val adult: Boolean,  val backdropPath: String?,  val genreIds: List<Int>?,  @PrimaryKey val id: Int,  val originalLanguage: String,  val originalTitle: String,  val overview: String,  val popularity: Double,  val posterPath: String?,  val releaseDate: String,  val title: String,  val video: Boolean,  val voteAverage: Double,  val voteCount: Int,  val isUpcoming: Boolean? = false,  val isTopRated: Boolean? = false,  val isSearched: Boolean? = false,  )  @Entity(tableName = Constant.TABLE\_MOVIE\_DETAIL)  data class MovieDetail(  val adult: Boolean,  val backdropPath: String?,  val budget: Int,  val genres: List<Int>,  val homepage: String?,  @PrimaryKey  val id: Int,  val imdbId: String?,  val originalLanguage: String,  val originalTitle: String,  val overview: String?,  val popularity: Double,  val posterPath: String?,  val productionCompanies: List<Int>,  val productionCountries: List<String>,  val releaseDate: Long,  val revenue: Int,  val runtime: Int?,  val spokenLanguages: List<String>,  val status: String,  val tagline: String?,  val title: String,  val video: Boolean,  val voteAverage: Double,  val voteCount: Int  )  @Entity(tableName = Constant.TABLE\_MOVIE\_GENRE)  data class MovieGenre(  @PrimaryKey  val id: Int,  val name: String  )  @Entity(tableName = Constant.TABLE\_PRODUCTION\_COMPANY)  data class ProductionCompany(  @PrimaryKey  @SerializedName("id")  val id: Int,  @SerializedName("logo\_path")  val logoPath: String?,  @SerializedName("name")  val name: String,  @SerializedName("origin\_country")  val originCountry: String  )  @Entity(tableName = Constant.TABLE\_PRODUCTION\_COUNTRY)  data class ProductionCountry(  @PrimaryKey  @SerializedName("iso\_3166\_1")  val iso31661: String,  @SerializedName("name")  val name: String  )  @Entity(tableName = Constant.TABLE\_SPOKEN\_LANGUAGE)  data class SpokenLanguage(  @PrimaryKey  @SerializedName("iso\_639\_1")  val iso6391: String,  @SerializedName("name")  val name: String  )  -----------model/local/AppDatabase-----------  @Database(  entities = [  Movie::class,  MovieDetail::class,  MovieGenre::class,  ProductionCompany::class,  ProductionCountry::class,  SpokenLanguage::class  ], version = 1, exportSchema = false  )  @androidx.room.TypeConverters(TypeConverters::class)  abstract class AppDatabase : RoomDatabase() {  abstract fun getMovieDao(): MovieDao  abstract fun getMoviesDao(): MoviesDao  companion object {  private var INSTANCE: AppDatabase? = null  fun getInstance(context: Context): AppDatabase {  if (INSTANCE == null) {  INSTANCE = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "movieDB"  ).allowMainThreadQueries().fallbackToDestructiveMigration().build()  }  return INSTANCE as AppDatabase  }  }  }  -----------model/local/MovieDao-----------  @Dao  interface MovieDao {  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun saveLatestMovie(data: Movie): Long  @Query("SELECT \* FROM ${Constant.TABLE\_MOVIE}")  fun getLatestMovie(): LiveData<List<Movie>>  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun saveMovieDetail(data: List<MovieDetail>): List<Long>  @Query("select \* from ${Constant.TABLE\_MOVIE\_DETAIL} where id = :movieId")  fun getMovieDetailById(movieId: Int): LiveData<MovieDetail>  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun saveMovieGenre(data: List<MovieGenre>)  @Query("select \* from ${Constant.TABLE\_MOVIE\_GENRE} where id = :genreId")  fun getMovieGenreById(genreId: Int): List<MovieGenre>  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun saveProductionCompany(data: List<ProductionCompany>)  @Query("select \* from ${Constant.TABLE\_PRODUCTION\_COMPANY} where id = :companyId")  fun getProductionCompanyById(companyId: Int): List<ProductionCompany>  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun saveProductionCountry(data: List<ProductionCountry>)  @Query("select \* from ${Constant.TABLE\_PRODUCTION\_COUNTRY} where iso31661 = :countryIso")  fun getProductionCountryById(countryIso: Int): List<ProductionCountry>  @Insert(onConflict = OnConflictStrategy.REPLACE)  fun saveSpokenLanguage(data: List<SpokenLanguage>)  @Query("select \* from ${Constant.TABLE\_SPOKEN\_LANGUAGE} where iso6391 = :languageIso")  fun getSpokenLanguageById(languageIso: Int): List<SpokenLanguage>  @Query("SELECT \* FROM Movie")  fun getNews(): LiveData<List<Movie>>  }  -----------model/local/MoviesDao-----------  @Dao  interface MoviesDao {  @Insert(onConflict = OnConflictStrategy.IGNORE)  fun saveMovies(movies: List<Movie>): List<Long>  @Query("SELECT \* FROM ${Constant.TABLE\_MOVIE} WHERE isUpcoming = 1")  fun getUpcomingMovies(): LiveData<List<Movie>>  @Query("SELECT \* FROM ${Constant.TABLE\_MOVIE} WHERE isTopRated = 1")  fun getTopRatedMovies(): LiveData<List<Movie>>  @Query("SELECT \* FROM ${Constant.TABLE\_MOVIE} WHERE isSearched = 1")  fun getSearchedMovies(): LiveData<List<Movie>>  }  -----------model/local/TypeConverters-----------  class TypeConverters {  @TypeConverter fun fromMovieGenresList(value: List<MovieGenre>) = Gson().toJson(value)  @TypeConverter fun toMovieGenresList(value: String) = Gson().fromJsonList<MovieGenre>(value)  @TypeConverter fun fromIntList(value: List<Int>) = Gson().toJson(value)  @TypeConverter fun toIntList(value: String) = Gson().fromJsonList<Int>(value)  @TypeConverter fun fromStringList(value: List<String>) = Gson().toJson(value)  @TypeConverter fun toStringList(value: String) = Gson().fromJsonList<String>(value)  inline fun <reified T> Gson.fromJsonList(value: String) =  fromJson<List<T>>(value, object: TypeToken<List<T>>() {}.type).toList()  }  -----------model/remote/data/-----------  @Entity(tableName = "UpcomingMovie")  data class MoviesListResponse(  @SerializedName("id")  @PrimaryKey  val id: Int,  // @SerializedName("dates")  // val dates: Dates,  @SerializedName("page")  val page: Int,  @SerializedName("results")  val results: List<MovieResponse>,  @SerializedName("total\_pages")  val total\_pages: Int,  @SerializedName("total\_results")  val total\_results: Int  )  data class Result(  val adult: Boolean,  val backdrop\_path: String,  val genre\_ids: List<Int>,  val id: Int,  val original\_language: String,  val original\_title: String,  val overview: String,  val popularity: Double,  val poster\_path: String,  val release\_date: String,  val title: String,  val video: Boolean,  val vote\_average: Double,  val vote\_count: Int  )  data class MovieDetailResponse(  @SerializedName("adult")  val adult: Boolean,  @SerializedName("backdrop\_path")  val backdropPath: String?,  @SerializedName("budget")  val budget: Int,  @SerializedName("genres")  val genres: List<MovieGenre>,  @SerializedName("homepage")  val homepage: String?,  @SerializedName("id")  val id: Int,  @SerializedName("imdb\_id")  val imdbId: String?,  @SerializedName("original\_language")  val originalLanguage: String,  @SerializedName("original\_title")  val originalTitle: String,  @SerializedName("overview")  val overview: String?,  @SerializedName("popularity")  val popularity: Double,  @SerializedName("poster\_path")  val posterPath: String?,  @SerializedName("production\_companies")  val productionCompanies: List<ProductionCompany>,  @SerializedName("production\_countries")  val productionCountries: List<ProductionCountry>,  @SerializedName("release\_date")  val releaseDate: String,  @SerializedName("revenue")  val revenue: Int,  @SerializedName("runtime")  val runtime: Int?,  @SerializedName("spoken\_languages")  val spokenLanguages: List<SpokenLanguage>,  @SerializedName("status")  val status: String,  @SerializedName("tagline")  val tagline: String?,  @SerializedName("title")  val title: String,  @SerializedName("video")  val video: Boolean,  @SerializedName("vote\_average")  val voteAverage: Double,  @SerializedName("vote\_count")  val voteCount: Int  ) {  fun toLocal(dao: MovieDao) = MovieDetail(  adult = adult,  backdropPath = backdropPath,  budget = budget,  genres = genres  .apply { dao.saveMovieGenre(this) }  .map { it.id },  homepage = homepage,  id = id,  imdbId = imdbId,  originalLanguage = originalLanguage,  originalTitle = originalTitle,  overview = overview,  popularity = popularity,  posterPath = posterPath,  productionCompanies = productionCompanies  .apply { dao.saveProductionCompany(this) }  .map { it.id },  productionCountries = productionCountries  .apply { dao.saveProductionCountry(this) }  .map { it.iso31661 },  releaseDate = LocalDate.parse(releaseDate).toEpochDay() ,  revenue = revenue,  runtime = runtime,  spokenLanguages = spokenLanguages  .apply { dao.saveSpokenLanguage(this) }  .map { it.iso6391 },  status = status,  tagline = tagline,  title = title,  video = video,  voteAverage = voteAverage,  voteCount = voteCount  )  }  data class MovieResponse(  @SerializedName("poster\_path")  val posterPath: String?,  @SerializedName("adult")  val adult: Boolean,  @SerializedName("overview")  val overview: String,  @SerializedName("release\_date")  val releaseDate: String,  @SerializedName("genre\_ids")  val genreIds: List<Int>?,  @SerializedName("id")  val id: Int,  @SerializedName("original\_title")  val originalTitle: String,  @SerializedName("original\_language")  val originalLanguage: String,  @SerializedName("title")  val title: String,  @SerializedName("backdrop\_path")  val backdropPath: String?,  @SerializedName("popularity")  val popularity: Double,  @SerializedName("vote\_count")  val voteCount: Int,  @SerializedName("video")  val video: Boolean,  @SerializedName("vote\_average")  val voteAverage: Double  ) {  fun toLocal(isUpcoming: Boolean = false, isTopRated: Boolean = false, isSearch: Boolean) = Movie(  adult = adult,  backdropPath = backdropPath,  genreIds = genreIds ?: listOf(),  id = id,  originalLanguage = originalLanguage,  originalTitle = originalTitle,  overview = overview,  popularity = popularity,  posterPath = posterPath,  releaseDate = releaseDate,  title = title,  video = video,  voteAverage = voteAverage,  voteCount = voteCount,  isUpcoming = isUpcoming,  isTopRated = isTopRated,  isSearched = isSearch  )  }  -----------model/remote/ApiService-----------  interface ApiService {  @GET(Constant.END\_POINT\_MOVIE\_DETAIL)  fun getMovieDetail(  @Path(Constant.END\_POINT\_MOVIE\_DETAIL\_ARG\_1) movieId: Int  ): Single<MovieDetailResponse>  @GET(END\_POINT\_UPCOMING\_MOVIES)  fun getUpComingMovies(  @Query("country") country : String? = null,  @Query("page") page : Int? = null  ): Single<MoviesListResponse>  @GET(END\_POINT\_TOP\_RATED\_MOVIES)  fun getTopRatedMovies(  @Query("country") country : String? = null,  @Query("page") page : Int? = null  ): Single<MoviesListResponse>  @GET(Constant.END\_POINT\_LATEST\_MOVIES)  fun getLatestMovie(): Single<MovieResponse>  @GET(END\_POINT\_SEARCH)  fun getMovieSearch(  @Query("query") searchText: String  ): Single<MoviesListResponse>  }  -----------model/remote/Constant-----------  object Constant {  const val BASE\_URL = "https://api.themoviedb.org/3/"  const val BASE\_IMAGE\_URL = "https://image.tmdb.org/t/p/original/"  const val AUTHORIZATION = "Authorization"  const val TOKEN\_MASTER = "a53a3bcb8e2f8efe6d2988a34d73e750"  const val TOKEN\_JOSH = "441f9a2f476e2d801a859a9955fe6188"  const val TOKEN\_ALEX = "a51ff8a6730be4f36331b0b345e9e775"  const val TOKEN\_THOMAS = ""  const val TOKEN\_LUAN = ""  const val END\_POINT\_LATEST\_MOVIES = "movie/latest"  const val END\_POINT\_POPULAR\_MOVIES = ""  const val END\_POINT\_UPCOMING\_MOVIES = "movie/upcoming"  const val END\_POINT\_TOP\_RATED\_MOVIES = "movie/top\_rated"  const val END\_POINT\_MOVIE\_DETAIL = "movie/{movie\_id}"  const val END\_POINT\_MOVIE\_DETAIL\_ARG\_1 = "movie\_id"  const val END\_POINT\_SEARCH = "search/movie"  const val TABLE\_MOVIE = "movie"  const val TABLE\_MOVIE\_DETAIL = "movieDetail"  const val TABLE\_MOVIE\_GENRE = "movieGenre"  const val TABLE\_PRODUCTION\_COMPANY = "productionCompany"  const val TABLE\_PRODUCTION\_COUNTRY = "productionCountry"  const val TABLE\_SPOKEN\_LANGUAGE = "spokenLanguage"  }  -----------model/remote/OkHttpInterceptors-----------  object OkHttpInterceptors {  class APIKey(  private val apiKey: String  ) : Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentUrl = chain.request().url  val newUrl = currentUrl.newBuilder().addQueryParameter("api\_key", apiKey).build()  val currentRequest = chain.request().newBuilder()  val newRequest = currentRequest.url(newUrl).build()  return chain.proceed(newRequest)  }  }  }  -----------model/remote/RetrofitBuilder-----------  object RetrofitBuilder {  val instanceLatestMovie: ApiService by lazy {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(OkHttpInterceptors.APIKey(TOKEN\_ALEX))  .addInterceptor(loggingInterceptor)  .build()  Retrofit.Builder()  .baseUrl(BASE\_URL)  .client(client)  .addConverterFactory(GsonConverterFactory.create())  .addCallAdapterFactory(RxJava2CallAdapterFactory.createAsync())  .client(  OkHttpClient.Builder()  .addInterceptor(OkHttpInterceptors.APIKey(TOKEN\_JOSH))  .addInterceptor(loggingInterceptor)  .build()  )  .build()  .create(ApiService::class.java)  }  }  -----------model/remote/IRepository-----------  interface IRepository {  fun getMovieDetail(movieId: Int)  fun getLatestMovie()  val isProcessing: MutableLiveData<Boolean>  fun getUpComingMovie()  fun getTopRatedMovie()  fun searchMovie(movieName: String)  }  -----------model/remote/LocalRepository-----------  class LocalRepository(private val appDatabase: AppDatabase) {  fun getLatestMovie() = appDatabase.getMovieDao().getLatestMovie()  fun saveLatestMovie(movie: Movie) = appDatabase.getMovieDao().saveLatestMovie(movie)  fun saveMovieDetail(data: List<MovieDetail>) = appDatabase.getMovieDao().saveMovieDetail(data)  fun getMovieDetailById(movieId: Int) = appDatabase.getMovieDao().getMovieDetailById(movieId)  fun getUpComingMovies() = appDatabase.getMoviesDao().getUpcomingMovies()  fun saveUpComingMovies(movies: List<Movie>) = appDatabase.getMoviesDao().saveMovies(movies)  fun getTopRatedMovies() = appDatabase.getMoviesDao().getTopRatedMovies()  fun saveTopRatedMovies(movies: List<Movie>) = appDatabase.getMoviesDao().saveMovies(movies)  fun getSearchMovies() = appDatabase.getMoviesDao().getSearchedMovies()  fun saveAsSearchMovies(movies: List<Movie>) = appDatabase.getMoviesDao().saveMovies(movies)  }  -----------model/remote/RemoteRepository-----------  class RemoteRepository(  private val apiService: ApiService = RetrofitBuilder.instanceLatestMovie  ) {  fun getLatestMovie() = apiService.getLatestMovie()  fun getMovieDetail(movieId: Int) = apiService.getMovieDetail(movieId)  fun loadUpcomingMovies() = apiService.getUpComingMovies()  fun loadTopRatedMovies() = apiService.getTopRatedMovies()  fun searchMovie(searchText: String) = apiService.getMovieSearch(searchText)  }  -----------model/remote/Repository-----------  class Repository(  private val appDatabase: AppDatabase,  private val localRepository: LocalRepository = LocalRepository(appDatabase),  private val remoteRepository: RemoteRepository = RemoteRepository()  ) : IRepository {  private val \_latestMovie = MutableLiveData<MovieResponse>()  val latestMovie: LiveData<MovieResponse> get() = \_latestMovie  val movieDetail = localRepository.getMovieDetailById(-1)  val upComingMovie = localRepository.getUpComingMovies()  val topRatedMovie = localRepository.getTopRatedMovies()  val searchedMovies = localRepository.getSearchMovies()  var compositeDisposable: CompositeDisposable = CompositeDisposable()  override fun getMovieDetail(movieId: Int) {  val disposable = remoteRepository.getMovieDetail(movieId)  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe ({ res ->  localRepository.saveMovieDetail(  listOf(res.toLocal(appDatabase.getMovieDao()))  )  },{  Log.i("error", it.message.toString())  })  compositeDisposable.add(disposable)  }  override fun getLatestMovie() {  val disposable = remoteRepository.getLatestMovie()  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe ({ res ->  \_latestMovie.postValue(res)  },{  Log.i("error", it.message.toString())  })  compositeDisposable.add(disposable)  }  override val isProcessing = MutableLiveData<Boolean>()  //override fun getUpComingMovie() = remoteRepository.loadUpcomingMovies()  override fun getUpComingMovie() {  val disposable = remoteRepository.loadUpcomingMovies()  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe ({ res ->  localRepository.saveUpComingMovies(  (res.results.map { result ->  result.toLocal(isUpcoming = true, isTopRated = false, isSearch = false)  })  )  },{  Log.i("error", it.message.toString())  })  compositeDisposable.add(disposable)  }  override fun getTopRatedMovie() {  val disposable = remoteRepository.loadTopRatedMovies()  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe ({ res ->  localRepository.saveTopRatedMovies(  (res.results.map { result ->  result.toLocal(isUpcoming = false, isTopRated = true, isSearch = false)  })  )  },{  Log.i("error", it.message.toString())  })  compositeDisposable.add(disposable)  }  override fun searchMovie(movieName: String) {  val disposable = remoteRepository.searchMovie(movieName)  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe ({ res ->  localRepository.saveAsSearchMovies(  (res.results.map { result ->  result.toLocal(isUpcoming = false, isTopRated = false, isSearch = true)  })  )  },{  Log.i("error", it.message.toString())  })  compositeDisposable.add(disposable)  }  }  -----------viewmodel/MovieListViewModel-----------  class MovieListViewModel(  application: Application,  private val db: AppDatabase = AppDatabase.getInstance(application),  private val repository: Repository = Repository(db)  ) : AndroidViewModel(application), Observable {  val latestMovie = repository.latestMovie  val movieDetail = repository.movieDetail  val upComingMovies = repository.upComingMovie  val topRatedMovies = repository.topRatedMovie  val searchedMovieLiveData = repository.searchedMovies  fun getUpcomingMovies() = upComingMovies.also { repository.getUpComingMovie() }  fun getTrendingMovies() = topRatedMovies.also { repository.getTopRatedMovie() }  fun getLatestMovies() = latestMovie.also { repository.getLatestMovie() }  fun getSearchedMovies(searchText: String) = searchedMovieLiveData.also { repository.searchMovie(searchText) }  fun getMovieDetail(movieId: Int) = movieDetail.also { repository.getMovieDetail(movieId) }  private val callbacks: PropertyChangeRegistry = PropertyChangeRegistry()  override fun addOnPropertyChangedCallback(callback: Observable.OnPropertyChangedCallback) {  callbacks.add(callback)  }  override fun removeOnPropertyChangedCallback(callback: Observable.OnPropertyChangedCallback) {  callbacks.remove(callback)  }  fun notifyChange() {  callbacks.notifyCallbacks(this, 0, null)  }  override fun onCleared() {  super.onCleared()  repository.compositeDisposable.dispose()  }  }  -----------viewmodel/ViewModelFactory-----------  fun <T : ViewModel> T.createFactory(): ViewModelProvider.Factory {  val viewModel = this  return object : ViewModelProvider.Factory {  @Suppress("UNCHECKED\_CAST")  override fun <T : ViewModel> create(modelClass: Class<T>): T = viewModel as T  }  }  -----------view.MainActivity-----------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var MovieListViewModel : MovieListViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  }  }  -----------view/CommonBindingAdapters-----------  class CommonBindingAdapters {  companion object {  @JvmStatic  @BindingAdapter("remoteSourceImage")  fun loadImageFromServer(imageView: ImageView, url: String?) = with(url) {  val path = "$BASE\_IMAGE\_URL$this"  Picasso  .get()  .load(path)  .error(R.drawable.baseline\_error\_24)  .placeholder(R.drawable.ic\_launcher\_background)  .into(imageView)  }  }  }  -----------view/DashboardViewPagerAdapter  class DashboardViewPagerAdapter(  fragment: Fragment  ): FragmentStateAdapter(fragment) {  override fun getItemCount(): Int = pageNames.size  override fun createFragment(position: Int): Fragment = when (position) {  0 -> LatestFragment()  1 -> UpcomingFragment()  2 -> TrendingFragment()  3 -> PopularFragment()  else -> LatestFragment()  }  companion object {  val pageNames = listOf(  "Latest Movies",  "Uncoming Movies",  "Trending Movies",  "Popular Movies"  )  }  }  -----------view/UpcomingRVAdapter-----------  class UpcomingRVAdapter(private val movieList: List<Movie>) :  RecyclerView.Adapter<UpcomingRVAdapter.UpcomingViewHolder>() {  private lateinit var upcomingMovieItemBinding: UpcomingMovieItemBinding  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): UpcomingViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  upcomingMovieItemBinding =  DataBindingUtil.inflate(layoutInflater, R.layout.upcoming\_movie\_item, parent, false)  return UpcomingViewHolder(upcomingMovieItemBinding)  }  override fun onBindViewHolder(holder: UpcomingViewHolder, position: Int) {  holder.bind(movieList[position])  }  inner class UpcomingViewHolder(binding: UpcomingMovieItemBinding) :  RecyclerView.ViewHolder(binding.root) {  fun bind(movie: Movie) {  upcomingMovieItemBinding.movie = movie  }  }  override fun getItemCount() = movieList.size  }  -----------view/DashboardFragment-----------  class DashboardFragment : Fragment(), MenuItem.OnActionExpandListener {    private lateinit var binding: FragmentDashboardBinding  private val viewModel by lazy {  requireActivity().run {  ViewModelProvider(  requireActivity(), MovieListViewModel(application).createFactory()  )[MovieListViewModel::class.java]  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View = FragmentDashboardBinding.inflate(inflater,container, false).apply{  dashboardFragment = this@DashboardFragment  binding = this  setHasOptionsMenu(true)  }.root  override fun onCreateOptionsMenu(menu: Menu, inflater: MenuInflater) {  super.onCreateOptionsMenu(menu, inflater)  inflater.inflate(R.menu.main, menu)  initSearchView(menu)  setupObserver()  }  private fun setupObserver() {  viewModel.searchedMovieLiveData.observe(viewLifecycleOwner) {  if( it == null ) return@observe  binding.recyclerViewSearchResult.layoutManager = LinearLayoutManager(context)  if(it.isNotEmpty()) {  binding.recyclerViewSearchResult.adapter = UpcomingRVAdapter(it)  }  }  }  private fun initSearchView(menu: Menu) {  val searchItem = menu.findItem(R.id.action\_search)  val searchView = MenuItemCompat.getActionView(searchItem) as SearchView  initializeSearch(searchView)  searchItem.setOnActionExpandListener(this)  }  -------attatch RxSearchObservable to searchView-----------  private fun initializeSearch(searchView: SearchView) {  val disposable = RxSearchObservable.fromView(searchView)  .debounce( 300, TimeUnit.MILLISECONDS)  .filter { it.isNotEmpty() }  .distinctUntilChanged()  .subscribe { viewModel.getSearchedMovies(it) }  }  override fun onStop() {  super.onStop()  }  override fun onMenuItemActionExpand(item: MenuItem): Boolean {  binding.recyclerViewSearchResult.visibility = View.VISIBLE  binding.layoutTabs.visibility = View.GONE  return true  }  override fun onMenuItemActionCollapse(item: MenuItem): Boolean {  binding.recyclerViewSearchResult.visibility = View.GONE  binding.layoutTabs.visibility = View.VISIBLE  return true  }  -----------RxSearchObservable for Debounce-----------  object RxSearchObservable {  fun fromView(searchView: SearchView): Observable<String> {  val subject = PublishSubject.create<String>()  searchView.setOnQueryTextListener(object : SearchView.OnQueryTextListener {  override fun onQueryTextSubmit(s: String): Boolean {  subject.onComplete()  return true  }  override fun onQueryTextChange(text: String): Boolean {  subject.onNext(text)  return true  }  })  return subject  }  }  companion object {  @JvmStatic  @BindingAdapter("setupWithTab", "fragment")  fun setViewPagerAdapter(viewPager: ViewPager2, tabLayout: TabLayout, fragment: Fragment) {  viewPager.adapter = DashboardViewPagerAdapter(fragment)  TabLayoutMediator(tabLayout, viewPager) { tab, position ->  tab.text = DashboardViewPagerAdapter.pageNames[position]  }.attach()  }  }  }  -----------view/LatestFragment-----------  class LatestFragment : Fragment() {  private val viewModel by lazy {  requireActivity().run {  ViewModelProvider(  requireActivity(), MovieListViewModel(application).createFactory()  )[MovieListViewModel::class.java]  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View = DataBindingUtil.inflate<FragmentLatestBinding>(  inflater, R.layout.fragment\_latest, container, false  ).apply {  vm = viewModel  }.root  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  viewModel.latestMovie.observe(viewLifecycleOwner) {  println(it?.title)  viewModel.notifyChange()  }  viewModel.movieDetail.observe(viewLifecycleOwner) {  if (it == null) return@observe  println(it.title)  }  }  override fun onResume() {  super.onResume()  viewModel.getLatestMovies()  }  }  -----------view/PopularFragment-----------  class PopularFragment : Fragment() {  // TODO: Rename and change types of parameters  private var param1: String? = null  private var param2: String? = null  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  arguments?.let {  param1 = it.getString(ARG\_PARAM1)  param2 = it.getString(ARG\_PARAM2)  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  // Inflate the layout for this fragment  return inflater.inflate(R.layout.fragment\_popular, container, false)  }  companion object {  /\*\*  \* Use this factory method to create a new instance of  \* this fragment using the provided parameters.  \*  \* @param param1 Parameter 1.  \* @param param2 Parameter 2.  \* @return A new instance of fragment PopularFragment.  \*/  // TODO: Rename and change types and number of parameters  @JvmStatic  fun newInstance(param1: String, param2: String) =  PopularFragment().apply {  arguments = Bundle().apply {  putString(ARG\_PARAM1, param1)  putString(ARG\_PARAM2, param2)  }  }  }  }  -----------view/TrendingFragment-----------  class TrendingFragment : Fragment() {  private lateinit var binding: FragmentTrendingBinding  private lateinit var db: AppDatabase  private lateinit var movieDao: MoviesDao  private lateinit var rvAdapter: UpcomingRVAdapter  private lateinit var movieList: List<MoviesListResponse>  private val viewModel by lazy {  requireActivity().run {  ViewModelProvider(  requireActivity(), MovieListViewModel(application).createFactory()  )[MovieListViewModel::class.java]  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View = DataBindingUtil.inflate<FragmentTrendingBinding>(  inflater, R.layout.fragment\_trending, container, false  ).apply {  binding = this  binding.rvUpMovies.layoutManager = LinearLayoutManager(context)  }.root  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  viewModel.topRatedMovies.observe(viewLifecycleOwner) {  if (it == null) return@observe  binding.rvUpMovies.adapter = UpcomingRVAdapter(it)  }  }  override fun onResume() {  super.onResume()  viewModel.getTrendingMovies()  }  }  -----------view/UpcomingFragment-----------  class UpcomingFragment : Fragment() {  private lateinit var binding: FragmentUpcomingBinding  private lateinit var db: AppDatabase  private lateinit var movieDao: MoviesDao  private lateinit var rvAdapter: UpcomingRVAdapter  private lateinit var movieList: List<MoviesListResponse>  private val viewModel by lazy {  requireActivity().run {  ViewModelProvider(  requireActivity(), MovieListViewModel(application).createFactory()  )[MovieListViewModel::class.java]  }  }  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View = DataBindingUtil.inflate<FragmentUpcomingBinding>(  inflater, R.layout.fragment\_upcoming, container, false  ).apply {  binding = this  binding.rvUpMovies.layoutManager = LinearLayoutManager(context)  }.root  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  viewModel.upComingMovies.observe(viewLifecycleOwner) {  if (it == null) return@observe  binding.rvUpMovies.adapter = UpcomingRVAdapter(it)  }  }  override fun onResume() {  super.onResume()  viewModel.getUpcomingMovies()  }  } | </end> |
| <hitle> | RxJavaSearchNewsApp Debounce | <chare> | 1 | <pext> | 03-10/RxJavaSearchNewsApp Debounce  --------------Gradle.module--------------  plugins {  id 'kotlin-kapt'  id 'kotlin-parcelize'  }  buildFeatures {  viewBinding true  }  }  dependencies {  //Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  //Convertor factory by Gson  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso  implementation 'com.squareup.picasso:picasso:2.71828'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  //RxJava2  implementation "io.reactivex.rxjava2:rxjava:2.2.7"  implementation "io.reactivex.rxjava2:rxandroid:2.1.1"  //RxJava2 with Retrofit  implementation "com.squareup.retrofit2:adapter-rxjava2:2.9.0"  }  --------------model/local/AppDatabase--------------  @Database(entities = [News::class], version = 1, exportSchema = false)  abstract class AppDatabase : RoomDatabase() {  abstract fun getNewsDao(): NewsDao  companion object {  private var INSTANCE: AppDatabase? = null  fun getInstance(context: Context): AppDatabase {  if (INSTANCE == null) {  INSTANCE = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "newsDB"  ).allowMainThreadQueries().build()  }  return INSTANCE as AppDatabase  }  }  }  --------------model/local/NewsDao--------------  @Dao  interface NewsDao {  @Insert(onConflict = OnConflictStrategy.IGNORE)  fun saveNews(news: List<News>): List<Long>  @Query("SELECT \* FROM News")  fun getNews(): LiveData<List<News>>  }  --------------model/remote/data/News--------------  @Entity(tableName = "News")  data class News(  @ColumnInfo(name = "author")  @SerializedName("author")  val author: String,  @ColumnInfo(name = "description")  @SerializedName("description")  val description: String,  @PrimaryKey(autoGenerate = false)  @ColumnInfo(name = "id")  @SerializedName("id")  val id: String,  @ColumnInfo(name = "image")  @SerializedName("image")  val image: String,  @ColumnInfo(name = "language")  @SerializedName("language")  val language: String,  @ColumnInfo(name = "published")  @SerializedName("published")  val published: String,  @ColumnInfo(name = "title")  @SerializedName("title")  val title: String,  @ColumnInfo(name = "url")  @SerializedName("url")  val url: String  )  --------------model/remote/data/NewsResponse--------------  data class NewsResponse(  @SerializedName("news")  val news: List<News>,  @SerializedName("status")  val status: String  )  --------------model/remote/ApiService--------------  interface ApiService {  @GET(END\_POINT)  fun getLatestNews(): Single<NewsResponse>  @GET(Constant.END\_POINT\_SEARCH)  fun searchNews(  @Query("keywords") keywords: String,  @Query("start\_date") start\_date: String? = null,  @Query("end\_date") end\_date: String? = null,  @Query("category") category: String? = null,  @Query("country") country: String? = null,  @Query("language") language: String? = null  ): Single<NewsResponse>  }  --------------model/remote/AuthInterceptor--------------  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  currentRequest.addHeader(AUTHORIZATION, TOKEN)  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  }  --------------model/remote/Constants--------------  object Constant {  const val BASE\_URL = "https://api.currentsapi.services/v1/"  const val END\_POINT = "latest-news"  const val END\_POINT\_SEARCH = "search"  const val AUTHORIZATION = "Authorization"  const val TOKEN = "ehTUs\_L7VNOevxSsW301L3Y6KhOmJ573Grs-VKu--uPjKPZF"  }  --------------model/remote/RetrofitBuilder--------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(AuthInterceptor())  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .addCallAdapterFactory(RxJava2CallAdapterFactory.createAsync())  .client(client)  .build()  }  return retrofit  }  }  --------------model/repository/IRepository--------------  interface IRepository {  fun getLatestNews(): LiveData<List<News>>  fun updateLatestNews()  fun searchNews(  keywords: String,  start\_date: String? = null,  end\_date: String? = null,  category: String? = null,  country: String? = null,  language: String? = null  ): Disposable  val isProcessing: MutableLiveData<Boolean>  val searchedNews: MutableLiveData<List<News>>  val compositeDisposable: CompositeDisposable  }  --------------model/repository/LocalRepository--------------  class LocalRepository(private val appDatabase: AppDatabase) {  fun getLatestNews() = appDatabase.getNewsDao().getNews()  fun saveNews(news: List<News>) = appDatabase.getNewsDao().saveNews(news)  }  --------------model/repository/RemoteRepository--------------  class RemoteRepository(private val apiService: ApiService) {  fun loadLatestNews() = apiService.getLatestNews()  fun searchNews(  keywords: String,  start\_date: String? = null,  end\_date: String? = null,  category: String? = null,  country: String? = null,  language: String? = null  ) = apiService.searchNews(  keywords,  start\_date,  end\_date,  category,  country,  language  )  }  --------------mdoel/repository/Repository--------------  class Repository(val localRepository: LocalRepository, private val remoteRepository: RemoteRepository): IRepository {  override var compositeDisposable: CompositeDisposable = CompositeDisposable()  override val searchedNews = MutableLiveData<List<News>>()  override val isProcessing = MutableLiveData<Boolean>()  override fun getLatestNews(): LiveData<List<News>> {  updateLatestNews()  return localRepository.getLatestNews()  }  override fun updateLatestNews() {  val disposable = remoteRepository.loadLatestNews()  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe({ res->  localRepository.saveNews(res.news)  },{  Log.i("newsapp", "Api call failed!")  })  compositeDisposable.add(disposable)  }  override fun searchNews(  keywords: String,  start\_date: String?,  end\_date: String?,  category: String?,  country: String?,  language: String?  ) = remoteRepository.searchNews(  keywords,  start\_date,  end\_date,  category,  country,  language  )  .observeOn(Schedulers.io())  .subscribeOn(AndroidSchedulers.mainThread())  .subscribe({  isProcessing.postValue(false)  searchedNews.postValue(it.news)  Log.i("newsapp", "Search Api call Success!")  },{  isProcessing.postValue(false)  it.printStackTrace()  Log.i("newsapp", "Search Api call failed!")  }).also { isProcessing.postValue(true) }  }  --------------view/MainActivity--------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: NewsViewModel  private var compositeDisposable = CompositeDisposable()  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViewModel()  setUpObserver()  initViews()  }  private fun initViews() {  binding.btnSearch.setOnClickListener {  viewModel.refreshNews()  }  // binding.btnSearch.setOnClickListener { \_ ->  // binding.edtSearch.text.toString().takeIf { x -> x.isNotBlank() }?.let {  // viewModel.searchNews(it)  // }  // }  -------------③RxJavaDebounceSearchView-------------  val disposable = RxSearchObservable.fromView(binding.edtSearch)  .debounce( 300, TimeUnit.MILLISECONDS)  .filter { it.isNotEmpty() }  .distinctUntilChanged()  .subscribe { viewModel.searchNews(it) }  compositeDisposable.add(disposable)  }  private fun setUpObserver() {  viewModel.isProcessing.observe(this) {  if(it) {  binding.progressBar.visibility = View.VISIBLE  } else {  binding.progressBar.visibility = View.GONE  }  }  viewModel.latestNews.observe(this) {  binding.rvNews.layoutManager = LinearLayoutManager(this)  binding.rvNews.adapter = NewsRvAdapter(this, it)  }  viewModel.searchedNews.observe(this) {  binding.rvNews.adapter = NewsRvAdapter(this, it)  }  }  private fun initViewModel() {  val remoteRepository = RemoteRepository(RetrofitBuilder.getRetrofit().create(ApiService::class.java))  val localRepository = LocalRepository(AppDatabase.getInstance(this.applicationContext))  val repository = Repository(localRepository, remoteRepository)  val factory = NewsViewModel(application, repository).createFactory()  viewModel = ViewModelProvider(this, factory)[NewsViewModel::class.java]  }  override fun onDestroy() {  super.onDestroy()  compositeDisposable.dispose()  }  --------------②RxJavaDebounceSearchView----------  object RxSearchObservable {  fun fromView(searchView: SearchView): Observable<String> {  val subject = PublishSubject.create<String>()  searchView.setOnQueryTextListener(object : SearchView.OnQueryTextListener {  override fun onQueryTextSubmit(s: String): Boolean {  subject.onComplete()  return true  }  override fun onQueryTextChange(text: String): Boolean {  subject.onNext(text)  return true  }  })  return subject  }  }  }  --------------view/NewsRvAdapter--------------  class NewsRvAdapter(private val context: Context, private val newsList: List<News>) :  RecyclerView.Adapter<NewsRvAdapter.NewsViewHolder>() {  private lateinit var binding: NewsItemBinding  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NewsViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NewsItemBinding.inflate(layoutInflater, parent, false)  return NewsViewHolder(binding.root)  }  override fun onBindViewHolder(holder: NewsViewHolder, position: Int) {  holder.apply {  val item = newsList[position]  item.apply {  newsTitle.text = title  //newsCategory.text = category?.get(0) ?: ""  newsAuthor.text = author  val url = newsList[position].image  Picasso.get().load(url).into(newsImg)  }  itemView.setOnClickListener {  /\* val intent = Intent(context, NewsDetailsActivity::class.java)  intent.putExtra("news",item)  context.startActivity(intent)\*/  }  }  }  override fun getItemCount(): Int {  return newsList.size  }  inner class NewsViewHolder(view: View) : RecyclerView.ViewHolder(view) {  val newsTitle = binding.txtTitle  val newsCategory = binding.txtCategory  val newsImg = binding.imgNews  val newsAuthor = binding.txtAuthor  }  }  --------------viewmodel/NewsViewModel--------------  class NewsViewModel(application: Application, val repository: IRepository): AndroidViewModel(application) {  val latestNews: LiveData<List<News>> = repository.getLatestNews()  val searchedNews: LiveData<List<News>> = repository.searchedNews  val isProcessing: LiveData<Boolean> = repository.isProcessing  fun refreshNews() {  repository.updateLatestNews()  }  fun searchNews(  keywords: String,  start\_date: String? = null,  end\_date: String? = null,  category: String? = null,  country: String? = null,  language: String? = null  ) {  repository.searchNews(  keywords,  start\_date,  end\_date,  category,  country,  language  ).also { repository.compositeDisposable.add(it) }  }  override fun onCleared() {  super.onCleared()  repository.compositeDisposable.dispose()  }  }  --------------viewmodel/NewsViewModelFactory--------------  fun <T: ViewModel> T.createFactory(): ViewModelProvider.Factory {  val viewModel = this  return object: ViewModelProvider.Factory {  @Suppress("UNCHECKED\_CAST")  override fun <T : ViewModel> create(modelClass: Class<T>): T = viewModel as T  }  }  --------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="#EFE8E8"  android:padding="5dp"  tools:context=".view.MainActivity">  -----------①RxJavaSearchView Debounce---------------  <androidx.appcompat.widget.SearchView  android:id="@+id/edtSearch"  android:layout\_width="0dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  android:layout\_marginEnd="100dp"  android:background="@drawable/search\_bg"  android:hint="@string/search\_here"  android:padding="10dp"  android:textColor="@color/black"  android:textColorHint="@color/primary\_color"  android:textSize="22sp"  app:layout\_constraintEnd\_toStartOf="@id/frame"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <FrameLayout  android:id="@+id/frame"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="5dp"  app:layout\_constraintBottom\_toBottomOf="@+id/edtSearch"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSearch"  android:layout\_width="wrap\_content"  android:layout\_height="50dp"  android:layout\_gravity="center"  android:text="Latest News" />  </FrameLayout>  <ProgressBar  android:id="@+id/progressBar"  style="?android:attr/progressBarStyleLarge"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:max="3"  android:progress="100"  android:visibility="gone"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintLeft\_toLeftOf="parent"  app:layout\_constraintRight\_toRightOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  tools:visibility="visible" />  <HorizontalScrollView  android:id="@+id/svLayout"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="8dp"  android:layout\_marginTop="8dp"  android:scrollbars="horizontal"  android:visibility="gone"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/edtSearch">  <LinearLayout  android:id="@+id/lnrLayout"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="8dp"  android:orientation="horizontal"  app:layout\_constraintTop\_toBottomOf="@+id/edtSearch">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnGeneral"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:background="@drawable/bg"  android:paddingEnd="10dp"  android:text="General"  android:clickable="true"  android:focusable="true"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnPolitics"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Politics"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnEntertainment"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Entertainment"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnFood"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Food"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnAcademic"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Academic"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSports"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Sports"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBusiness"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Business"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnRegional"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Regional"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  </LinearLayout>  </HorizontalScrollView>  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:elevation="10dp"  android:padding="10dp"  app:cardBackgroundColor="#FFFFFF"  app:cardCornerRadius="30dp"  app:layout\_constraintTop\_toBottomOf="@id/svLayout">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvNews"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp"  app:layout\_constraintTop\_toBottomOf="@+id/svLayout" />  </androidx.cardview.widget.CardView>  </androidx.constraintlayout.widget.ConstraintLayout>  --------------news\_item.xml--------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView 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="100dp"  android:layout\_margin="8dp"  android:elevation="15dp"  app:cardBackgroundColor="#E7E9EC"  app:cardCornerRadius="20dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgNews"  android:layout\_width="100dp"  android:layout\_height="100dp"  android:layout\_marginEnd="10dp"  android:scaleType="centerCrop"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtCategory"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="15dp"  android:background="@color/black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:textColor="@color/white"  android:textSize="16sp"  android:textStyle="italic"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toTopOf="parent"  tools:text="politics" />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="5dp"  android:maxLines="2"  android:textColor="@color/black"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtCategory"  tools:text="amazon" />  <TextView  android:id="@+id/txtAuthor"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="10dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  tools:text="Charles Ryan" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView> | </end> |
| <hitle> | RxJavaDemo5 flatMap | <chare> | 1 | <pext> | 03-10/ RxJavaDemo5 flatMap  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.button.setOnClickListener {  Observable.just("Titanic", "Conjuring", "Cars")  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .map { it -> "$it is old movie" }  .subscribe(  { item -> Log.i("tag","$item") }, // This is onNext Block  { error -> Log.i("tag","Error is $error") }, // This is onError Block  { Log.i("tag","Task got completed") } // This is onComplete Block  )  Observable.just(1, 2, 3, 4)  .flatMap { item -> Observable.just(item + 2) }  .subscribeOn(Schedulers.io())  .subscribe { values -> Log.i("tag","$values") }  Observable.merge(  Observable.interval(250, TimeUnit.MILLISECONDS).map { i -> "JAVA" },  Observable.interval(250, TimeUnit.MILLISECONDS).map { i -> "Kotlin" })  .take(10)  .subscribe { result -> Log.i("tag","$result") }  Observable.just("Swimming,", "Playing", "Boating", "Fishing")  .take(2)  .subscribe { result -> Log.i("tag", "Your main 2 hobbies are $result") }  Log.i("tag","Which team wins the repeated world cup in Football")  Observable.just("France", "Argentina", "England", "USA")  .repeat(2)  .subscribe { result -> Log.i("tag", "$result") }  Observable.range(1, 50)  .buffer(8)  .subscribe { println("Result: $it") }  Observable.range(1, 50)  .buffer(5) // range from initial to max [1,2,3,4,5] = creating a set of total buffer size  .take(4) // out of total result, get up to the count into the take input  .subscribe { println("Result: $it") }  }  }  }  fun main() {  // val groceryShopping = listOf(  // ShoppingBag(listOf("Mango", "Apple", "Banana")),  // ShoppingBag(listOf("LadyFinger", "Broccoli", "Onion")),  // ShoppingBag(listOf("Spices", "Knife", "Plates"))  // )  //  // val clothesShopping = listOf(  // ShoppingBag(listOf("Pant", "Shirt", "Tie")),  // ShoppingBag(listOf("Socks", "Shoes", "Jeans")),  // ShoppingBag(listOf("Jackets", "Caps", "Muffler"))  // )  //  // val grocery = groceryShopping.flatMap {  // it.items  // }  // println(grocery)  //  // val clothes = clothesShopping.map {  // it.items  // }  // println(clothes)  Maybe.just("Example of Maybe Observable")  .subscribe(  { result -> println("result is $result") },  { error -> println("error is $error") },  { println("Completed the task") }  )  Completable.create { completableEmitter ->  completableEmitter.onComplete()  completableEmitter.onError(Exception("something wrong!!"))  }  Flowable.just("Example of Flowable Observable")  .subscribe(  { result -> println("result is $result") },  { error -> println("error is $error") },  { println("Completed the task") }  )  }  class ShoppingBag(val items: List<String>) | </end> |
| <hitle> | DIDemo1 Dagger2DogApiMVVM | <chare> | 1 | <pext> | 03-13/DIDemo1 Dagger2 MVVM DogApi  ----------------Gradle.module----------------  plugins {  id 'kotlin-kapt'  }  }  dependencies {  //dagger  implementation 'com.google.dagger:dagger:2.42'  kapt 'com.google.dagger:dagger-compiler:2.42'  //Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  //Convertor factory by Gson  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //for logging interceptor  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  //for RxJava  implementation 'io.reactivex.rxjava3:rxjava:3.1.6'  implementation 'io.reactivex.rxjava3:rxandroid:3.0.2'  implementation 'com.squareup.retrofit2:adapter-rxjava3:2.9.0'  //Picasso  implementation 'com.squareup.picasso:picasso:2.71828'  //for Room Database  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  }  ----------------Gradle.project----------------  plugins {  id 'com.android.application' version '7.3.1' apply false  id 'com.android.library' version '7.3.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.20' apply false  }  ----------------MyApplication----------------  class MyApplication:Application() {  lateinit var appComponent: AppComponent  override fun onCreate() {  super.onCreate()  appComponent = DaggerAppComponent.builder()  .appModule(AppModule(this))  .apiModule(ApiModule())  .build()  }  fun getNetComponent(): AppComponent {  return appComponent  }  }  ----------------di/ApiM odule----------------  @Module  class ApiModule {  // @Provides  // @Singleton  // fun provideRepository(): Repository {  // return Repository(provideAPIService())  // }  //  // @Provides  // @Singleton  // fun provideAPIService(): ApiService {  // return provideRetrofit().create(ApiService::class.java)  // }  @Provides  @Singleton  fun provideClient():OkHttpClient {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  return OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  }  @Provides  @Singleton  fun provideRetrofit(): Retrofit {  return Retrofit.Builder()  .addConverterFactory(GsonConverterFactory.create())  .addCallAdapterFactory(RxJava3CallAdapterFactory.create())  .client(provideClient())  .baseUrl(BASE\_URL)  .build()  }  }  ----------------di/AppComponent----------------  @Singleton  @Component(modules = [AppModule::class, ApiModule::class])  interface AppComponent {  fun inject(mainActivity: MainActivity)  }  ----------------di/AppModule----------------  @Module  class AppModule(private val application: Application) {  @Provides  @Singleton  fun provideApplication(): Application {  return application  }  }  ------------------di/ApiModule-------------------  @Module  class ApiModule {  // @Provides  // @Singleton  // fun provideRepository(): Repository {  // return Repository(provideAPIService())  // }  //  // @Provides  // @Singleton  // fun provideAPIService(): ApiService {  // return provideRetrofit().create(ApiService::class.java)  // }  @Provides  @Singleton  fun provideClient():OkHttpClient {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  return OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  }  @Provides  @Singleton  fun provideRetrofit(): Retrofit {  return Retrofit.Builder()  .addConverterFactory(GsonConverterFactory.create())  .addCallAdapterFactory(RxJava3CallAdapterFactory.create())  .client(provideClient())  .baseUrl(BASE\_URL)  .build()  }  }  ----------------model/remote/ApiService----------------  interface ApiService {  @GET(END\_POINT)  fun getRandomDog(): Single<DogResponse>  }  ----------------model/remote/DogResponse----------------  data class DogResponse(  val message: String,  val status: String  )  ----------------model/Constants----------------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  ----------------model/Repository----------------  class Repository(private val apiService: ApiService) {  fun getRandomDog() = apiService.getRandomDog()  }  ----------------view/MainActivity----------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: DogViewModel  private lateinit var factory: DogViewModelFactory  @Inject  lateinit var retrofit: Retrofit  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  (application as MyApplication).getNetComponent().inject(this)  initViews()  setupViewModels()  setupObserver()  }  private fun setupObserver() {  with(viewModel) {  dogResponse.observe(this@MainActivity) {  Picasso.get()  .load(it.message)  .placeholder(R.drawable.ic\_launcher\_background)  .into(binding.imageView)  }  }  }  private fun initViews() {  binding.button.setOnClickListener {  viewModel.getRandomDog()  }  }  private fun setupViewModels() {  if(this::retrofit.isInitialized) {  factory = DogViewModelFactory(repository = Repository(retrofit.create(ApiService::class.java)))  viewModel = ViewModelProvider(this, factory)[DogViewModel::class.java]  }  }  }  ----------------viewmodel/DogViewModel----------------  class DogViewModel(private val repository: Repository): ViewModel() {  val dogResponse = MutableLiveData<DogResponse>()  private lateinit var disposable: Disposable  fun getRandomDog() {  disposable = repository.getRandomDog()  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .subscribe({  dogResponse.postValue(it)  },{  print(it.message.toString())  })  }  override fun onCleared() {  super.onCleared()  if(this::disposable.isInitialized) {  disposable.dispose()  }  }  }  ----------------viewmodel/DogViewModelFactory----------------  class DogViewModelFactory(private val repository: Repository): ViewModelProvider.Factory {  override fun <T : ViewModel> create(modelClass: Class<T>, extras: CreationExtras): T {  return if (modelClass.isAssignableFrom(DogViewModel::class.java)) {  DogViewModel(this.repository) as T  } else {  throw IllegalAccessException("View model not found")  }  }  } | </end> |
| <hitle> | DaggerHiltDogApp | <chare> | 1 | <pext> | 03-14/DaggerHiltDogApp  --------------project setup---------------  If you need Dagger Hilt + Rxjava3 + Retrofit + Glide then go for below set of dependencies  Step 1 : add dependency into your build.gradle (Module: app)  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'dagger.hilt.android.plugin'  id 'kotlin-kapt'  }  android {  namespace 'com.example.daggerhitdogapp'  compileSdk 33  defaultConfig {  applicationId "com.example.daggerhitdogapp"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures{  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  //Hilt dependency  implementation 'com.google.dagger:hilt-android:2.42'  kapt 'com.google.dagger:hilt-android-compiler:2.42'  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  //Rx-Java  implementation 'io.reactivex.rxjava3:rxjava:3.1.6'  implementation 'io.reactivex.rxjava3:rxandroid:3.0.2'  implementation "com.github.akarnokd:rxjava3-retrofit-adapter:3.0.0"  // Glide  implementation 'com.github.bumptech.glide:glide:4.14.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.14.2'  // ktx activity with view model injection  implementation 'androidx.activity:activity-ktx:1.6.1'  }  Step 2: add dependency into you build.gradle(:app)  // Top-level build file where you can add configuration options common to all sub-projects/modules.  buildscript {  repositories {  google()  mavenCentral()  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.42'  }  }  plugins {  id 'com.android.application' version '7.2.1' apply false  id 'com.android.library' version '7.2.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.20' apply false  }  task clean(type: Delete) {  delete rootProject.buildDir  }  --------------App-------------------------  @HiltAndroidApp  class App: Application() {  }  -----------------di/NetworkModule----------------------  @Module  @InstallIn(SingletonComponent::class)  object NetworkModule {  @Singleton  @Provides  fun provideBaseUrl(): String {  return BASE\_URL  }  @Singleton  @Provides  fun provideConverterFactory(): Converter.Factory {  return GsonConverterFactory.create()  }  @Singleton  @Provides  fun provideRxJavaAdapterFactory(): CallAdapter.Factory {  return RxJava3CallAdapterFactory.create()  }  @Singleton  @Provides  fun provideLoggingInterceptor(): HttpLoggingInterceptor {  return HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)  }  @Singleton  @Provides  fun provideClients(httpLoggingInterceptor: HttpLoggingInterceptor): OkHttpClient {  val okHttpClient = OkHttpClient.Builder()  .addInterceptor(httpLoggingInterceptor)  return okHttpClient.build()  }  @Singleton  @Provides  fun provideRetrofit(  baseUrl: String,  converterFactory: Converter.Factory,  callAdapter: CallAdapter.Factory,  okHttpClient: OkHttpClient  ): Retrofit {  return Retrofit.Builder()  .baseUrl(baseUrl)  .addConverterFactory(converterFactory)  .addCallAdapterFactory(callAdapter)  .client(okHttpClient)  .build()  }  @Singleton  @Provides  fun provideAPIService(retrofit: Retrofit): ApiService {  return retrofit.create(ApiService::class.java)  }  @Singleton  @Provides  fun provideRepository(apiService: ApiService): Repository {  return Repository(apiService)  }  }  ---------------model/remote/dto/ApiResponse------------------------  data class ApiResponse(  val message: String,  val status: String  )  ------------model/remote/ApiService---------------------------  interface ApiService {  @GET(END\_POINT)  fun getResultFromApi(): Single<ApiResponse>  }  ----------------model/remote/Constants-----------------------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  class Repository @Inject constructor(private val apiService: ApiService) {  fun getResultFromNetwork() = apiService.getResultFromApi()  }  --------------------------model/repositories/Repository--------------  class Repository @Inject constructor(private val apiService: ApiService) {  fun getResultFromNetwork() = apiService.getResultFromApi()  }  -------------viewmodel/DogViewModel--------------------------  @HiltViewModel  class DogViewModel @Inject constructor(private val repository: Repository): ViewModel() {  val dogLiveDataFromNetworkSource = MutableLiveData<ApiResponse>()  val errorLiveData = MutableLiveData<String>()  private lateinit var disposable: Disposable  fun getDogFromRepository() {  disposable = repository.getResultFromNetwork()  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .subscribe({  dogLiveDataFromNetworkSource.postValue(it)  },{  errorLiveData.postValue(it.message.toString())  })  }  override fun onCleared() {  super.onCleared()  if(this::disposable.isInitialized) {  disposable.dispose()  }  }  }  -------------view/MainActivity--------------------------  @AndroidEntryPoint  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private val viewModel: DogViewModel by viewModels()  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  setupObserver()  }  private fun initViews() {  binding.button.setOnClickListener {  viewModel.getDogFromRepository()  }  }  private fun setupObserver() {  viewModel.dogLiveDataFromNetworkSource.observe(this) {  Glide.with(this)  .load(it.message)  .placeholder(R.drawable.ic\_launcher\_background)  .error(android.R.drawable.ic\_dialog\_alert)  .into(binding.imageView)  }  viewModel.errorLiveData.observe(this) {  Toast.makeText(this, it, Toast.LENGTH\_SHORT).show()  }  }  }  -------------androidManifest--------------------------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <application  android:name=".App"  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.DaggerHiltDogApp"  tools:targetApi="31">  <activity  android:name=".view.MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  -----------activity\_main.xml----------------------------  imageView, button | </end> |
| <hitle> | DaggerHiltNewsApp | <chare> | 1 | <pext> | 03-14/DaggerHiltNewsApp  -------------Gradle.module----------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'dagger.hilt.android.plugin'  id 'kotlin-kapt'  id 'kotlin-parcelize'  }  android {  namespace 'com.example.newsapp'  compileSdk 33  defaultConfig {  applicationId "com.example.newsapp"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures {  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  //Hilt dependency  implementation 'com.google.dagger:hilt-android:2.42'  kapt 'com.google.dagger:hilt-android-compiler:2.42'  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  //Rx-Java  implementation 'io.reactivex.rxjava3:rxjava:3.1.6'  implementation 'io.reactivex.rxjava3:rxandroid:3.0.2'  implementation "com.github.akarnokd:rxjava3-retrofit-adapter:3.0.0"  // Glide  implementation 'com.github.bumptech.glide:glide:4.14.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.14.2'  implementation 'com.squareup.picasso:picasso:2.71828'  // ktx activity with view model injection  implementation 'androidx.activity:activity-ktx:1.6.1'  }  -------------Gradle.Project-------------  // Top-level build file where you can add configuration options common to all sub-projects/modules.  buildscript {  repositories {  google()  mavenCentral()  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.42'  }  }  plugins {  id 'com.android.application' version '7.2.1' apply false  id 'com.android.library' version '7.2.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.20' apply false  }  task clean(type: Delete) {  delete rootProject.buildDir  }  -------------news\_item.xml-------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView 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="100dp"  android:layout\_margin="8dp"  android:elevation="15dp"  app:cardBackgroundColor="#E7E9EC"  app:cardCornerRadius="20dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imgNews"  android:layout\_width="100dp"  android:layout\_height="100dp"  android:layout\_marginEnd="10dp"  android:scaleType="centerCrop"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/txtCategory"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="15dp"  android:background="@color/black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:textColor="@color/white"  android:textSize="16sp"  android:textStyle="italic"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toTopOf="parent"  tools:text="politics" />  <TextView  android:id="@+id/txtTitle"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="5dp"  android:maxLines="2"  android:textColor="@color/black"  android:textSize="20sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtCategory"  tools:text="amazon" />  <TextView  android:id="@+id/txtAuthor"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginStart="10dp"  android:layout\_marginTop="10dp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toEndOf="@+id/imgNews"  app:layout\_constraintTop\_toBottomOf="@+id/txtTitle"  tools:text="Charles Ryan" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  -------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="#EFE8E8"  android:padding="5dp"  tools:context=".view.MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/edtSearch"  android:layout\_width="0dp"  android:layout\_height="50dp"  android:layout\_margin="10dp"  android:layout\_marginEnd="100dp"  android:background="@drawable/search\_bg"  android:hint="@string/search\_here"  android:padding="10dp"  android:textColor="@color/black"  android:textColorHint="@color/primary\_color"  android:textSize="22sp"  app:layout\_constraintEnd\_toStartOf="@id/frame"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <FrameLayout  android:id="@+id/frame"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="5dp"  app:layout\_constraintBottom\_toBottomOf="@+id/edtSearch"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSearch"  android:layout\_width="wrap\_content"  android:layout\_height="50dp"  android:layout\_gravity="center"  android:background="@drawable/search\_bg" />  <androidx.appcompat.widget.AppCompatImageView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:elevation="5dp"  android:src="@drawable/ic\_baseline\_search\_24" />  </FrameLayout>  <ProgressBar  android:id="@+id/progressBar"  style="?android:attr/progressBarStyleLarge"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:max="3"  android:progress="100"  android:visibility="gone"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintLeft\_toLeftOf="parent"  app:layout\_constraintRight\_toRightOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  tools:visibility="visible" />  <HorizontalScrollView  android:id="@+id/svLayout"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_margin="8dp"  android:layout\_marginTop="8dp"  android:scrollbars="horizontal"  android:visibility="gone"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/edtSearch">  <LinearLayout  android:id="@+id/lnrLayout"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="8dp"  android:orientation="horizontal"  app:layout\_constraintTop\_toBottomOf="@+id/edtSearch">  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnGeneral"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:background="@drawable/bg"  android:paddingEnd="10dp"  android:text="General"  android:clickable="true"  android:focusable="true"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnPolitics"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Politics"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnEntertainment"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Entertainment"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnFood"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Food"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnAcademic"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Academic"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnSports"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Sports"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnBusiness"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Business"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnRegional"  android:layout\_width="wrap\_content"  android:layout\_height="35dp"  android:layout\_marginEnd="10dp"  android:background="@drawable/bg"  android:elevation="10dp"  android:fontFamily="sans-serif-black"  android:paddingStart="10dp"  android:paddingEnd="10dp"  android:text="Regional"  android:textAllCaps="false"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="bold" />  </LinearLayout>  </HorizontalScrollView>  <androidx.cardview.widget.CardView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="10dp"  android:elevation="10dp"  android:padding="10dp"  app:cardBackgroundColor="#FFFFFF"  app:cardCornerRadius="30dp"  app:layout\_constraintTop\_toBottomOf="@id/svLayout">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvNews"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:elevation="10dp"  app:layout\_constraintTop\_toBottomOf="@+id/svLayout" />  </androidx.cardview.widget.CardView>  </androidx.constraintlayout.widget.ConstraintLayout>  -------------di/NetworkModule-------------  @Module  @InstallIn(SingletonComponent::class)  object NetworkModule {  @Singleton  @Provides  fun provideBaseUrl(): String {  return BASE\_URL  }  @Singleton  @Provides  fun provideConverterFactory(): Converter.Factory {  return GsonConverterFactory.create()  }  @Singleton  @Provides  fun provideRxJavaAdapterFactory(): CallAdapter.Factory {  return RxJava3CallAdapterFactory.create()  }  @Singleton  @Provides  fun provideLoggingInterceptor(): HttpLoggingInterceptor {  return HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)  }  @Singleton  @Provides  fun provideClients(httpLoggingInterceptor: HttpLoggingInterceptor): OkHttpClient {  val okHttpClient = OkHttpClient.Builder()  .addInterceptor(AuthInterceptor())  .addInterceptor(httpLoggingInterceptor)  return okHttpClient.build()  }  @Singleton  @Provides  fun provideRetrofit(  baseUrl: String,  converterFactory: Converter.Factory,  callAdapter: CallAdapter.Factory,  okHttpClient: OkHttpClient  ): Retrofit {  return Retrofit.Builder()  .baseUrl(baseUrl)  .addConverterFactory(converterFactory)  .addCallAdapterFactory(callAdapter)  .client(okHttpClient)  .build()  }  @Singleton  @Provides  fun provideAPIService(retrofit: Retrofit): ApiService {  return retrofit.create(ApiService::class.java)  }  @Singleton  @Provides  fun provideRepository(apiService: ApiService): Repository {  return Repository(apiService)  }  }  -------------model/remote/data/-------------  data class News(  val author: String,  val description: String,  val id: String,  val image: String,  val language: String,  val published: String,  val title: String,  val url: String  )  data class NewsResponse(  val news: List<News>,  val status: String  )  -------------model/remote/ApiService-------------  interface ApiService {  @GET(END\_POINT)  fun getLatestNewsFromApi(): Single<NewsResponse>  }  -------------model/remote/AuthInterceptor-------------  class AuthInterceptor: Interceptor {  override fun intercept(chain: Interceptor.Chain): Response {  val currentRequest = chain.request().newBuilder()  currentRequest.addHeader(AUTHORIZATION, TOKEN)  val newRequest = currentRequest.build()  return chain.proceed(newRequest)  }  }  -------------model/remote/Constant-------------  object Constant {  const val BASE\_URL = "https://api.currentsapi.services/v1/"  const val END\_POINT = "latest-news"  const val AUTHORIZATION = "Authorization"  const val TOKEN = "ehTUs\_L7VNOevxSsW301L3Y6KhOmJ573Grs-VKu--uPjKPZF"  }  -------------model/remote/RetrofitBuilder-------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::retrofit.isInitialized) {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder()  .addInterceptor(AuthInterceptor())  .addInterceptor(loggingInterceptor)  .build()  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .client(client)  .build()  }  return retrofit  }  }  -------------model/remote/Repository-------------  class Repository @Inject constructor(private val apiService: ApiService) {  fun getLatestNewsFromNetwork() = apiService.getLatestNewsFromApi()  }  -------------view/MainActivity-------------  @AndroidEntryPoint  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private val viewModel: NewsViewModel by viewModels()  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpObserver()  }  private fun setUpObserver() {  viewModel.latestNews.observe(this) {  binding.rvNews.layoutManager = LinearLayoutManager(this)  binding.rvNews.adapter = NewsRvAdapter(this, it)  }  }  }  -------------view/NewsRvAdapter-------------  class NewsRvAdapter(private val context: Context, private val newsList: List<News>) :  RecyclerView.Adapter<NewsRvAdapter.NewsViewHolder>() {  private lateinit var binding: NewsItemBinding  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NewsViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NewsItemBinding.inflate(layoutInflater, parent, false)  return NewsViewHolder(binding.root)  }  override fun onBindViewHolder(holder: NewsViewHolder, position: Int) {  holder.apply {  val item = newsList[position]  item.apply {  newsTitle.text = title  //newsCategory.text = category?.get(0) ?: ""  newsAuthor.text = author  val url = newsList[position].image  Picasso.get().load(url).into(newsImg)  }  itemView.setOnClickListener {  /\* val intent = Intent(context, NewsDetailsActivity::class.java)  intent.putExtra("news",item)  context.startActivity(intent)\*/  }  }  }  override fun getItemCount(): Int {  return newsList.size  }  inner class NewsViewHolder(view: View) : RecyclerView.ViewHolder(view) {  val newsTitle = binding.txtTitle  val newsCategory = binding.txtCategory  val newsImg = binding.imgNews  val newsAuthor = binding.txtAuthor  }  }  -------------viewmodel/NewsViewModel-------------  @HiltViewModel  class NewsViewModel @Inject constructor(private val repository: Repository): ViewModel() {  val latestNews = MutableLiveData<List<News>>()  val errorLatesNews = MutableLiveData<String>()  private lateinit var disposable: Disposable  fun refreshNews() {  disposable = repository.getLatestNewsFromNetwork()  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .subscribe({  latestNews.postValue(it.news)  },{  errorLatesNews.postValue(it.message.toString())  })  }  override fun onCleared() {  super.onCleared()  if(this::disposable.isInitialized) {  disposable.dispose()  }  }  }  -------------App-------------  @HiltAndroidApp  class App: Application() {  }  -------------AndroidManifest-------------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <application  -------------important add App into Manifest-------------  android:name=".App"  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.NewsApp"  tools:targetApi="31">  <activity  android:name=".view.MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest> | </end> |
| <hitle> | JetPackComposerDemo1 Starting | <chare> | 1 | <pext> | 03-15/JetPackComposerDemo1 Starting  Create the project with “Empty Compose Activity (Material3)”  @ExperimentalMaterial3Api  class MainActivity : ComponentActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  JetPackComposeDemo1Theme {  // A surface container using the 'background' color from the theme  Surface(  modifier = Modifier.fillMaxSize(),  color = MaterialTheme.colorScheme.background  ) {  Greeting("Android")  }  }  }  }  }  @Composable  @ExperimentalMaterial3Api  fun Greeting(name: String) {  Card(modifier = Modifier  .fillMaxWidth()  .wrapContentHeight()  .background(color = Color.Cyan)  ) {  Row(modifier = Modifier.clip(RectangleShape).padding(10.dp)) {  Image(  painter = painterResource(id = R.drawable.ic\_launcher\_background),  contentDescription = "background",  modifier = Modifier.size(110.dp)  )  Column {  Text(  text = "Title",  fontSize = 25.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  Spacer(modifier = Modifier.padding(10.dp))  Text(  text = "Desription",  fontSize = 20.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  }  }  }  // Text(text = "Hello $name!")  }  @Preview(showBackground = true)  @Composable  @ExperimentalMaterial3Api  fun DefaultPreview() {  JetPackComposeDemo1Theme {  Greeting("Android")  }  } | </end> |
| <hitle> | JetpackComposeDemo1 Horizontal/Vertical RecyclerView Expand animation | <chare> | 1 | <pext> | 03-15/JetPackComposerDemo1 RecyclerView  @ExperimentalMaterial3Api  class MainActivity : ComponentActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  JetPackComposeDemo1Theme {  // A surface container using the 'background' color from the theme  Surface(  modifier = Modifier.fillMaxSize(),  color = MaterialTheme.colorScheme.background  ) {  // Greeting("Android")  DefaultPreview()  }  }  }  }  }  @Composable  @ExperimentalMaterial3Api  fun Greeting(name: String) {  Card(modifier = Modifier  .fillMaxWidth()  .wrapContentHeight()  .background(color = Color.Cyan)  ) {  Row(modifier = Modifier  .clip(RectangleShape)  .padding(10.dp)) {  Image(  painter = painterResource(id = R.drawable.ic\_launcher\_background),  contentDescription = "background",  modifier = Modifier.size(110.dp)  )  Column {  Text(  text = "Title",  fontSize = 25.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  Spacer(modifier = Modifier.padding(10.dp))  Text(  text = "Desription",  fontSize = 20.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  }  }  }  // Text(text = "Hello $name!")  }  @Preview(showBackground = true)  @Composable  @ExperimentalMaterial3Api  fun DefaultPreview() {  JetPackComposeDemo1Theme {  PrepareSportsList()  }  }  @ExperimentalMaterial3Api  @Composable  fun PrepareSportsList() {  val listOfSports = listOf(  Sports(R.drawable.ic\_launcher\_background, "Football", "This is 9 player game"),  Sports(R.drawable.ic\_launcher\_background, "Basketball", "This is 9 player game"),  Sports(R.drawable.ic\_launcher\_background, "Cricket", "This is 9 player game"),  Sports(R.drawable.ic\_launcher\_background, "Pool", "This is 9 player game"),  Sports(R.drawable.baseline\_sports\_volleyball\_24, "VolleyBall", "This is 9 player game"),  Sports(R.drawable.ic\_launcher\_background, "Swimming", "This is 9 player game"),  )  Column {  MakeAListOfGames1(listOfSports)  Spacer(modifier = Modifier.padding(20.dp))  MakeAListOfGames2(listOfSports)  }  }  @ExperimentalMaterial3Api  @Composable  fun MakeAListOfGames1(sports: List<Sports>) {  LazyRow {  items(sports) {sport ->  SportItemForHorizontal(sport = sport)  }  }  }  @ExperimentalMaterial3Api  @Composable  fun SportItemForHorizontal(sport: Sports) {  Card(modifier = Modifier  .fillMaxWidth()  .wrapContentHeight()  .wrapContentWidth()  .background(color = Color.Cyan)  ) {  Row(modifier = Modifier  .clip(RectangleShape)  .padding(5.dp)) {  Image(  painter = painterResource(id = sport.image),  contentDescription = "background",  modifier = Modifier.size(60.dp)  )  Text(  text = sport.name,  fontSize = 17.sp,  modifier = Modifier.padding(horizontal = 10.dp)  )  Spacer(modifier = Modifier.padding(10.dp))  Text(  text = sport.about,  fontSize = 13.sp,  modifier = Modifier.padding(horizontal = 10.dp)  )  }  }  }  @ExperimentalMaterial3Api  @Composable  fun MakeAListOfGames2(sports: List<Sports>) {  LazyColumn {  items(sports) {sport ->  SportItem(sport = sport)  }  }  }  @ExperimentalMaterial3Api  @Composable  fun SportItem(sport: Sports) {  var expanded by remember { mutableStateOf(false) }  Column(modifier = Modifier.animateContentSize(  animationSpec = spring(  dampingRatio = Spring.DampingRatioHighBouncy,  stiffness = Spring.StiffnessLow  )  )) {  Row(modifier = Modifier  .background(Color.Cyan)  .fillMaxWidth()  ) {  Image(  painter = painterResource(id = sport.image),  contentDescription = "background",  modifier = Modifier  .size(60.dp)  .clip(CircleShape)  .border(5.dp, Color.Green, CircleShape)  )  Spacer(modifier = Modifier.padding(15.dp))  Column {  Text(  text = sport.name,  fontSize = 25.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  Spacer(modifier = Modifier.padding(10.dp))  Text(  text = sport.about,  fontSize = 20.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  }  Spacer(modifier = Modifier.weight(1f))  ExpandCollapseItemButon(  expanded = expanded,  onClick = { expanded = !expanded }  )  }  if (expanded) {  AboutSportsInDetail(sport.about)  }  }  }  @Composable  fun AboutSportsInDetail(about: String, modifier: Modifier = Modifier) {  Column(modifier = modifier.padding(10.dp)) {  Text(  text = about,  fontSize = 25.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  Spacer(modifier = Modifier.padding(10.dp))  Text(  text = "game rating 10.0",  fontSize = 20.sp,  modifier = Modifier.padding(horizontal = 16.dp)  )  }  }  @Composable  fun ExpandCollapseItemButon(expanded: Boolean, onClick: () -> Unit) {  IconButton(onClick = onClick) {  Icon(  imageVector =  if (expanded) ImageVector.vectorResource(id = R.drawable.baseline\_expand\_less\_24)  else ImageVector.vectorResource(id = R.drawable.baseline\_expand\_more\_24),  tint = Color.Magenta,  contentDescription = "Spring Effect"  )  }  }  data class Sports(@DrawableRes val image: Int, val name: String, val about: String) | </end> |
| <hitle> | ComposeMVVM Dog DaggerHilt | <chare> | 1 | <pext> | 03-16/ComposeMVVM  ----------------------gradle.module----------------------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'dagger.hilt.android.plugin'  id 'kotlin-kapt'  }  android {  namespace 'com.example.composemvvm'  compileSdk 33  defaultConfig {  applicationId "com.example.composemvvm"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  vectorDrawables {  useSupportLibrary true  }  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures {  compose true  }  composeOptions {  kotlinCompilerExtensionVersion '1.2.0'  }  packagingOptions {  resources {  excludes += '/META-INF/{AL2.0,LGPL2.1}'  }  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'  implementation 'androidx.activity:activity-compose:1.3.1'  implementation "androidx.compose.ui:ui:$compose\_version"  implementation "androidx.compose.ui:ui-tooling-preview:$compose\_version"  implementation 'androidx.compose.material3:material3:1.0.0-alpha11'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  androidTestImplementation "androidx.compose.ui:ui-test-junit4:$compose\_version"  debugImplementation "androidx.compose.ui:ui-tooling:$compose\_version"  debugImplementation "androidx.compose.ui:ui-test-manifest:$compose\_version"  implementation "androidx.compose.material:material-icons-extended:$compose\_version"  //landscape - glide for compose  implementation "com.github.skydoves:landscapist-glide:1.5.0"  //Dagger - Hilt  implementation "com.google.dagger:hilt-android:2.44.2"  kapt "com.google.dagger:hilt-android-compiler:2.44.2"  kapt "androidx.hilt:hilt-compiler:1.0.0"  implementation 'androidx.hilt:hilt-navigation-compose:1.1.0-alpha01'  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  // RxJava 3  implementation 'io.reactivex.rxjava3:rxjava:3.1.6'  implementation 'io.reactivex.rxjava3:rxandroid:3.0.2'  implementation "com.github.akarnokd:rxjava3-retrofit-adapter:3.0.0"  //compose with liveDate  implementation 'androidx.compose.runtime:runtime-livedata:1.3.3'  }  ----------------------Gradle.project----------------------  buildscript {  ext {  compose\_version = '1.2.0'  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.44.2'  classpath "com.android.tools.build:gradle:7.0.3"  }  }// Top-level build file where you can add configuration options common to all sub-projects/modules.  plugins {  id 'com.android.application' version '7.4.1' apply false  id 'com.android.library' version '7.4.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.0' apply false  }  ----------------------di/NetworkModule----------------------  @Module  @InstallIn(SingletonComponent::class)  object NetworkModule {  @Singleton  @Provides  fun provideBaseUrl(): String {  return BASE\_URL  }  @Singleton  @Provides  fun provideConverterFactory(): Converter.Factory {  return GsonConverterFactory.create()  }  @Singleton  @Provides  fun provideRxJavaAdapterFactory(): CallAdapter.Factory {  return RxJava3CallAdapterFactory.create()  }  @Singleton  @Provides  fun provideLoggingInterceptor(): HttpLoggingInterceptor {  return HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)  }  @Singleton  @Provides  fun provideClients(httpLoggingInterceptor: HttpLoggingInterceptor): OkHttpClient {  val okHttpClient = OkHttpClient.Builder()  .addInterceptor(httpLoggingInterceptor)  return okHttpClient.build()  }  @Singleton  @Provides  fun provideRetrofit(  baseUrl: String,  converterFactory: Converter.Factory,  callAdapter: CallAdapter.Factory,  okHttpClient: OkHttpClient  ): Retrofit {  return Retrofit.Builder()  .baseUrl(baseUrl)  .addConverterFactory(converterFactory)  .addCallAdapterFactory(callAdapter)  .client(okHttpClient)  .build()  }  @Singleton  @Provides  fun provideAPIService(retrofit: Retrofit): ApiService {  return retrofit.create(ApiService::class.java)  }  @Singleton  @Provides  fun provideRepository(apiService: ApiService): Repository {  return RepositoryImplementation(apiService)  }  }  ----------------------model/remote/dto/ApiResponse----------------------  data class ApiResponse(  val message: String,  val status: String  )  ----------------------model/remote/ApiService----------------------  interface ApiService {  @GET(END\_POINT)  fun getResultFromApi(): Observable<ApiResponse>  }  ----------------------model/remote/Constants----------------------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  ----------------------model/repository/Repository----------------------  interface Repository {  fun makeApiCall(): Observable<ApiResponse>  }  ----------------------model/repository/RepositoryImplementation----------------------  class RepositoryImplementation @Inject constructor(private val apiService: ApiService): Repository {  override fun makeApiCall(): Observable<ApiResponse> {  return apiService.getResultFromApi()  }  }  ----------------------view/MainActivity----------------------  @AndroidEntryPoint  class MainActivity : ComponentActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  ComposeMVVMTheme {  // A surface container using the 'background' color from the theme  Surface(  modifier = Modifier.fillMaxSize(),  color = MaterialTheme.colorScheme.background  ) {  MainScreen()  }  }  }  }  }  @Composable  fun Greeting(name: String) {  Text(text = "Hello $name!")  }  @Preview(showBackground = true)  @Composable  fun DefaultPreview() {  ComposeMVVMTheme {  MainScreen()  }  }  ----------------------view/MainScreen----------------------  @Composable  fun MainScreen(viewModel: DogViewModel = hiltViewModel()) {  val response = viewModel.responseFromApi.observeAsState()  Column(Modifier.fillMaxWidth()) {  response.value?.let {  GlideImage(  modifier = Modifier  .fillMaxWidth()  .fillMaxHeight(0.8f),  imageModel = it.message  )  }  Button(  onClick = { viewModel.getDataFromApi() },  modifier = Modifier.fillMaxWidth()) {  Text("Get Dog")  }  }  }  ----------------------viewmodel/DogViewModel----------------------  @HiltViewModel  class DogViewModel @Inject constructor(private val repository: Repository): ViewModel() {  val responseFromApi = MutableLiveData<ApiResponse>()  private lateinit var disposable: Disposable  fun getDataFromApi() {  disposable = repository.makeApiCall()  .subscribeOn(Schedulers.io())  .observeOn(AndroidSchedulers.mainThread())  .subscribe({  responseFromApi.postValue(it)  },{  it.printStackTrace()  })  }  override fun onCleared() {  super.onCleared()  if(this::disposable.isInitialized) {  disposable.dispose()  }  }  }  ----------------------App----------------------  @HiltAndroidApp  class App: Application() {  }  ----------------------AndroidManifest----------------------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <application  android:name=".App"  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.ComposeMVVM"  tools:targetApi="31">  <activity  android:name=".view.MainActivity"  android:exported="true"  android:label="@string/app\_name"  android:theme="@style/Theme.ComposeMVVM">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest> | </end> |
| <hitle> | JetpackComposerLoginDemo Register | <chare> | 1 | <pext> | 03-16/ JetpackComposerLoginDemo Register  --------Gradle------------  implementation "androidx.compose.material:material-icons-extended:$compose\_version"  -------MainActivity-------------  class MainActivity : ComponentActivity() {  private lateinit var context: Context  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  JetpackComposerLoginDemoTheme {  // A surface container using the 'background' color from the theme  Surface(  modifier = Modifier.fillMaxSize(),  color = MaterialTheme.colorScheme.background  ) {  context = LocalContext.current  // LoginUI()  RegisterUI()  }  }  }  }  fun login(email: String, password: String) {  if (email.isNotEmpty() && password.isNotEmpty()) {  showToast("Login success")  } else {  showToast("Login Failed!")  }  }  private fun showToast(message: String) {  Toast.makeText(context, message, Toast.LENGTH\_SHORT).show()  }  @Preview(showBackground = true)  @Composable  fun DefaultPreview() {  JetpackComposerLoginDemoTheme {  RegisterUI()  }  }  @Composable  fun RegisterUI() {  var email = remember { mutableStateOf("") }  var password = remember { mutableStateOf("") }  var phone = remember { mutableStateOf("") }  var age by remember { mutableStateOf("") }  var showPassword = remember { mutableStateOf(false) }  var isUnder18 by remember { mutableStateOf(false) }  Box(  modifier = Modifier  .fillMaxSize()  .background(MaterialTheme.colorScheme.secondary),  contentAlignment = Alignment.Center  ) {  Column(  modifier = Modifier.padding(10.dp),  verticalArrangement = Arrangement.spacedBy(10.dp),  horizontalAlignment = Alignment.CenterHorizontally  ) {  TextField(  value = email.value,  onValueChange = { email.value = it },  shape = RoundedCornerShape(24.dp),  label = { Text(text = stringResource(id = R.string.enter\_email)) },  placeholder = { Text(text = stringResource(id = R.string.enter\_email\_place\_holder)) },  leadingIcon = {  Icon(  imageVector = Icons.Default.Email,  contentDescription = stringResource(id = R.string.enter\_email)  )  },  keyboardOptions = KeyboardOptions(  keyboardType = KeyboardType.Email,  imeAction = ImeAction.Done  )  )  TextField(  value = password.value,  onValueChange = { password.value = it },  shape = RoundedCornerShape(24.dp),  label = { Text(text = stringResource(id = R.string.enter\_password)) },  placeholder = { Text(text = stringResource(id = R.string.enter\_password\_place\_holder)) },  leadingIcon = {  Icon(  imageVector = Icons.Default.Lock,  contentDescription = stringResource(id = R.string.enter\_password)  )  },  trailingIcon = {  IconButton(  onClick = { showPassword.value = !showPassword.value }  ) {  Icon(  imageVector = if (showPassword.value) Icons.Outlined.VisibilityOff  else Icons.Outlined.Visibility,  contentDescription = stringResource(id = R.string.password\_info)  )  }  },  visualTransformation = if (showPassword.value) VisualTransformation.None  else PasswordVisualTransformation()  )  TextField(  value = phone.value,  onValueChange = { phone.value = it },  shape = RoundedCornerShape(24.dp),  label = { Text(text = stringResource(id = R.string.enter\_phone)) },  placeholder = { Text(text = stringResource(id = R.string.enter\_phone\_place\_holder)) },  leadingIcon = {  Icon(  imageVector = Icons.Default.Phone,  contentDescription = stringResource(id = R.string.enter\_phone)  )  },  trailingIcon = {  Icon(  imageVector = Icons.Default.Info,  contentDescription = stringResource(id = R.string.enter\_phone)  )  },  keyboardOptions = KeyboardOptions(  keyboardType = KeyboardType.Phone,  imeAction = ImeAction.Done  )  )  TextField(  value = age,  onValueChange = {  age = it  isUnder18 = false  },  shape = RoundedCornerShape(24.dp),  label = { Text(text = stringResource(id = R.string.enter\_age)) },  placeholder = { Text(text = stringResource(id = R.string.enter\_age\_place\_holder)) },  leadingIcon = {  Icon(  imageVector = Icons.Default.Numbers,  contentDescription = stringResource(id = R.string.enter\_age)  )  },  isError = isUnder18,  keyboardOptions = KeyboardOptions(  keyboardType = KeyboardType.Number,  imeAction = ImeAction.Done  ),  keyboardActions = KeyboardActions(  onDone = {  isUnder18 = validateAge(inputText = age)  }  )  )  if(isUnder18) {  Text(  modifier = Modifier.padding(16.dp),  text = stringResource(id = R.string.must\_age),  color = MaterialTheme.colorScheme.error  )  }  Button(onClick = { }) {  Text(text = "Register")  }  }  }  }  private fun validateAge(inputText: String): Boolean {  return inputText.toInt() < 18  }  @Composable  fun LoginUI() {  var email by remember {  mutableStateOf("")  }  var password by remember {  mutableStateOf("")  }  var showPassword by remember {  mutableStateOf(false)  }  Column(  modifier = Modifier  .padding(10.dp)  .fillMaxSize()  .background(color = Pink80),  verticalArrangement = Arrangement.Center,  horizontalAlignment = Alignment.CenterHorizontally  ) {  TextField(  value = email,  onValueChange = { email = it },  shape = RoundedCornerShape(24.dp),  label = { Text(text = stringResource(id = R.string.enter\_email)) },  placeholder = { Text(text = stringResource(id = R.string.enter\_email\_place\_holder)) },  leadingIcon = {  Icon(  imageVector = Icons.Default.Email,  contentDescription = stringResource(  id = R.string.enter\_email  )  )  },  trailingIcon = {  Icon(  imageVector = Icons.Default.Info,  contentDescription = stringResource(  id = R.string.enter\_email  )  )  }  )  Spacer(modifier = Modifier.padding(10.dp))  OutlinedTextField(  value = password,  onValueChange = { password = it },  shape = RoundedCornerShape(24.dp),  label = { Text(text = stringResource(id = R.string.enter\_password)) },  placeholder = { Text(text = stringResource(id = R.string.enter\_password\_place\_holder)) },  leadingIcon = {  Icon(  imageVector = Icons.Default.Lock, contentDescription = stringResource(  id = R.string.enter\_password  )  )  },  trailingIcon = {  IconButton(onClick = { showPassword = !showPassword }) {  Icon(  imageVector = if (showPassword) Icons.Outlined.VisibilityOff  else Icons.Outlined.Visibility,  contentDescription = stringResource(id = R.string.password\_info)  )  }  },  visualTransformation = if (showPassword) VisualTransformation.None else PasswordVisualTransformation()  )  Spacer(modifier = Modifier.padding(20.dp))  OutlinedButton(onClick = { login(email, password) }) {  Text(text = "Login")  }  }  }  } | </end> |
| <hitle> | JetpackComposeBoxDiscussion | <chare> | 1 | <pext> | 03-17/JetpackComposeBoxDiscussion  class MainActivity : ComponentActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  BoxDemo1()  }  }  }  @Composable  private fun BoxDemo1() {  Box(  modifier = Modifier  .background(color = Color.Red)  .fillMaxSize()  ) {  Text(text = "TL",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.TopStart)  )  Text(text = "TE",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.TopEnd)  )  Text(text = "BS",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.BottomStart)  )  Text(text = "BE",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.BottomEnd)  )  Text(text = "TC",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.TopCenter)  )  Text(text = "BC",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.BottomCenter)  )  Text(text = "C",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.Center)  )  Text(text = "CS",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.CenterStart)  )  Text(text = "CE",  modifier = Modifier  .background(color = Color.Yellow)  .padding(10.dp)  .align(Alignment.CenterEnd)  )  }  }  @Composable  private fun BoxDemo4() {  Box(  modifier = Modifier  .background(color = Color.Red)  .size(200.dp, 200.dp)  ) {  Box(  modifier = Modifier  .background(color = Color.Blue)  .size(100.dp, 100.dp)  .align(Alignment.TopEnd)  ) {  }  Text(text = "Hiiiii",  modifier = Modifier  .background(color = Color.White)  .size(90.dp, 50.dp)  .align(Alignment.BottomCenter)  )  }  }  @Composable  private fun BoxDemo3() {  Box(  modifier = Modifier  .background(color = Color.Red)  .size(200.dp, 200.dp)  ) {  Box(  modifier = Modifier  .background(color = Color.Blue)  .size(100.dp, 100.dp)  ) {  }  Text(text = "Hiiiii",  modifier = Modifier  .background(color = Color.White)  .size(90.dp, 50.dp)  )  }  }  @Composable  private fun BoxDemo2() {  Box(  modifier = Modifier  .background(color = Color.Red)  .size(200.dp, 200.dp)  ) {}  }  @Composable  fun Greeting(name: String) {  Text(text = "Hello $name!")  }  @Preview(showBackground = true)  @Composable  fun DefaultPreview() {  JetpackCmposeBoxDiscussionTheme {  Greeting("Android")  }  } | </end> |
| <hitle> | CustomViewDemo Face | <chare> | 1 | <pext> | 03-20/CustomViewDemo Face  -------important things for custom view---------------  class MyCustomView(context: Context) : View(context) {  override fun onMeasure(widthMeasureSpec: Int, heightMeasureSpec: Int) {  super.onMeasure(widthMeasureSpec, heightMeasureSpec)  }  override fun onLayout(changed: Boolean, left: Int, top: Int, right: Int, bottom: Int) {  super.onLayout(changed, left, top, right, bottom)  }  override fun onDraw(canvas: Canvas?) {  super.onDraw(canvas)  }  }  -------extends View to customize------  class RatingBarExpressions(context: Context, attr: AttributeSet) : View(context, attr) {  private var size = 0  private var faceColor = DEFAULT\_FACE\_COLOR  private var eyeColor = DEFAULT\_EYES\_COLOR  private var mouthColor = DEFAULT\_MOUTH\_COLOR  private var borderColor = DEFAULT\_BORDER\_COLOR  private var borderWidth = DEFAULT\_BORDER\_WIDTH  private val paint = Paint()  private val mouthPath = Path()  init {  paint.isAntiAlias = true  setUpAttributes(attr)  }  var happyState = HAPPY  set(state) {  field = state  invalidate()  }  private fun setUpAttributes(attr: AttributeSet) {  val typeArray =  context.theme.obtainStyledAttributes(attr, R.styleable.RatingBarResultView, 0, 0)  happyState = typeArray.getInt(R.styleable.RatingBarResultView\_state, HAPPY.toInt()).toLong()  eyeColor = typeArray.getColor(R.styleable.RatingBarResultView\_eyeColor, DEFAULT\_EYES\_COLOR)  faceColor =  typeArray.getColor(R.styleable.RatingBarResultView\_faceColor, DEFAULT\_FACE\_COLOR)  mouthColor =  typeArray.getColor(R.styleable.RatingBarResultView\_mouthColor, DEFAULT\_MOUTH\_COLOR)  borderColor =  typeArray.getColor(R.styleable.RatingBarResultView\_borderColor, DEFAULT\_BORDER\_COLOR)  borderWidth = typeArray.getDimension(  R.styleable.RatingBarResultView\_borderWidth,  DEFAULT\_BORDER\_WIDTH  )  typeArray.recycle()  }  override fun onMeasure(widthMeasureSpec: Int, heightMeasureSpec: Int) {  super.onMeasure(widthMeasureSpec, heightMeasureSpec)  size = measuredWidth.coerceAtMost(measuredHeight)  setMeasuredDimension(size, size)  }  override fun draw(canvas: Canvas) {  super.draw(canvas)  drawFaceBackground(canvas)  drawEyes(canvas)  drawMouth(canvas)  }  private fun drawMouth(canvas: Canvas) {  mouthPath.reset()  mouthPath.moveTo(size \* 0.15f, size \* 0.65f)  if (happyState == HAPPY) {  mouthPath.quadTo(size \* 0.5f, size \* 0.75f, size \* 0.85f, size \* 0.65f)  mouthPath.quadTo(size \* 0.5f, size \* 0.98f, size \* 0.15f, size \* 0.65f)  }  if (happyState == SAD) {  mouthPath.quadTo(size \* 0.5f, size \* 0.65f, size \* 0.15f, size \* 0.65f)  mouthPath.quadTo(size \* 0.5f, size \* 0.55f, size \* 0.85f, size \* 0.65f)  }  if (happyState == JUST\_OKAY) {  mouthPath.quadTo(size \* 0.5f, size \* 0.60f, size \* 0.15f, size \* 0.65f)  mouthPath.quadTo(size \* 0.5f, size \* 0.60f, size \* 0.85f, size \* 0.65f)  }  paint.color = mouthColor  paint.style = Paint.Style.FILL  canvas.drawPath(mouthPath, paint)  }  private fun drawEyes(canvas: Canvas) {  paint.color = eyeColor  paint.style = Paint.Style.FILL  val leftEye = RectF(size \* 0.32f, size \* 0.23f, size \* 0.43f, size \* 0.50f)  canvas.drawOval(leftEye, paint)  val rightEye = RectF(size \* 0.57f, size \* 0.23f, size \* 0.68f, size \* 0.50f)  canvas.drawOval(rightEye, paint)  }  private fun drawFaceBackground(canvas: Canvas) {  paint.color = faceColor  paint.style = Paint.Style.FILL  val radius = size / 2f  canvas.drawCircle(size / 2f, size / 2f, radius, paint)  paint.color = borderColor  paint.style = Paint.Style.STROKE  paint.strokeWidth = borderWidth  canvas.drawCircle(size / 2f, size / 2f, radius - borderWidth / 2f, paint)  }  companion object {  private const val DEFAULT\_FACE\_COLOR = YELLOW  private const val DEFAULT\_EYES\_COLOR = BLACK  private const val DEFAULT\_MOUTH\_COLOR = BLACK  private const val DEFAULT\_BORDER\_COLOR = GRAY  private const val DEFAULT\_BORDER\_WIDTH = 4.0f  const val HAPPY = 0L  const val SAD = 1L  const val JUST\_OKAY = 2L  }  }  --------------use customview in 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"  xmlns:app="http://schemas.android.com/apk/res-auto"  tools:context=".MainActivity">  <com.example.customviewdemo.RatingBarExpressions  android:layout\_gravity="center"  android:layout\_width="200dp"  android:layout\_height="200dp"  app:eyeColor="#448AFF"  app:mouthColor = "#E36E95"  app:faceColor= "#FFFF00"  app:borderColor="#3E280C"  app:state="happy"  android:layout\_margin="20dp"  />  <com.example.customviewdemo.RatingBarExpressions  android:layout\_gravity="center"  android:layout\_width="200dp"  android:layout\_height="200dp"  app:eyeColor="#448AFF"  app:mouthColor = "#E36E95"  app:faceColor= "#D3B78D"  app:borderColor="#3E280C"  app:state="sad"  android:layout\_margin="20dp"  />  <com.example.customviewdemo.RatingBarExpressions  android:layout\_gravity="center"  android:layout\_width="200dp"  android:layout\_height="200dp"  app:eyeColor="#448AFF"  app:mouthColor = "#FFFF00"  app:faceColor= "#FF6E40"  app:borderColor="#3E280C"  app:state="okay"  />  </LinearLayout> | </end> |
| <hitle> | Prasad JPMorgan customview | <chare> | 1 | <pext> | 03-20/Prasad JPMorgan customview  class CircularProgressView @JvmOverloads constructor(  contx: Context,  val attrs: AttributeSet? = null,  val defStyleAttr: Int = 0  ) : View(contx, attrs, defStyleAttr) {  var progressColor: Int = Color.rgb(62, 152, 199)  set(value) {  field = value  initProgressPaint()  postInvalidate()  }  var progressStrokeWidth: Float = 16f  set(value) {  field = value  postInvalidate()  }  var progress: Int = 40  set(value) {  field = value  postInvalidate()  }  var progressTextSize: Float = 24f  set(value) {  field = value  initTextPaint()  postInvalidate()  }  var progressTextColor: Int = Color.rgb(255, 90, 96)  set(value) {  field = value  initTextPaint()  postInvalidate()  }  var pathColor: Int = Color.rgb(214, 214, 214)  set(value) {  field = value  initPathPaint()  postInvalidate()  }  var pathStrokeWidth: Float = 16f  set(value) {  field = value  postInvalidate()  }  var pathPaint: Paint = Paint()  var textPaint: Paint = Paint()  var progressPaint: Paint = Paint()  fun initPathPaint(){  pathPaint.apply {  strokeWidth = pathStrokeWidth  color = pathColor  style = Paint.Style.STROKE  isAntiAlias = true  }  }  fun initTextPaint() {  textPaint.apply {  color = progressTextColor  textSize = progressTextSize  isAntiAlias = true  textAlign = Paint.Align.CENTER  }  }  fun initProgressPaint() {  progressPaint.apply {  color = progressColor  strokeWidth = progressStrokeWidth  strokeCap = Paint.Cap.ROUND  isAntiAlias = true  style = Paint.Style.STROKE  }  }  init {  // Extract design time attributes init corresponding properties in this class  val styledAttr: TypedArray = context.obtainStyledAttributes(attrs, R.styleable.CircularProgressView, defStyleAttr, 0)  progressColor = styledAttr.getColor(R.styleable.CircularProgressView\_progress\_color, Color.rgb(62, 152, 199))  progressStrokeWidth = styledAttr.getDimension(R.styleable.CircularProgressView\_progress\_stroke\_width, 16f)  progress = styledAttr.getInteger(R.styleable.CircularProgressView\_progress, 40)  progressTextSize = styledAttr.getDimension(R.styleable.CircularProgressView\_progress\_text\_size, 24f)  progressTextColor = styledAttr.getColor(R.styleable.CircularProgressView\_progress\_text\_color, Color.rgb(255, 90, 96))  pathColor = styledAttr.getColor(R.styleable.CircularProgressView\_path\_color, Color.rgb(214, 214, 214))  pathStrokeWidth = styledAttr.getDimension(R.styleable.CircularProgressView\_path\_stroke\_width, 16f)  initPathPaint()  initTextPaint()  initProgressPaint()  }  override fun onMeasure(widthMeasureSpec: Int, heightMeasureSpec: Int) {  super.onMeasure(widthMeasureSpec, heightMeasureSpec)  setMeasuredDimension(widthMeasureSpec, heightMeasureSpec)  }  override fun onDraw(canvas: Canvas?) {  super.onDraw(canvas)  val cx = (width/2).toFloat()  val cy = (height/2).toFloat()  var min = if(width < height) width else height  val radius = (min/2).toFloat() - pathStrokeWidth/2  // Draw path  canvas?.drawCircle(cx, cy, radius, pathPaint)  canvas?.drawText("$progress%", cx, cy+(progressTextSize/3).toFloat(), textPaint)  val delta = (progressStrokeWidth/2).toFloat()  val sweepAngle = ((progress\*360)/100).toFloat()  var left = 0f+delta  var top = 0f+delta  var right = width.toFloat() - delta  var bottom = height.toFloat() - delta  if(width > height) {  val dif = (width - height) / 2  left = left + dif  right = right - dif  } else if(width < height) {  val dif = (height-width) / 2  top = top + dif  bottom = bottom - dif  }  canvas?.drawArc(left, top, right, bottom, 270f, sweepAngle, false, progressPaint)  }  override fun onSaveInstanceState(): Parcelable? {  val bundle = Bundle()  bundle.putParcelable(INSTANCE\_STATE, super.onSaveInstanceState())  bundle.putInt(PROGRESS, progress)  bundle.putInt(PROGRESS\_COLOR, progressColor)  bundle.putFloat(PROGRESS\_STROKE\_WIDTH, progressStrokeWidth)  bundle.putInt(TEXT\_COLOR, progressTextColor)  bundle.putFloat(TEXT\_SIZE, progressTextSize)  bundle.putInt(PATH\_COLOR, pathColor)  bundle.putFloat(PATH\_STROKE\_WIDTH, pathStrokeWidth)  return bundle  }  override fun onRestoreInstanceState(state: Parcelable?) {  if(state !is Bundle) {  return  }  val bundle = state as Bundle  progress = bundle.getInt(PROGRESS)  progressColor = bundle.getInt(PROGRESS\_COLOR)  progressStrokeWidth = bundle.getFloat(PROGRESS\_STROKE\_WIDTH)  progressTextColor = bundle.getInt(TEXT\_COLOR)  progressTextSize = bundle.getFloat(TEXT\_SIZE)  pathColor = bundle.getInt(PATH\_COLOR)  pathStrokeWidth = bundle.getFloat(PATH\_STROKE\_WIDTH)  super.onRestoreInstanceState(bundle.getParcelable(INSTANCE\_STATE))  }  companion object {  const val PROGRESS = "progress"  const val PROGRESS\_COLOR = "progress\_color"  const val PROGRESS\_STROKE\_WIDTH = "progress\_stroke\_width"  const val TEXT\_COLOR = "text\_color"  const val TEXT\_SIZE = "text\_size"  const val PATH\_COLOR = "path\_color"  const val PATH\_STROKE\_WIDTH = "path\_stroke\_width"  const val INSTANCE\_STATE = "saved\_instance"  }  }  class MainActivity : AppCompatActivity() {  lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnChangeProgress.setOnClickListener {  val randomProgress = Random.nextInt(0, 100)  binding.cpv.progress = randomProgress  Log.d("RandomProgress", "New Progress value = $randomProgress")  }  }  }  <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <com.aapolis.customviewdemo.CircularProgressView  android:layout\_width="250dp"  android:layout\_height="150dp"  android:id="@+id/cpv"  app:progress="75"  app:progress\_color="#3e98c7"  app:path\_color="#d6d6d6"  app:progress\_stroke\_width="16dp"  app:path\_stroke\_width="16dp"  app:progress\_text\_color="#ff5056"  app:progress\_text\_size="25sp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintLeft\_toLeftOf="parent"  app:layout\_constraintRight\_toRightOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <Button  android:id="@+id/btn\_change\_progress"  android:layout\_marginTop="16dp"  android:text="Set Random Progress"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/cpv"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | NavGraphDemo1 passdataBTWfragment | <chare> | 1 | <pext> | 03-21/NavGraphDemo1 passdataBTWfragment  ------------Gradle--------------  def nav\_version = "2.5.3"  // Java language implementation  implementation "androidx.navigation:navigation-fragment:$nav\_version"  implementation "androidx.navigation:navigation-ui:$nav\_version"  // Kotlin  implementation "androidx.navigation:navigation-fragment-ktx:$nav\_version"  implementation "androidx.navigation:navigation-ui-ktx:$nav\_version"  // Feature module Support  implementation "androidx.navigation:navigation-dynamic-features-fragment:$nav\_version"  // Testing Navigation  androidTestImplementation "androidx.navigation:navigation-testing:$nav\_version"  // Jetpack Compose Integration  implementation "androidx.navigation:navigation-compose:$nav\_version"  ---------------howto navigation graph----------------------  1. In ResourceManager, click menu +/Navigation Resource file, name it app\_flow.xml  2. Open navigation graph by double click app\_flow.xml  3. in navigation editor of app\_flow.xml, +/create new destination/add a fragment  4. Define actions by connect from one to another  5. Define animations  5.1. slide\_left.xml  <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android">  <translate  android:duration = "300"  android:fromXDelta="0%"  android:fromYDelta="0%"  android:toXDelta="100%"  android:toYDelta="0%" />  </set>  5.2. Slide\_right.xml  <set xmlns:android="http://schemas.android.com/apk/res/android">  <translate  android:duration = "300"  android:fromXDelta="100%"  android:fromYDelta="0%"  android:toXDelta="0%"  android:toYDelta="0%" />  </set>  6. Define animations in an action, in attribute or in xml  app:enterAnim="@anim/slide\_left"  app:exitAnim="@anim/slide\_right"  app:popEnterAnim="@anim/slide\_left"  app:popExitAnim="@anim/slide\_right"  7. Completed app\_flow.xml  <?xml version="1.0" encoding="utf-8"?>  <navigation xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/app\_flow"  app:startDestination="@id/splashUI">  <fragment  android:id="@+id/splashUI"  android:name="com.example.navgraphdemo1.SplashUI"  android:label="fragment\_splash\_u\_i"  tools:layout="@layout/fragment\_splash\_u\_i" >  <action  android:id="@+id/action\_splashUI\_to\_registrationUI"  app:destination="@id/registrationUI" />  <action  android:id="@+id/action\_splashUI\_to\_loginUI"  app:destination="@id/loginUI" />  <action  android:id="@+id/action\_splashUI\_to\_dashboardUI"  app:destination="@id/dashboardUI" />  </fragment>  <fragment  android:id="@+id/registrationUI"  android:name="com.example.navgraphdemo1.RegistrationUI"  android:label="fragment\_registration\_u\_i"  tools:layout="@layout/fragment\_registration\_u\_i" >  <action  android:id="@+id/action\_registrationUI\_to\_dashboardUI"  app:destination="@id/dashboardUI" />  </fragment>  <fragment  android:id="@+id/loginUI"  android:name="com.example.navgraphdemo1.LoginUI"  android:label="fragment\_login\_u\_i"  tools:layout="@layout/fragment\_login\_u\_i" >  <action  android:id="@+id/action\_loginUI\_to\_dashboardUI"  app:destination="@id/dashboardUI" />  </fragment>  <fragment  android:id="@+id/dashboardUI"  android:name="com.example.navgraphdemo1.DashboardUI"  android:label="fragment\_dashboard\_u\_i"  tools:layout="@layout/fragment\_dashboard\_u\_i" >  <action  android:id="@+id/action\_dashboardUI\_to\_categoryUI"  app:destination="@id/categoryUI" />  </fragment>  <fragment  android:id="@+id/categoryUI"  android:name="com.example.navgraphdemo1.CategoryUI"  android:label="fragment\_category\_u\_i"  tools:layout="@layout/fragment\_category\_u\_i" >  <action  android:id="@+id/action\_categoryUI\_to\_subCategoryUI"  app:destination="@id/subCategoryUI" />  </fragment>  <fragment  android:id="@+id/subCategoryUI"  android:name="com.example.navgraphdemo1.SubCategoryUI"  android:label="fragment\_sub\_category\_u\_i"  tools:layout="@layout/fragment\_sub\_category\_u\_i" >  <action  android:id="@+id/action\_subCategoryUI\_to\_placeOrderUI"  app:destination="@id/placeOrderUI" />  </fragment>  <fragment  android:id="@+id/placeOrderUI"  android:name="com.example.navgraphdemo1.PlaceOrderUI"  android:label="fragment\_place\_order\_u\_i"  tools:layout="@layout/fragment\_place\_order\_u\_i" >  <action  android:id="@+id/action\_placeOrderUI\_to\_dashboardUI"  app:destination="@id/dashboardUI"  app:enterAnim="@anim/slide\_left"  app:exitAnim="@anim/slide\_right"  app:popEnterAnim="@anim/slide\_left"  app:popExitAnim="@anim/slide\_right" />  </fragment>  </navigation>  -----------howto navhost------------------------  Add navhost to MainActivity  In XML design, Drag and Drop Containers/NavHostFragment  By XML it looks like this  <?xml version="1.0" encoding="utf-8"?>  <androidx.fragment.app.FragmentContainerView xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/fragmentContainerView"  android:name="androidx.navigation.fragment.NavHostFragment"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  app:defaultNavHost="true"  app:navGraph="@navigation/app\_flow"  tools:layout\_editor\_absoluteX="1dp"  tools:layout\_editor\_absoluteY="1dp" />  --------------howto navigate-------------------  In code, run navigation by  binding.btnNext.setOnClickListener {  findNavController().navigate(R.id.action\_placeOrderUI\_to\_dashboardUI)  }  --------Pass data between destinations SafeArgs-----------------  1. Add the argment (Arguments +) to the target desgination which receives argments  In NavGraph  <fragment  android:id="@+id/subCategoryUI"  android:name="com.example.navgraphdemo1.SubCategoryUI"  android:label="fragment\_sub\_category\_u\_i"  tools:layout="@layout/fragment\_sub\_category\_u\_i" >  <action  android:id="@+id/action\_subCategoryUI\_to\_placeOrderUI"  app:destination="@id/placeOrderUI" />  <argument  android:name="category\_id"  app:argType="integer"  android:defaultValue="-1" />  </fragment>  2. Override a destination argument in the target action  <fragment  android:id="@+id/categoryUI"  android:name="com.example.navgraphdemo1.CategoryUI"  android:label="fragment\_category\_u\_i"  tools:layout="@layout/fragment\_category\_u\_i" >  <action  android:id="@+id/action\_categoryUI\_to\_subCategoryUI"  app:destination="@id/subCategoryUI" >  <argument  android:name="category\_id"  app:argType="integer"  android:defaultValue="-1" />  </action>  </fragment>  3. Pass data in the start destination  binding.btnNext.setOnClickListener {  // val bundle = Bundle()  // bundle.putInt("category\_id", 20)  // findNavController().navigate(R.id.action\_categoryUI\_to\_subCategoryUI, bundle)  val action = CategoryUIDirections.actionCategoryUIToSubCategoryUI(30)  findNavController().navigate(action)  }  4. Receive data in the target destination  class SubCategoryUI : Fragment() {  private lateinit var binding: FragmentSubCategoryUIBinding  private val args: SubCategoryUIArgs by navArgs()  override fun onCreateView(  inflater: LayoutInflater, container: ViewGroup?,  savedInstanceState: Bundle?  ): View? {  binding = FragmentSubCategoryUIBinding.inflate(layoutInflater, container, false)  return binding.root  }  override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)  // val categoryId = arguments?.getInt("category\_id")  val categoryId = args.categoryId.toString()  binding.btnNext.text = categoryId.toString()  binding.btnNext.setOnClickListener {  findNavController().navigate(R.id.action\_subCategoryUI\_to\_placeOrderUI)  }  }  } | </end> |
| <hitle> | NavGraphBottomAppbarDemo | <chare> | 1 | <pext> | 03-21/NavGraphBottomAppbarDemo  ------------Gradle--------------  def nav\_version = "2.5.3"  // Java language implementation  implementation "androidx.navigation:navigation-fragment:$nav\_version"  implementation "androidx.navigation:navigation-ui:$nav\_version"  // Kotlin  implementation "androidx.navigation:navigation-fragment-ktx:$nav\_version"  implementation "androidx.navigation:navigation-ui-ktx:$nav\_version"  // Feature module Support  implementation "androidx.navigation:navigation-dynamic-features-fragment:$nav\_version"  // Testing Navigation  androidTestImplementation "androidx.navigation:navigation-testing:$nav\_version"  // Jetpack Compose Integration  implementation "androidx.navigation:navigation-compose:$nav\_version"  -------------app\_menu.xml---------------------------  <?xml version="1.0" encoding="utf-8"?>  <menu xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:android="http://schemas.android.com/apk/res/android">  --------------menu ids should be fragment/screen ids------------  <item  android:id="@+id/homeUI"  android:title="Home"  android:icon="@drawable/baseline\_home\_24"  />  <item  android:id="@+id/accountUI"  android:title="Account"  android:icon="@drawable/baseline\_manage\_accounts\_24"  />  <item  android:id="@+id/orderUI"  android:title="Order"  android:icon="@drawable/baseline\_manage\_history\_24"  />  </menu>  -------------Navigation Graph---------------------------  1. Create navigation graph features\_flow.xml  2. Create destinations homeUI, accountUI, orderUI  <?xml version="1.0" encoding="utf-8"?>  <navigation xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/features\_flow"  app:startDestination="@id/homeUI">  <fragment  android:id="@+id/homeUI"  android:name="com.example.navgraphbottomappbardemo.HomeUI"  android:label="Home UI"  tools:layout="@layout/fragment\_home\_u\_i" />  <fragment  android:id="@+id/accountUI"  android:name="com.example.navgraphbottomappbardemo.AccountUI"  android:label="Account UI"  tools:layout="@layout/fragment\_account\_u\_i" />  <fragment  android:id="@+id/orderUI"  android:name="com.example.navgraphbottomappbardemo.OrderUI"  android:label="Order UI"  tools:layout="@layout/fragment\_order\_u\_i" />  </navigation>  -------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <com.google.android.material.bottomnavigation.BottomNavigationView  android:id="@+id/bottomNavigationView"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:menu="@menu/app\_menu"  />  <androidx.fragment.app.FragmentContainerView  android:id="@+id/fragmentContainerView"  android:name="androidx.navigation.fragment.NavHostFragment"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  app:defaultNavHost="true"  app:navGraph="@navigation/features\_flow"  tools:layout\_editor\_absoluteX="1dp"  tools:layout\_editor\_absoluteY="1dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toTopOf="@id/bottomNavigationView"  />  </androidx.constraintlayout.widget.ConstraintLayout>  --------------MainActivity--------------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var navController: NavController  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setupNavController()  }  private fun setupNavController() {  navController = binding.fragmentContainerView.getFragment<NavHostFragment>().navController  -------------Important to have the same menu=fragment id---------  setupActionBarWithNavController(navController, AppBarConfiguration(setOf(R.id.homeUI, R.id.accountUI, R.id.orderUI)))  binding.bottomNavigationView.setupWithNavController(navController)  }  } | </end> |
| <hitle> | NavigationDrawByNavGraph | <chare> | 1 | <pext> | 03-21/NavigationDrawByNavGraph  ------------Gradle--------------  def nav\_version = "2.5.3"  // Java language implementation  implementation "androidx.navigation:navigation-fragment:$nav\_version"  implementation "androidx.navigation:navigation-ui:$nav\_version"  // Kotlin  implementation "androidx.navigation:navigation-fragment-ktx:$nav\_version"  implementation "androidx.navigation:navigation-ui-ktx:$nav\_version"  // Feature module Support  implementation "androidx.navigation:navigation-dynamic-features-fragment:$nav\_version"  // Testing Navigation  androidTestImplementation "androidx.navigation:navigation-testing:$nav\_version"  // Jetpack Compose Integration  implementation "androidx.navigation:navigation-compose:$nav\_version"  -------------app\_menu.xml---------------------------  <?xml version="1.0" encoding="utf-8"?>  <menu xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:android="http://schemas.android.com/apk/res/android">  <group android:checkableBehavior="single">  <item  android:id="@+id/homeUI"  android:title="Home"  android:icon="@drawable/baseline\_home\_24"  />  <item  android:id="@+id/accountUI"  android:title="Account"  android:icon="@drawable/baseline\_manage\_accounts\_24"  />  <item  android:id="@+id/orderUI"  android:title="Order"  android:icon="@drawable/baseline\_manage\_history\_24"  />  </group>  </menu>  -------------Navigation Graph---------------------------  1. Create navigation graph app\_draw.xml  2. Create destinations homeUI, accountUI, orderUI  <?xml version="1.0" encoding="utf-8"?>  <navigation xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:id="@+id/app\_drawer"  app:startDestination="@id/homeUI">  <fragment  android:id="@+id/homeUI"  android:name="com.example.navigationdrawbynavgraph.HomeUI"  android:label="Home"  tools:layout="@layout/fragment\_home\_u\_i" />  <fragment  android:id="@+id/accountUI"  android:name="com.example.navigationdrawbynavgraph.AccountUI"  android:label="Account"  tools:layout="@layout/fragment\_account\_u\_i" />  <fragment  android:id="@+id/orderUI"  android:name="com.example.navigationdrawbynavgraph.OrderUI"  android:label="Order"  tools:layout="@layout/fragment\_order\_u\_i" />  </navigation>  -------------activity\_main.xml-----------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.drawerlayout.widget.DrawerLayout  xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:id="@+id/drawerLayout"  >  <androidx.fragment.app.FragmentContainerView  android:id="@+id/fragmentContainerView"  android:name="androidx.navigation.fragment.NavHostFragment"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  app:defaultNavHost="true"  app:navGraph="@navigation/app\_drawer"  tools:layout\_editor\_absoluteX="1dp"  tools:layout\_editor\_absoluteY="1dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent"  />  ------------- important layer order in xml----------------  <com.google.android.material.navigation.NavigationView  app:menu="@menu/app\_menu"  android:id="@+id/navigationView"  android:layout\_width="wrap\_content"  android:layout\_height="match\_parent"  android:layout\_gravity="start"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  />  </androidx.drawerlayout.widget.DrawerLayout>  --------------MainActivity--------------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var listener: NavController.OnDestinationChangedListener  private lateinit var navController: NavController  private lateinit var appBarConfiguration: AppBarConfiguration  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setupNavController()  }  private fun setupNavController() {  navController = binding.fragmentContainerView.getFragment<NavHostFragment>().navController  appBarConfiguration = AppBarConfiguration( navController.graph, binding.drawerLayout )  binding.navigationView.setupWithNavController(navController)  setupActionBarWithNavController(navController, appBarConfiguration)  listener = NavController.OnDestinationChangedListener { controller, destination, arguments ->  if(destination.id == R.id.homeUI) {  supportActionBar?.setBackgroundDrawable(ColorDrawable(getColor(R.color.teal\_200)))  } else if (destination.id == R.id.accountUI) {  supportActionBar?.setBackgroundDrawable(ColorDrawable(getColor(R.color.teal\_700)))  } else if (destination.id == R.id.orderUI) {  supportActionBar?.setBackgroundDrawable(ColorDrawable(getColor(R.color.purple\_200)))  }  }  }  override fun onSupportNavigateUp(): Boolean {  navController = binding.fragmentContainerView.getFragment<NavHostFragment>().navController  return navController.navigateUp(appBarConfiguration) || super.onSupportNavigateUp()  }  override fun onResume() {  super.onResume()  navController.addOnDestinationChangedListener(listener)  }  override fun onPause() {  super.onPause()  navController.removeOnDestinationChangedListener(listener)  }  } | </end> |
| <hitle> | CoroutineDemo1 | <chare> | 1 | <pext> | 03-22/CoroutineDemo1  ------------Gradle--------------  implementation 'org.jetbrains.kotlinx:kotlinx-coroutines-android:1.3.9'  ------------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/text\_view"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_centerInParent="true"  android:text="0"  />  <Button  android:id="@+id/button\_1"  android:layout\_centerInParent="true"  android:layout\_below="@id/text\_view"  android:onClick="updateCounter"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Update Counter"  />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Excute Task"  android:onClick="doAction"  android:layout\_below="@id/button\_1"  android:layout\_centerInParent="true"  />  </RelativeLayout>  -----------------MainActivity--------------------  class MainActivity : AppCompatActivity() {  lateinit var counterText: TextView  @SuppressLint("MissingInflatedId")  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  counterText = findViewById(R.id.text\_view)  Log.d("abc", "${Thread.currentThread().name}")  }  fun updateCounter(view: View) {  Log.d("abc", "${Thread.currentThread().name}")  counterText.text = "${counterText.text.toString().toInt() + 1}"  }  private fun executeLongRunningTask() {  for (i in 1..100000000000L) {  }  }  fun doAction(view: View) {  // thread (start = true) {  // executeLongRunningTask()  // }  CoroutineScope(Dispatchers.IO).launch {  Log.d("abc", "1- ${Thread.currentThread().name}")  }  GlobalScope.launch(Dispatchers.Main) {  Log.d("abc", "2- ${Thread.currentThread().name}")  }  MainScope().launch(Dispatchers.Default) {  Log.d("abc", "3- ${Thread.currentThread().name}")  }  }  }  -------------Running Result---------------------  abc main  abc 2- main  abc 3- DefaultDispatcher-worker-2  abc 1- DefaultDispatcher-worker-2 | </end> |
| <hitle> | CoroutineMVVMDemo2 RoomDB | <chare> | 1 | <pext> | 03-22/CoroutineMVVMDemo2 Room  -----------------Gradle------------------  plugins {  id 'kotlin-kapt'  }  dependencies {  //Room database  def room\_version = "2.5.1"  implementation("androidx.room:room-runtime:2.5.1")  kapt "androidx.room:room-compiler:2.5.1"  implementation("androidx.room:room-ktx:$room\_version")  implementation("org.jetbrains.kotlinx:kotlinx-coroutines-android:1.6.4")  implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.6.0'  }  -----------------model/local/AppDatabase-----------------  @Database(entities = [Blog::class], version = 1, exportSchema = false)  abstract class AppDatabase : RoomDatabase() {  abstract fun getBlogDao(): BlogDao?  companion object {  private var INSTANCE: AppDatabase? = null  fun getInstance(context: Context): AppDatabase? {  if (INSTANCE == null) {  INSTANCE = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "BlogDB"  ).build()  }  return INSTANCE  }  }  }  -----------------model/local/Blog-----------------  @Entity(tableName = "Blog")  data class Blog(@PrimaryKey @ColumnInfo(name = "title") var title: String)  -----------------model/local/BlogDao-----------------  @Dao  interface BlogDao {  @Insert  suspend fun insert(blog: Blog)  @Delete  suspend fun delete(blog: Blog)  @Query("SELECT \* FROM Blog")  suspend fun getAllBlogs(): List<Blog>  }  -----------------view/adapter/BlogAdapter-----------------  class BlogAdapter(  val viewModel: BlogViewModel,  val arrayList: List<Blog>,  private val context: Context  ) : RecyclerView.Adapter<BlogAdapter.BlogViewHolder>() {  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): BlogViewHolder {  var root = LayoutInflater.from(parent.context).inflate(R.layout.item\_view, parent, false)  return BlogViewHolder(root)  }  override fun onBindViewHolder(holder: BlogViewHolder, position: Int) {  holder.bind(arrayList[position])  }  override fun getItemCount(): Int {  if (arrayList.isEmpty()) {  Toast.makeText(context, "No blog", Toast.LENGTH\_SHORT).show()  }  return arrayList.size  }  inner class BlogViewHolder(private val binding: View) : RecyclerView.ViewHolder(binding) {  fun bind(blog: Blog) {  val text1 = binding.findViewById<TextView>(R.id.txt\_blog)  val image = binding.findViewById<ImageButton>(R.id.delete)  text1.text = blog.title  image.setOnClickListener {  viewModel.delete(blog)  notifyItemRemoved(arrayList.indexOf(blog))  }  }  }  }  -----------------view/MainActivity-----------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var mainViewModel: BlogViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  mainViewModel = ViewModelProvider(this)[BlogViewModel::class.java]  binding.blogRecycler.layoutManager = LinearLayoutManager(this)  binding.btnBlog.setOnClickListener {  addBlog()  }  observeData()  }  private fun observeData() {  mainViewModel.allBlogs.observe(this) {  binding.blogRecycler.adapter = BlogAdapter(mainViewModel, it, this)  }  }  private fun addBlog() {  binding.apply {  mainViewModel.insert(Blog(titleBlog.text.toString()))  titleBlog.text?.clear()  }  }  }  fun main() = runBlocking {  repeat(5000000) {  delay(500L)  println("Power of coroutines")  }  }  -----------------viewmodel/BlogViewModel-----------------  class BlogViewModel(application: Application) : AndroidViewModel(application) {  var allBlogs: MutableLiveData<List<Blog>> = MutableLiveData()  val dao = application.let { AppDatabase.getInstance(it) }?.getBlogDao()  fun insert(blog: Blog) {  viewModelScope.launch(Dispatchers.IO) {  dao?.insert(blog)  }  getAllBlog()  }  fun delete(blog: Blog) {  viewModelScope.launch(Dispatchers.IO) {  dao?.delete(blog)  }  }  fun getAllBlog(): LiveData<List<Blog>> {  viewModelScope.launch(Dispatchers.IO) {  allBlogs.postValue(dao?.getAllBlogs())  }  return allBlogs  }  }  -----------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".view.MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/title\_blog"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:gravity="center"  android:hint="@string/enter\_blog"  app:layout\_constraintTop\_toTopOf="parent" />  <Button  android:id="@+id/btn\_blog"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Add"  app:layout\_constraintTop\_toBottomOf="@+id/title\_blog" />  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/blogRecycler"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_margin="10dp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/btn\_blog">  </androidx.recyclerview.widget.RecyclerView>  </androidx.constraintlayout.widget.ConstraintLayout>  -----------------item\_view.xml-----------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView 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\_margin="5dp"  app:cardCornerRadius="10dp"  app:cardElevation="10dp"  android:layout\_height="wrap\_content">  <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="wrap\_content"  android:background="#EAE4E4"  android:padding="10dp">  <TextView  android:id="@+id/txt\_blog"  android:textColor="@color/black"  android:textSize="30sp"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <ImageButton  android:id="@+id/delete"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:clickable="true"  android:focusable="true"  android:src="@drawable/ic\_baseline\_delete\_24"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView> | </end> |
| <hitle> | CoroutineMVVMRxJavaDemo3 Dog | <chare> | 1 | <pext> | 03-22/CoroutineMVVMRxJavaDemo3 Dog  ------------------Gradle-------------------  // RxJava 2  implementation "io.reactivex.rxjava2:rxjava:2.2.7"  implementation "io.reactivex.rxjava2:rxandroid:2.1.1"  //RxJava2 with Retrofit  implementation "com.squareup.retrofit2:adapter-rxjava2:2.9.0"  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  //Picasso for Image Loading  implementation 'com.squareup.picasso:picasso:2.71828'  //Coroutines  implementation("org.jetbrains.kotlinx:kotlinx-coroutines-android:1.6.4")  implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.6.1'  ------------------model/remote/ApiService------------------  interface ApiService {  @GET(END\_POINT)  suspend fun getRandomDog(): Response<DogResponse>  }  ------------------model/remote/Constants------------------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  const val END\_POINT = "random"  }  ------------------model/remote/DogResponse------------------  data class DogResponse(  val message: String,  val status: String  )  ------------------model/remote/RetrofitBuilder------------------  object RetrofitBuilder {  private lateinit var retrofit: Retrofit  fun getRetrofit(): Retrofit {  if (!this::retrofit.isInitialized) {  retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .addConverterFactory(GsonConverterFactory.create())  .build()  }  return retrofit  }  }  ------------------view/MainActivity------------------  class MainActivity : AppCompatActivity() {  private lateinit var viewModel: DogViewModel  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpViewModel()  }  private fun setUpViewModel() {  viewModel = ViewModelProvider(this)[DogViewModel::class.java]  viewModel.dogResponse.observe(this) {  Picasso.get()  .load(it.message)  .into(binding.dogImage)  }  viewModel.error.observe(this) {  Toast.makeText(this, it.toString(), Toast.LENGTH\_SHORT).show()  }  binding.btnRequest.setOnClickListener {  viewModel.getRandomDog()  }  }  }  ------------------viewmodel/DogViewModel------------------  class DogViewModel : ViewModel() {  val dogResponse = MutableLiveData<DogResponse>()  val error = MutableLiveData<String>()  private lateinit var retrofit: Retrofit  private lateinit var apiService: ApiService  private val handler = CoroutineExceptionHandler { \_, exception ->  error.postValue(exception.message.toString())  }  fun getRandomDog() {  retrofit = RetrofitBuilder.getRetrofit()  apiService = retrofit.create(ApiService::class.java)  viewModelScope.launch(Dispatchers.IO + handler) {  val result = apiService.getRandomDog()  if (result.isSuccessful) {  dogResponse.postValue(result.body())  } else {  error.postValue(result.message().toString())  }  }  }  } | </end> |
| <hitle> | CoroutineAsyncAwait | <chare> | 1 | <pext> | 03-23/CoroutineAsyncAwait  ------------Await understand-----------------  fun main() {  runBlocking {  println("0s")  for (i in 1L..10) {  launch { myTimer(i) }  }  val job1 = async { getAccessToken() } // taking 3s  val job2 = async { getApiKey() } // taking 1s  println("Main- not waiting async")  // delay(10000)  val resultFromJob1 = job1.await()  println("Main- waiting getAccessToken done")  val resultFromJob2 = job2.await()  println("Main- getApiKey done already but waiting getAccessToken done")  val userInfo = getUserInfo(resultFromJob1, resultFromJob2) //taking 5s  println("Main- waiting getUserInfo done because it's not async call")  }  }  private suspend fun myTimer(i: Long) {  delay(i\*1000)  println("${i}s")  }  private suspend fun getAccessToken(): String {  delay(3000)  val value = "iajdsfjkgyhiojsdfoisjdfoijsoij45697345jojdf"  println("getAccessToken done")  return value  }  private suspend fun getApiKey(): String {  delay(1000)  val value = "api\_key\_98723984234"  println("getApiKey done")  return value  }  private suspend fun getUserInfo(apiKey: String, accessToken: String){  delay(5000)  println("Thank you for api key $apiKey and access token $accessToken")  println("getUserInfo done")  }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  0s  Main- not waiting async  1s  getApiKey done  2s  3s  getAccessToken done  Main- waiting getAccessToken done  Main- getApiKey done already but waiting getAccessToken done  4s  5s  6s  7s  8s  Thank you for api key iajdsfjkgyhiojsdfoisjdfoijsoij45697345jojdf and access token api\_key\_98723984234  getUserInfo done  Main- waiting getUserInfo done because it's not async call  9s  10s  --------------------------Async vs Launch------------------  fun main() {  runBlocking {  println("Timer starts at 0s")  for (i in 1L..10) {  val job = async { myAsyncTimer(i) }  val anyhowWait = job.await()  }  }  }  suspend fun myAsyncTimer(i: Long): String {  delay(1000)  println("${i}s")  return ""  }  fun main() {  runBlocking {  println("Timer starts at 0s")  for (i in 1L..10) {  launch { myLaunchTimer(i) }  }  }  }  suspend fun myLaunchTimer(i: Long) {  delay(i\*1000)  println("${i}s")  } | </end> |
| <hitle> | CoroutinesScopes join/cancel | <chare> | 1 | <pext> | 03-23/CoroutinesScopes  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  val scope = CoroutineScope(CoroutineName("My Custom scope"))  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnNext.setOnClickListener {  startActivity(Intent(this, SecondActivity::class.java))  finish()  }  playWithScope()  }  private fun playWithScope1() {  scope.launch {  Log.i("pmh", this.coroutineContext.toString())  //CoroutineName(My Custom scope), StandaloneCoroutine{Active}@f8ffcae, Dispatchers.Default  }  }  private fun playWithScope2() {  GlobalScope.launch {  while (true) {  delay(1000L)  Log.i("pmh", "Global scope is running")  }  }  lifecycleScope.launch {  while (true) {  delay(1000L)  Log.i("pmh", "lifecycle scope is running")  }  }  //When the second activity is launched, Global scope is running still, but lifecycle scope is stopped  }  private fun playWithScope3() {  scope.launch {  val job1 = launch {  while (true) {  Log.i("pmh", "Job 1 is running - immediate line")  delay(1000L)  Log.i("pmh", "Job 1 is running - this line run a bit late") // this line is never run  }  }  Log.i("pmh", "Job1 cancelling")  job1.cancel()  Log.i("pmh", "Job1 cancelled")  }  //Job1 cancelling  //Job1 cancelled  //Job1 is running - immediate line  //It means when cancel is called, between cancelling and cancelled, there's some time taking that we may expect some of code in job1 is run.  }  private fun playWithScope4() {  scope.launch {  val job1 = launch {  while (true) {  Log.i("pmh", "Job 1 is running")  }  }  Log.i("pmh", "Job1 cancelling")  job1.cancel()  Log.i("pmh", "Job1 cancelled")  }  // This will make dangerous result - infinite loop because of cancelling time  }  private fun playWithScope5() {  scope.launch {  val job1 = launch {  while (true) {  Log.i("pmh", "Job 1 is running - immediate line")  delay(1000L)  Log.i("pmh", "Job 1 is running - this line run a bit late") // this line is never run  }  }  Log.i("pmh", "Job1 cancelling")  job1.cancel()  job1.join()  Log.i("pmh", "Job1 cancelled")  }  //Job1 cancelling  //Job1 is running - immediate line  //Job1 cancelled  //difference with playWithScope3 is when join is called, it goes to job1 and try to run and complete the job1.  //but while trying to complete job1, job1 already got notified to cancel and it's cancelled in the middle of running job1  }  private fun playWithScope6() {  scope.launch {  val job1 = launch {  while (isActive) {  Log.i("pmh", "Job 1 is running")  }  }  Log.i("pmh", "Job1 cancelling")  job1.cancel()  job1.join()  Log.i("pmh", "Job1 cancelled")  }  //Job1 cancelling  //Job1 is running - this line is called several times  //Job1 cancelled  //To completely stop job1 before running, we may use isActive.  //but still it's not the perfect solution  }  private fun playWithScope7() {  scope.launch {  val job1 = launch {  while (true) {  ensureActive()  Log.i("pmh", "Job 1 is running")  }  }  Log.i("pmh", "Job1 cancelling")  job1.cancel()  job1.join()  Log.i("pmh", "Job1 cancelled")  }  //Job1 cancelling  //Job1 is running - It's never called or less times  //Job1 cancelled  //ensureActive() is more reliable  }  private fun playWithScope8() {  scope.launch {  val job1 = launch {  while (true) {  ensureActive()  Log.i("pmh", "Job 1 is running")  }  }  Log.i("pmh", "Job1 cancelling")  job1.cancelAndJoin()  Log.i("pmh", "Job1 cancelled")  }  //Job1 cancelling - we can use cancelAndJoin()  //Job1 is running - It's never called or less times  //Job1 cancelled  //ensureActive() is more reliable  }  private fun playWithScope() {  val mainScope = scope.launch {  val job1 = launch {  while (true) {  ensureActive()  Log.i("pmh", "Job 1 is running")  }  }  val job2 = launch {  while(true) {  ensureActive()  Log.i("pmh", "Job2 is running")  }  }  }  runBlocking {  delay(20L)  Log.i("pmh", "mainScope cancelling")  mainScope.cancelAndJoin()  Log.i("pmh", "mainScope cancelled")  }    //we can use scope cancellation  //Job1 is running - several times  //Job2 is running - several times  //mainScope cancelling  //Job1 is running or Job2 is running - still mainScope is not completely cancelled  //mainScope cancelled  }  override fun onPause() {  super.onPause()  Log.i("pmh", "onPause called for MainActivity")  }  override fun onStop() {  super.onStop()  Log.i("pmh", "onPause called for onStop")  }  override fun onDestroy() {  super.onDestroy()  Log.i("pmh", "onPause called for onDestroy")  }  } | </end> |
| <hitle> | SharedFlowDemo1 Coroutine | <chare> | 1 | <pext> | 03-23/SharedFlowDemo1 CoroutineFlow  class MainActivity : AppCompatActivity() {  private lateinit var binding:ActivityMainBinding  private lateinit var flow: Flow<Int>  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  CoroutineScope(Dispatchers.Main).launch {  flow.collect {  Log.i("pmh", it.toString())  }  }  makeEmmiter()  }  private fun makeEmmiter() {  flow = flow {  Log.i("pmh", "Flow started")  (0..20).forEach {  delay(500L)  emit(it)  Log.i("pmh", "Flow emmited $it")  }  }.flowOn(Dispatchers.Default)  }    //Flow started  //Flow emmited 0  //Flow emmited 1  //0  //1  //2  //Flow emmited 2  //3  //Flow emmited 3  //...  //20  //Flow emmited 20  } | </end> |
| <hitle> | UnitTestingDemo1 Basic | <chare> | 1 | <pext> | 03-24/ValidateRegistrationTest  object ValidateRegistration {  private val existingUsers = listOf("Thomas", "Joshua", "Luan", "Alex", "TEST")  fun isValidationInput(  userName: String,  password: String,  confirmPassword: String,  mobileNumber: String  ): Boolean {  if (userName.isEmpty() || password.isEmpty() || confirmPassword.isEmpty()) {  return false  }  if(userName in existingUsers) {  return false  }  if(password != confirmPassword) {  return false  }  if(password.length < 6) {  return false  }  if(mobileNumber.isEmpty()) {  return false  }  if(mobileNumber.length != 10) {  return false  }  return true  }  }  -------------Create Test-----------------  On class name right click and choose Generate/Test…  In the test file, change Jupiter to Junit  internal class ValidateRegistrationTest {  @Before  fun setUp() {  // init work  }  @After  fun tearDown() {  //tear down  }  @Test  fun `When username is empty THEN return False`() {  val output = ValidateRegistration.isValidationInput(  EMPTY\_STRING,  TEST\_PASSWORD,  TEST\_CONFIRM\_PASSWORD,  TEST\_MOBILE\_NUMBER  )  assertFalse(output)  }  @Test  fun `When password is empty THEN return False`() {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  EMPTY\_STRING,  TEST\_CONFIRM\_PASSWORD,  TEST\_MOBILE\_NUMBER  )  assertFalse(output)  }  @Test  fun `When confirmPassword is empty THEN return False`() {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  TEST\_PASSWORD,  EMPTY\_STRING,  TEST\_MOBILE\_NUMBER  )  assertFalse(output)  }  @Test  fun `When username is existing user THEN return False`() {  val output = ValidateRegistration.isValidationInput(  EXISTING\_USER,  TEST\_PASSWORD,  EMPTY\_STRING,  TEST\_MOBILE\_NUMBER  )  assertFalse(output)  }  @Test  fun `When password & confirmPassword not same THEN return False`() {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  TEST\_PASSWORD,  TEST\_CONFIRM\_PASSWORD,  TEST\_MOBILE\_NUMBER  )  assertFalse(output)  }  @Test  fun `When password length is less than 6 THEN return False` () {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  TEST\_PASSWORD,  TEST\_CONFIRM\_PASSWORD,  TEST\_MOBILE\_NUMBER  )  assertFalse(output)  }  @Test  fun `When mobileNumber is empty THEN return False` () {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  TEST\_PASSWORD,  TEST\_CONFIRM\_PASSWORD,  EMPTY\_STRING  )  assertFalse(output)  }  @Test  fun `When mobileNumber length is less than 10 THEN return False` () {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  TEST\_PASSWORD,  TEST\_CONFIRM\_PASSWORD,  Mobile\_NUMBER\_ERROR\_CASE1  )  assertFalse(output)  }  @Test  fun `When mobileNumber length is greater than 10 THEN return False` () {  val output = ValidateRegistration.isValidationInput(  TEST\_USER\_NAME,  TEST\_PASSWORD,  TEST\_CONFIRM\_PASSWORD,  Mobile\_NUMBER\_ERROR\_CASE2  )  assertFalse(output)  }  private companion object {  const val TEST\_USER\_NAME = "test"  const val TEST\_PASSWORD = "12345"  const val TEST\_CONFIRM\_PASSWORD = "12345"  const val TEST\_MOBILE\_NUMBER = "2233342223"  const val EMPTY\_STRING = ""  const val EXISTING\_USER = "TEST"  const val Mobile\_NUMBER\_ERROR\_CASE1 = "223435"  const val Mobile\_NUMBER\_ERROR\_CASE2 = "223435234534"  }  } | </end> |
| <hitle> | MVVMCalculatorUnitTesting SimpleViewModel | <chare> | 1 | <pext> | 03-24/MVVMCalculatorUnitTesting Simple ViewModel  ------------Gradle------------------  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  testImplementation 'org.mockito:mockito-inline:3.9.0'  testImplementation "org.mockito.kotlin:mockito-kotlin:4.0.0"  androidTestImplementation "androidx.test:rules:1.3.0"  testImplementation 'androidx.arch.core:core-testing:2.1.0'  ------------------model/Input------------------  data class Input(  val a: Int,  val b: Int  )  ------------------viewmodel/CalculatorViewModel------------------  class CalculatorViewModel: ViewModel() {  val result = MutableLiveData<Int>()  fun add(input: Input) = with(input) {  result.value = a + b  }  fun sub(input: Input) = with(input) {  result.value = a - b  }  fun mul(input: Input) = with(input) {  result.value = a \* b  }  fun div(input: Input) = with(input) {  result.value = a / b  }  }  ------------------\*unittesting(test)/viewmodel/CalculatorViewModelTest------------------  @RunWith(MockitoJUnitRunner::class)  class CalculatorViewModelTest {  @get:Rule  val instantTaskExecutorRule = InstantTaskExecutorRule()  @Mock  private lateinit var resultObserver: Observer<Int>  private lateinit var viewModel: CalculatorViewModel  @Before  fun setup() {  viewModel = CalculatorViewModel()  }  @Test  fun `GIVEN a and b WHEN add got invoked THEN it return add value to Observer`() {  viewModel = CalculatorViewModel()  viewModel.result.observeForever(resultObserver)  viewModel.add(Input(5, 15))  verify(resultObserver).onChanged(20)  }  @After  fun tearDown() {  viewModel.result.removeObserver(resultObserver)  }  } | </end> |
| <hitle> | UnitTestingRoomDatabase | <chare> | 1 | <pext> | 03-27/RoomTesting UnitTesting RoomDatabase  Create NoteDaoTest in androidTest/local/dao  @RunWith(AndroidJUnit4::class)  class NoteDaoTest {  private lateinit var appDatabase: AppDatabase  private lateinit var noteDao: NoteDao  @Before  fun setup() {  ------get applicationContext in Test by InstrumentationRegistry.getInstrumentation---------------  val content = InstrumentationRegistry.getInstrumentation().targetContext.applicationContext  -------Room.inMemoryDatabaseBuilder------------  appDatabase = Room.inMemoryDatabaseBuilder(content, AppDatabase::class.java).build()  noteDao = appDatabase.getNoteDao()  }  @Test  fun testInsertIntoAppDatabase() {  val note = Note(  "First note",  "Sample body",  "27th Mar 2023",  "12345",  "19",  "White",  "false",  "false",  0  )  val index: Long = noteDao.insert(note)  val saveNote = noteDao.findNoteByIndex(index)  assertNotNull(saveNote)  assertEquals(index, saveNote?.index)  }  @Test  fun getAllNotesFromAppDatabase() {  val insertList = mutableListOf<Note>()  for (i in 0..3) {  val note = Note(  "First note $i",  "Sample body $i",  "27th Mar 2023 $i",  "12345",  "19",  "White",  "false",  "false",  0  )  insertList.add(note)  val insertIndex: Long = noteDao.insert(note)  }  val notesFromDB = noteDao.getAllNotes()  assertEquals(insertList.size, notesFromDB.size)  }  @Test  fun deleteNoteFromAppDatabase() {  val note = Note(  "First note",  "Sample body",  "27th Mar 2023",  "12345",  "19",  "White",  "false",  "false",  0  )  val index: Long = noteDao.insert(note)  val saveNote = noteDao.findNoteByIndex(index)  noteDao.delete(saveNote!!)  assertNull(noteDao.findNoteByIndex(index))  }  }  -------------------------Gradle---------------------------  plugins {  id 'kotlin-kapt'  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  testImplementation 'org.mockito:mockito-inline:3.9.0'  testImplementation "org.mockito.kotlin:mockito-kotlin:4.0.0"  androidTestImplementation "androidx.test:rules:1.3.0"  testImplementation 'androidx.arch.core:core-testing:2.1.0'  // Room database  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  }  -------------------------local/entity/Note-------------------------  @Entity(tableName = "Notes")  data class Note(  @ColumnInfo(name = "title") val title: String,  @ColumnInfo(name = "body") val body: String,  @ColumnInfo(name = "date") val date: String,  @ColumnInfo(name = "passcode") val passcode: String,  @ColumnInfo(name = "bodyFontSize") val bodyFontSize: String = "14",  @ColumnInfo(name = "textColor") val textColor: String,  @ColumnInfo(name = "isStarred") val isStarred: String,  @ColumnInfo(name = "isLocked") val isLocked: String,  @PrimaryKey(autoGenerate = true) var index: Long = 0,  @ColumnInfo(name = "noteType") val noteType: String? = "",  @ColumnInfo(name = "noteType2") val noteType2: String? = "",  )  -------------------------local/dao/NoteDao-------------------------  @Dao  interface NoteDao {  @Insert  fun insert(note: Note): Long  @Delete  fun delete(note: Note)  @Update  fun update(note: Note)  @Query("Select \* from Notes")  fun getAllNotes(): MutableList<Note>  @Query("SELECT \* FROM notes WHERE `index` =:index")  fun findNoteByIndex(index: Long): Note?  }  -------------------------Room DB Migration-------------------------  val MIGRATION\_1\_2 = object : Migration(1, 2) {  override fun migrate(database: SupportSQLiteDatabase) {  database.execSQL("Alter TABLE Notes ADD COLUMN noteType TEXT")  }  }  -------------------------local/AppDatabase-------------------------  @Database(entities = [Note::class], version = 1, exportSchema = false)  abstract class AppDatabase : RoomDatabase() {  abstract fun getNoteDao(): NoteDao  companion object {  private var INSTANCE: AppDatabase? = null  fun getInstance(context: Context): AppDatabase? {  if (INSTANCE == null) {  INSTANCE = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "noteDB"  ).allowMainThreadQueries()  .build()  }  return INSTANCE  }  }  }  -------------------------ui/NotesAdapter-------------------------  class NotesAdapter(private val context: Context, private val notes: List<Note>) :  RecyclerView.Adapter<NotesAdapter.NoteViewHolder>() {  private lateinit var binding: NoteItemBinding  private lateinit var bindingPasscodeDialogBinding: PasscodeDialogBinding  override fun getItemCount() = notes.size  override fun onBindViewHolder(holder: NoteViewHolder, position: Int) {  holder.apply {  val note = notes[position]  bind(note)  itemView.setOnClickListener {  if (note.isLocked == "true") {  showLockedNoteDialog(note, position)  } else {  //make a intent to show details of note  }  }  }  }  private fun showLockedNoteDialog(note: Note, position: Int) {  val view = bindingPasscodeDialogBinding.root  val builder = AlertDialog.Builder(context).apply {  setView(view)  setTitle("Passcode required")  setPositiveButton("view") { d, \_ ->  val dialogPassCode = bindingPasscodeDialogBinding.edtPasscode.text.toString()  if (dialogPassCode != note.passcode) {  Toast.makeText(context, "Wrong passcode", Toast.LENGTH\_SHORT).show()  } else {  bindingPasscodeDialogBinding.edtPasscode.text?.clear()  d.dismiss()  }  }  setNegativeButton("Cancel") { d, \_ ->  d.dismiss()  }  }  if (view.parent != null) {  (view.parent as ViewGroup).removeView(view)  }  builder.show()  }  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NoteViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = NoteItemBinding.inflate(layoutInflater, parent, false)  bindingPasscodeDialogBinding = PasscodeDialogBinding.inflate(  layoutInflater, parent,  false  )  return NoteViewHolder(binding.root)  }  inner class NoteViewHolder(v: View) : RecyclerView.ViewHolder(v) {  fun bind(note: Note) {  binding.apply {  noteTitle.text = note.title  noteDate.text = note.date  if (note.isStarred == "true") {  starIcon.setImageResource(R.drawable.ic\_baseline\_star\_24)  }  if (note.isLocked == "true") {  lockIcon.setImageResource(R.drawable.ic\_baseline\_lock\_24)  }  }  }  }  }  -------------------------ui/MainActivity-------------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var notesAdapter: NotesAdapter  private var noteDao: NoteDao? = null  private lateinit var noteList: MutableList<Note>  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initDB()  initView()  createNewNote()  }  private fun createNewNote() {  binding.btnAddNewNote.setOnClickListener {  val note = Note(  binding.notesTitle.text.toString(),  "Sample body",  "1st Feb 2022",  "12345",  "19",  "White",  "false",  "false",  0  )  noteDao?.insert(note)  initView()  Toast.makeText(this, "new note added!!", Toast.LENGTH\_SHORT).show()  }  }  private fun initDB() {  val appDB: AppDatabase? = application.let { AppDatabase.getInstance(applicationContext) }  noteDao = appDB?.getNoteDao()  }  private fun initView() {  noteList = ArrayList()  noteList = noteDao?.getAllNotes()!!  notesAdapter = NotesAdapter(this,noteList)  if (noteList.size > 0) {  binding.notesRecyclerView.layoutManager = GridLayoutManager(this, 2)  binding.notesRecyclerView.adapter = notesAdapter  }  }  }  -------------------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".ui.MainActivity">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/notesTitle"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  app:layout\_constraintTop\_toTopOf="parent" />  <Button  android:id="@+id/btnAddNewNote"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Add new note"  app:layout\_constraintTop\_toBottomOf="@+id/notesTitle" />  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/notesRecyclerView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Hello World!"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/btnAddNewNote" />  </androidx.constraintlayout.widget.ConstraintLayout>  -------------------------note\_item.xml-------------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView 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="155dp"  android:layout\_height="180dp"  android:layout\_margin="10dp"  android:padding="10dp"  app:cardBackgroundColor="#1E88E5"  app:cardCornerRadius="10dp"  app:cardElevation="10dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@drawable/note\_card\_color\_selector">  <ImageButton  android:id="@+id/starIcon"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_marginStart="5dp"  android:layout\_marginTop="5dp"  android:background="#00FFFFFF"  android:contentDescription="This is for fav feature"  android:src="@drawable/ic\_baseline\_star\_border\_24"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <ImageButton  android:id="@+id/lockIcon"  android:layout\_width="30dp"  android:layout\_height="30dp"  android:layout\_marginStart="5dp"  android:layout\_marginTop="5dp"  android:background="#00FFFFFF"  android:contentDescription="This is for fav feature"  android:src="@drawable/ic\_baseline\_lock\_open\_24"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <TextView  android:id="@+id/noteTitle"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:fontFamily="sans-serif-condensed"  android:textColor="@color/white"  android:textSize="29sp"  android:textStyle="bold"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/lockIcon"  tools:text="Title of Note" />  <TextView  android:id="@+id/noteDate"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_margin="5dp"  android:fontFamily="sans-serif-condensed"  android:textColor="@color/white"  android:textSize="18sp"  android:textStyle="bold"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  tools:text="March, 23 2022" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView>  -------------------------passcode\_dialog.xml-------------------------  <?xml version="1.0" encoding="utf-8"?>  <androidx.cardview.widget.CardView 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"  android:layout\_margin="10dp"  android:padding="10dp"  app:cardBackgroundColor="#1E88E5"  app:cardCornerRadius="10dp"  app:cardElevation="10dp">  <androidx.constraintlayout.widget.ConstraintLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:padding="10dp">  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/edtPasscode"  android:layout\_width="250dp"  android:layout\_height="wrap\_content"  android:layout\_marginTop="25dp"  android:background="@color/white"  android:gravity="center"  android:hint="Enter passcode"  android:inputType="number"  android:maxLength="10"  android:maxLines="1"  android:padding="10dp"  android:textColor="@color/black"  android:textColorHint="#554F4F"  android:textStyle="bold"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout>  </androidx.cardview.widget.CardView> | </end> |
| <hitle> | Mockito Verify | <chare> | 1 | <pext> | Mockito Verify methods are used to check that certain behavior happened. We can use Mockito verify methods at the end of the testing method code to make sure that specified methods are called.  -------------Mockito verify() simple example-----------------  Mockito verify() method can be used to test number of method invocations too. We can test exact number of times, at least once, at least, at most number of invocation times for a mocked method.  @Test  void test() {  List<String> mockList = mock(List.class);  mockList.add("Pankaj");  mockList.size();    verify(mockList).add("Pankaj");  }  Above verify method will pass if add("Pankaj") is called only once on the mocked list object. It’s the same as calling with times(1) argument with verify method.  verify(mockList, times(1)).size();  If we want to make sure a method is called but we don’t care about the argument, then we can use ArgumentMatchers with verify method.  verify(mockList).add(anyString());  verify(mockList).add(any(String.class));  verify(mockList).add(ArgumentMatchers.any(String.class));  Note that org.mockito.Mockito class provides static methods for most of the useful methods in the Mockito framework, this helps us in writing fluent code by importing them using import static.  ------------------Mockito verify with number of times------------------  Mockito verify() method is overloaded, the second one is verify(T mock, VerificationMode mode). We can use it to verify for the invocation count.  verify(mockList, times(1)).size(); //same as normal verify method  verify(mockList, atLeastOnce()).size(); // must be called at least once  verify(mockList, atMost(2)).size(); // must be called at most 2 times  verify(mockList, atLeast(1)).size(); // must be called at least once  verify(mockList, never()).clear(); // must never be called  ------------------verifyNoMoreInteractions()------------------  This method can be used after all the verify methods to make sure that all the interactions are verified. It will fail the test if there are any unverified interactions on the mocked object.  // all interactions are verified, so below will pass  verifyNoMoreInteractions(mockList);  mockList.isEmpty();  // isEmpty() is not verified, so below will fail  verifyNoMoreInteractions(mockList);  The second invocation of verifyNoMoreInteractions() will fail with the error  ------------------verifyZeroInteractions()------------------  verifyZeroInteractions() method behavior is same as verifyNoMoreInteractions() method.  Map mockMap = mock(Map.class);  Set mockSet = mock(Set.class);  verify(mockList).isEmpty();  verifyZeroInteractions(mockList, mockMap, mockSet);  ------------------Mockito verify only method call------------------  If we want to verify that only one method is being called, then we can use only() with verify method.  Map mockMap = mock(Map.class);  mockMap.isEmpty();  verify(mockMap, only()).isEmpty();  ------------------Mockito Verify Order of Invocation------------------  We can use InOrder to verify the order of invocation. We can skip any method to verify, but the methods being verified must be invoked in the same order.  InOrder inOrder = inOrder(mockList, mockMap);  inOrder.verify(mockList).add("Pankaj");  inOrder.verify(mockList, calls(1)).size();  inOrder.verify(mockList).isEmpty();  inOrder.verify(mockMap).isEmpty(); | </end> |
| <hitle> | CoroutineRetrofitMVVMMockito, MockK UnitTesting | <chare> | 1 | <pext> | 03-27/CoroutineRetrofitMVVMMockito, CoroutineRetrofitMVVMMockK  -----------------------Gradle-----------------------  id 'kotlin-kapt'  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  testImplementation 'org.mockito:mockito-inline:3.9.0'  testImplementation "org.mockito.kotlin:mockito-kotlin:4.0.0"  androidTestImplementation "androidx.test:rules:1.3.0"  testImplementation 'androidx.arch.core:core-testing:2.1.0'  // Room database  def roomVersion = "2.4.2"  implementation("androidx.room:room-runtime:$roomVersion")  kapt("androidx.room:room-compiler:$roomVersion")  //coroutine  def coroutine\_version = "1.6.4"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:$coroutine\_version"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-core:$coroutine\_version"  //In coroutines for viewmodel scope  implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:2.2.0"  testImplementation "org.jetbrains.kotlinx:kotlinx-coroutines-test:$coroutine\_version"  // Retrofit & OkHttp  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  // Glide  implementation 'com.github.bumptech.glide:glide:4.13.2'  annotationProcessor 'com.github.bumptech.glide:compiler:4.13.2'  //lifecycle  implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.4.1'  implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.4.1'  //Mockk unit testing  testImplementation "io.mockk:mockk:1.12.0"  -----------------------model/ApiClient-----------------------  object ApiClient {  val retrofit: Retrofit by lazy {  val loggingInterceptor = HttpLoggingInterceptor()  loggingInterceptor.level = HttpLoggingInterceptor.Level.BODY  val client = OkHttpClient.Builder()  .addInterceptor(loggingInterceptor)  .build()  val retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .client(client)  .addConverterFactory(GsonConverterFactory.create())  .build()  retrofit  }  }  -----------------------model/ApiService-----------------------  interface ApiService {  @GET("random")  suspend fun getRandomDog(): Response<Dog>  companion object {  fun getInstance() = ApiClient.retrofit.create(ApiService::class.java)  }  }  -----------------------model/Constants-----------------------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  }  -----------------------model/Dog-----------------------  data class Dog(  val message: String,  val status: String  )  -----------------------model/Repository-----------------------  class Repository(private val apiService: ApiService) {  suspend fun getRandomDog() = apiService.getRandomDog()  }  ----------------------view/MainActivity-----------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var dogViewModel: DogViewModel  private lateinit var dogViewModelFactory: DogViewModelFactory  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViewModel()  setUpViews()  setUpObserver()  }  private fun initViewModel() {  dogViewModelFactory = DogViewModelFactory(Repository(ApiService.getInstance()))  dogViewModel = ViewModelProvider(this, dogViewModelFactory)[DogViewModel::class.java]  }  private fun setUpObserver() {  dogViewModel.dogResponse.observe(this) {  Glide.with(this)  .load(it.message)  .into(binding.imageOfDog)  }  dogViewModel.progress.observe(this) {  if (it) {  binding.loadingSpinner.visibility = View.VISIBLE  } else {  binding.loadingSpinner.visibility = View.GONE  }  }  dogViewModel.error.observe(this) {  Toast.makeText(this, it, Toast.LENGTH\_SHORT).show()  }  }  private fun setUpViews() {  binding.btnSearch.setOnClickListener {  dogViewModel.getDog()  }  }  }  -----------------------viewmodel/DogViewModel-----------------------  class DogViewModel(private val repository: Repository) : ViewModel() {  val dogResponse = MutableLiveData<Dog>()  val error = MutableLiveData<String>()  val progress = MutableLiveData<Boolean>()  fun getDog() {  viewModelScope.launch {  try {  progress.postValue(true)  val response = repository.getRandomDog()  if (response.isSuccessful) {  response.body()?.let {  dogResponse.postValue(it)  progress.postValue(false)  }  } else {  progress.postValue(false)  error.postValue(ERROR\_MESSAGE)  }  } catch (exception: Exception) {  progress.postValue(false)  error.postValue(exception.message)  Log.i("tag", exception.toString())  }  return@launch  }  }  companion object {  const val ERROR\_MESSAGE = "Internal Server error"  }  }  -----------------------viewmodel/DogViewModelFactory-----------------------  class DogViewModelFactory constructor(private val repository: Repository) :  ViewModelProvider.Factory {  override fun <T : ViewModel> create(modelClass: Class<T>): T {  return if (modelClass.isAssignableFrom(DogViewModel::class.java)) {  DogViewModel(this.repository) as T  } else {  throw IllegalArgumentException("View model not found")  }  }  }  -------------start UnitTesting------------  Create test code in test folder  -----------------------Test/model/Constants-----------------------  object Constants {  const val SUCCESS\_RESULT\_WITH\_DATA = """{"message":"https://images.dog.ceo/breeds/shihtzu/n02086240\_1770.jpg","status":"success"}"""  const val SUCCESS\_RESULT\_WITHOUT\_DATA = """{"message":"","status":"success"}"""  const val FAILURE\_RESULT = """{"message":"","status":"failure"}"""  }  -----------------------Test/utis/MainDispatcherRule-----------------------  class MainDispatcherRule(  val testDispatcher: TestDispatcher = UnconfinedTestDispatcher(),  ) : TestWatcher() {  override fun starting(description: Description) {  Dispatchers.setMain(testDispatcher)  }  override fun finished(description: Description) {  Dispatchers.resetMain()  }  }  -----------------------Test/viewmodel/DogViewModelTest(Mockito)-----------------------  @ExperimentalCoroutinesApi  @RunWith(MockitoJUnitRunner::class)  class DogViewModelTest {  @Mock  private lateinit var repository: Repository  @Mock  private lateinit var successObserver: Observer<Dog>  @Mock  private lateinit var failureObserver: Observer<String>  @Mock  private lateinit var progressObserver: Observer<Boolean>  @get:Rule  val instantTaskExecutorRule = InstantTaskExecutorRule()  @get:Rule  val mainDispatcherRule = MainDispatcherRule()  private lateinit var viewModel: DogViewModel  @Before  fun setup() {  viewModel = DogViewModel(repository)  }  @Test  fun `GIVEN success Stub or Mock data WHEN getRandomDog invoked THEN received success result`() {  runTest {  val responseFromAPI = Response.success(  Gson().fromJson(  Constants.SUCCESS\_RESULT\_WITH\_DATA,  Dog::class.java  )  )  doReturn(responseFromAPI).`when`(repository).getRandomDog()  val viewModel = DogViewModel(repository)  viewModel.apply {  progress.observeForever(progressObserver)  dogResponse.observeForever(successObserver)  getDog()  verify(progressObserver).onChanged(true)  verify(repository).getRandomDog()  verify(progressObserver).onChanged(false)  val expectedResult = Gson().fromJson(Constants.SUCCESS\_RESULT\_WITH\_DATA, Dog::class.java)  verify(successObserver).onChanged(expectedResult)  }  }  }  @Test  fun `GIVEN success Stub or Mock data WHEN getRandomDog invoked THEN received success with no data result`() {  runTest {  val responseFromAPI = Response.success(  Gson().fromJson(  Constants.SUCCESS\_RESULT\_WITHOUT\_DATA,  Dog::class.java  )  )  doReturn(responseFromAPI).`when`(repository).getRandomDog()  viewModel.apply {  progress.observeForever(progressObserver)  dogResponse.observeForever(successObserver)  getDog()  verify(progressObserver).onChanged(true)  verify(repository).getRandomDog()  verify(progressObserver).onChanged(false)  val expectedResult =  Gson().fromJson(Constants.SUCCESS\_RESULT\_WITHOUT\_DATA, Dog::class.java)  verify(successObserver).onChanged(expectedResult)  }  }  }  @Test  fun `GIVEN failure Stub or Mock data WHEN getRandomDog invoked THEN received failure result`() {  runTest {  val responseFromAPI = Response.error<String>(  500,  ResponseBody.create(  "text/plain".toMediaTypeOrNull(),  TEST\_ERROR\_MESSAGE  )  )  doReturn(responseFromAPI).`when`(repository).getRandomDog()  val viewModel = DogViewModel(repository)  viewModel.apply {  progress.observeForever(progressObserver)  error.observeForever(failureObserver)  getDog()  verify(progressObserver).onChanged(true)  verify(repository).getRandomDog()  verify(progressObserver).onChanged(false)  verify(failureObserver).onChanged(TEST\_ERROR\_MESSAGE)  }  }  }  @Test(expected = RuntimeException::class)  fun `GIVEN no internet WHEN getRandomDog invoked THEN received exception as result`() {  runTest {  val exception = RuntimeException()  doThrow(exception).`when`(repository.getRandomDog())  viewModel.apply {  progress.observeForever(progressObserver)  error.observeForever(failureObserver)  getDog()  verify(progressObserver).onChanged(true)  verify(repository).getRandomDog()  verify(progressObserver).onChanged(false)  verify(failureObserver).onChanged(TEST\_EXCEPTION)  }  }  }  @After  fun tearDown() {  viewModel.apply {  if(dogResponse.hasActiveObservers()) {  dogResponse.removeObserver(successObserver)  }  if(error.hasActiveObservers()) {  error.removeObserver(failureObserver)  }  progress.removeObserver(progressObserver)  }  }  private companion object {  const val TEST\_ERROR\_MESSAGE = "Internal Server error"  const val TEST\_EXCEPTION = "NO INTERNET AVAILABLE"  }  }  --------------------Test/viewmodel/DogViewModelTest(MockK)------------------------------  //@ExperimentalCoroutinesApi  class DogViewModelTest {  @get:Rule  val instantTaskExecutorRule = InstantTaskExecutorRule()  @get:Rule  val mainDispatcherRule = MainDispatcherRule()  private val repository = mockk<Repository>()  private val successObserver = mockk<Observer<Dog>>(relaxUnitFun = true)  private val failureObserver = mockk<Observer<String>>(relaxed = true)  private val progressObserver = mockk<Observer<Boolean>>(relaxed = true)  private lateinit var viewModel: DogViewModel  @Before  fun setUp() {  viewModel = DogViewModel(repository)  }  @Test  fun `GIVEN success Stub or Mock data WHEN getRandomDog invoked THEN received success result`() {  // runTest {  val responseFromAPI = Response.success(  Gson().fromJson(  Constants.SUCCESS\_RESULT\_WITH\_DATA,  Dog::class.java  )  )  coEvery { repository.getRandomDog() } returns responseFromAPI  viewModel.apply {  progress.observeForever(progressObserver)  dogResponse.observeForever(successObserver)  getDog()  coVerify {  progressObserver.onChanged(true)  repository.getRandomDog()  progressObserver.onChanged(false)  }  val expectedResult =  Gson().fromJson(Constants.SUCCESS\_RESULT\_WITH\_DATA, Dog::class.java)  coVerify { successObserver.onChanged(expectedResult) }  }  // }  }  @Test  fun `GIVEN success Stub or Mock data WHEN getRandomDog invoked THEN received success with no data result`() {  // runTest {  val responseFromAPI = Response.success(  Gson().fromJson(  Constants.SUCCESS\_RESULT\_WITHOUT\_DATA,  Dog::class.java  )  )  coEvery { repository.getRandomDog() } returns responseFromAPI  viewModel.apply {  progress.observeForever(progressObserver)  dogResponse.observeForever(successObserver)  getDog()  coVerify {  progressObserver.onChanged(true)  repository.getRandomDog()  progressObserver.onChanged(false)  }  val expectedResult =  Gson().fromJson(Constants.SUCCESS\_RESULT\_WITHOUT\_DATA, Dog::class.java)  coVerify { successObserver.onChanged(expectedResult) }  }  // }  }  @Test  fun `GIVEN failure Stub or Mock data WHEN getRandomDog invoked THEN received failure result`() {  // runTest {  Response.error<String>(  500,  TEST\_ERROR\_MESSAGE.toResponseBody("text/plain".toMediaType())  )  coEvery { repository.getRandomDog().isSuccessful } returns false  viewModel.apply {  progress.observeForever(progressObserver)  error.observeForever(failureObserver)  getDog()  coVerify {  progressObserver.onChanged(true)  repository.getRandomDog()  progressObserver.onChanged(false)  failureObserver.onChanged(TEST\_ERROR\_MESSAGE)  }  }  // }  }  @Test(expected = AssertionError::class)  fun `GIVEN no internet WHEN getRandomDog invoked THEN received exception as result`() {  // runTest {  coEvery { repository.getRandomDog() } throws AssertionError(TEST\_EXCEPTION)  viewModel.apply {  progress.observeForever(progressObserver)  error.observeForever(failureObserver)  getDog()  coVerify {  progressObserver.onChanged(true)  repository.getRandomDog()  progressObserver.onChanged(false)  failureObserver.onChanged(TEST\_EXCEPTION)  }  }  // }  }  @After  fun tearDown() {  viewModel.apply {  if (dogResponse.hasActiveObservers()) {  dogResponse.removeObserver(successObserver)  }  if (error.hasActiveObservers()) {  error.removeObserver(failureObserver)  }  progress.removeObserver(progressObserver)  }  }  private companion object {  const val TEST\_ERROR\_MESSAGE = "Internal Server error"  const val TEST\_EXCEPTION = "NO INTERNET AVAILABLE"  }  } | </end> |
| <hitle> | EspressoDemo1 ActivityUIUnitTesting | <chare> | 1 | <pext> | 03-28/EspressoDemo1 ActivityUIUnitTesting  ------------------------Gradle-------------------  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  //required for scena  testImplementation 'androidx.test.ext:junit-ktx:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  androidTestImplementation('androidx.test.espresso:espresso-core:3.5.1')  androidTestImplementation('androidx.test:runner:1.5.2')  androidTestImplementation('androidx.test:rules:1.5.0')  }  ------------------------MainActivity------------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setName()  }  private fun setName() {  binding.btnClick.setOnClickListener {  binding.txtDisplay.text = binding.edtInput.text.toString()  }  }  }  ------------------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".MainActivity">  <TextView  android:id="@+id/txtDisplay"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="40sp"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/edtInput"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="enter name"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.appcompat.widget.AppCompatButton  android:background="#1565C0"  android:textColor="@color/white"  android:id="@+id/btnClick"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Set your Name"  app:layout\_constraintBottom\_toBottomOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout>  ------------------------androidTest/MainActivityTest------------------------  Right click on MainActivity and Generate/Test…/choose androidTest folder  @RunWith(AndroidJUnit4::class)  class MainActivityTest {  @get:Rule  var activityScenarioRule = ActivityScenarioRule(MainActivity::class.java)  @Test  fun givenNameToSetWhenButtonClickedThenSetOnTextView() {  onView(withId(R.id.edtInput)).perform(typeText(TEST\_NAME), closeSoftKeyboard())  onView(withId(R.id.btnClick)).perform(click())  onView(withId(R.id.txtDisplay)).check(matches(withText(TEST\_NAME)))  }  companion object {  const val TEST\_NAME = "Thomas Jones"  }  /\* @Before  fun setUp() {  }  @After  fun tearDown() {  }\*/  } | </end> |
| <hitle> | EspressoDemo2 IntentUIUnitTesting | <chare> | 1 | <pext> | 03-28/EspressoDemo2 IntentUIUnitTesting  ---------------------Gradle---------------------  //required for scenario  testImplementation 'androidx.test.ext:junit-ktx:1.1.5'  androidTestImplementation('androidx.test.espresso:espresso-core:3.5.1')  androidTestImplementation('androidx.test:runner:1.5.2')  androidTestImplementation('androidx.test:rules:1.5.0')  //for intents testing using espresso  androidTestImplementation 'androidx.test.espresso:espresso-intents:3.5.1'  ---------------------model/Quote---------------------  data class Quote(val quote: String, val author: String)  ---------------------viewmodel/QuotesViewModel---------------------  class QuotesViewModel :ViewModel(){  var quotes = MutableLiveData<Quote>()  private lateinit var listOfQuotes: ArrayList<Quote>  fun fetchQuotes() {  quotes.postValue(dataSet())  }  fun dataSet(): Quote {  listOfQuotes = ArrayList()  listOfQuotes.apply {  add(Quote("ABC", "BCD"))  add(Quote("ABC", "Hello"))  add(Quote("Hiii", "BCD"))  add(Quote("Hiii", "Hello"))  add(Quote("Hiii", "BCD"))  add(Quote("Hiii", "Hello"))  add(Quote("ABC", "BCD"))  add(Quote("Hiii", "Hello"))  add(Quote("ABC", "BCD"))  add(Quote("Hiii", "Hello"))  }  val randomIndex = Random.nextInt(listOfQuotes.size)  return listOfQuotes[randomIndex]  }  }  ---------------------MainActivity---------------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var viewModel: QuotesViewModel  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpViewModel()  setNextQuote()  }  private fun setUpViewModel() {  viewModel = ViewModelProvider(this)[QuotesViewModel::class.java]  }  private fun setNextQuote() {  binding.apply {  btnNextQuote.setOnClickListener {  viewModel.fetchQuotes()  }  viewModel.quotes.observe(this@MainActivity) {  txtQuote.text = it.quote  txtAuthor.text = it.author  }  shareQuote.setOnClickListener {  val intent = Intent(Intent.ACTION\_SEND)  intent.type = "text/plain"  intent.putExtra("data",txtQuote.text.toString())  startActivity(intent)  }  }  }  }  ---------------------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"  android:orientation="vertical"  android:padding="15dp">  <com.google.android.material.floatingactionbutton.FloatingActionButton  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:layout\_margin="10dp"  android:layout\_marginEnd="20dp"  android:backgroundTint="@color/white"  android:clickable="true"  android:contentDescription="@string/app\_name"  android:elevation="20dp"  android:focusable="true"  android:src="@drawable/baseline\_share\_24"  android:tint="@color/white"  app:layout\_anchor="@id/card\_view"  app:layout\_anchorGravity="center"  android:id="@+id/shareQuote"  app:layout\_constraintBottom\_toBottomOf="@+id/card\_view"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/card\_view" />  <androidx.cardview.widget.CardView  android:id="@+id/card\_view"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:padding="10dp"  app:cardBackgroundColor="#1D2B3A"  app:cardCornerRadius="20dp"  app:cardElevation="5dp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical">  <TextView  android:id="@+id/txtQuote"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:fontFamily="cursive"  android:text="@string/my\_quote"  android:textColor="@color/white"  android:textSize="45sp"  android:textStyle="italic" />  <TextView  android:id="@+id/txtAuthor"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:fontFamily="monospace"  android:gravity="bottom|start"  android:text="@string/my\_quote\_author"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="italic" />  </LinearLayout>  </androidx.cardview.widget.CardView>  <androidx.appcompat.widget.AppCompatImageButton  android:clickable="true"  android:focusable="true"  android:id="@+id/btnNextQuote"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:backgroundTint="@color/white"  android:src="@drawable/baseline\_arrow\_circle\_right\_24"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout>  ---------------------androidTest/MainActivityTest---------------------  @RunWith(AndroidJUnit4::class)  class MainActivityTest {  @get:Rule  var activityScenarioRule = ActivityScenarioRule(MainActivity::class.java)  @Test  fun givenNameToSetWhenButtonClickedThenSetOnTextView() {  onView(withId(R.id.btnNextQuote)).perform(click())  onView(withId(R.id.btnNextQuote)).perform(click())  onView(withId(R.id.btnNextQuote)).perform(click())  onView(withId(R.id.txtQuote)).check(matches(withText(TEST\_QUOTE)))  }  @Test  fun whenShareQuoteButtonInvokedThenIntentToChooserShouldBeShown() {  Intents.init()  val expected = allOf(hasAction(Intent.ACTION\_SEND))  onView(withId(R.id.shareQuote)).perform(click())  intended(expected)  Intents.release()  }  companion object {  const val TEST\_QUOTE = "ABC"  }  } | </end> |
| <hitle> | EspressoDemo3 RecyclerView UIUnitTesting | <chare> | 1 | <pext> | 03-28/EspressoDemo3 RecyclerView UIUnitTesting  ---------------------Gradle----------------  //required for scenario  testImplementation 'androidx.test.ext:junit-ktx:1.1.5'  androidTestImplementation('androidx.test.espresso:espresso-core:3.5.1')  androidTestImplementation('androidx.test:runner:1.5.2')  androidTestImplementation('androidx.test:rules:1.5.0')  //for intents testing using espresso  androidTestImplementation 'androidx.test.espresso:espresso-intents:3.5.1'  //for recyclerview or may be more view components  androidTestImplementation 'androidx.test.espresso:espresso-contrib:3.5.1'  ---------------------CustomAdapter---------------------  internal class CustomAdapter constructor(  private val mDataSet: List<String>,  private val mContext: Context  ) :  RecyclerView.Adapter<CustomAdapter.ViewHolder>() {  internal class ViewHolder(v: View) : RecyclerView.ViewHolder(v) {  val textView: TextView  // We'll use this field to showcase matching the holder from the test.  var isInTheMiddle = false  init {  textView = v.findViewById<View>(R.id.textView) as TextView  }  }  override fun onCreateViewHolder(viewGroup: ViewGroup, viewType: Int): ViewHolder {  // Create a new view.  val v: View = LayoutInflater.from(viewGroup.context)  .inflate(R.layout.text\_row\_item, viewGroup, false)  return ViewHolder(v)  }  override fun onBindViewHolder(viewHolder: ViewHolder, position: Int) {  if (position == mDataSet.size / 2 /\* calculate middle element position \*/) {  viewHolder.isInTheMiddle = true  viewHolder.textView.text = mContext.resources.getString(R.string.middle)  } else {  viewHolder.isInTheMiddle = false  viewHolder.textView.text = mDataSet[position]  }  }  // Return the size of your data set (invoked by the layout manager)  override fun getItemCount(): Int {  return mDataSet.size  }  }  ---------------------MainActivity---------------------  class MainActivity : AppCompatActivity() {  private val DATASET\_COUNT = 50  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)  // Create a RecyclerView, a LayoutManager, a data Adapter and wire everything up.  val recyclerView = findViewById<RecyclerView>(R.id.recyclerView)  val layoutManager = LinearLayoutManager(applicationContext)  recyclerView.layoutManager = layoutManager  val dataSet: MutableList<String> = ArrayList(DATASET\_COUNT)  for (i in 0 until DATASET\_COUNT) {  dataSet.add(getString(R.string.item\_element\_text) + i)  }  val adapter = CustomAdapter(dataSet, applicationContext)  recyclerView.adapter = adapter  }  }  ---------------------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="10dp"  tools:context=".MainActivity">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recyclerView"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent" />  </LinearLayout>  ---------------------text\_row\_item.xml---------------------  <?xml version="1.0" encoding="utf-8"?>  <FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="@dimen/list\_item\_height"  android:layout\_marginLeft="@dimen/margin\_medium"  android:layout\_marginRight="@dimen/margin\_medium"  android:gravity="center\_vertical">  <TextView  android:id="@+id/textView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/element\_text"/>  </FrameLayout>  ---------------------androidTest/MainActivityTest---------------------  @RunWith(AndroidJUnit4::class)  class MainActivityTest {  @get:Rule  var activityScenarioRule = ActivityScenarioRule(MainActivity::class.java)  @Test  fun whenScrollTheRecyclerViewThenItemWithTextShouldExist() {  onView(withId(R.id.recyclerView)).perform(  RecyclerViewActions.scrollTo<RecyclerView.ViewHolder>(  ViewMatchers.hasDescendant(ViewMatchers.withText("$ITEM\_IN\_LIST$PARTICULAR\_POSITION"))  )  )  }  @Test  fun whenScrollTheRecyclerViewToParticularPositionCheckItsText() {  onView(withId(R.id.recyclerView)).perform(  RecyclerViewActions.actionOnItemAtPosition<RecyclerView.ViewHolder>(  PARTICULAR\_POSITION, ViewActions.click()  )  )  val itemAtElementText = "$ITEM\_IN\_LIST$PARTICULAR\_POSITION"  onView(withText(itemAtElementText)).check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  }  @Test  fun whenScrollTheRecyclerViewToParticularPositionPosition() {  onView(withId(R.id.recyclerView)).perform(  RecyclerViewActions.actionOnItemAtPosition<RecyclerView.ViewHolder>(  PARTICULAR\_POSITION, ViewActions.click()  )  )  }  private companion object {  const val NOT\_IN\_LIST = "not in the list"  const val ITEM\_IN\_LIST = "This is element #"  const val PARTICULAR\_POSITION = 30  }  } | </end> |
| <hitle> | CleanArchitectureDemo StateFlow UnitTestingDogViewModel | <chare> | 1 | <pext> | 03-29/ CleanArchitectureDemo StateFlow UnitTestingDogViewModel  \*\*\*Source Tree\*\*\*  Data/entity/Dog.kt  Data/local/(RoomDB)  Data/network/ApiService.kt  Data/network/Constants.kt  Data/network/Repository.kt  Data/network/RepositoryImplementation.kt  Di/AppRepoModule.kt  Di/NetworkModule.kt  Domain/  Presentation/activity/MainActivity.kt  Presentation/util/AppExtensions.kt  Presentation/UIState.kt  Viewmodel/DogViewModel  App.kt  --------Gradle.Module--------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'kotlin-kapt'  id 'dagger.hilt.android.plugin'  }  android {  namespace 'com.example.cleanarchitecturedemo'  compileSdk 33  defaultConfig {  applicationId "com.example.cleanarchitecturedemo"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  //testInstrumentationRunner = "com.example.android.dagger.CustomTestRunner"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures{  viewBinding true  dataBinding true  }  }  kapt {  correctErrorTypes true  }  hilt {  enableTransformForLocalTests = true  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  //Retrofit--------------------------------------------------------------------------------------  def retrofitVersion = "2.9.0"  implementation "com.squareup.retrofit2:retrofit:$retrofitVersion"  implementation "com.squareup.retrofit2:converter-gson:$retrofitVersion"  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  //Coroutine-------------------------------------------------------------------------------------  def coroutine\_version = "1.6.4"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:$coroutine\_version"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-core:$coroutine\_version"  testImplementation "org.jetbrains.kotlinx:kotlinx-coroutines-test:$coroutine\_version"  //Lifecycle-------------------------------------------------------------------------------------  def lifecycle\_version = "2.6.1"  implementation("androidx.lifecycle:lifecycle-viewmodel-ktx:$lifecycle\_version")  implementation("androidx.lifecycle:lifecycle-livedata-ktx:$lifecycle\_version")  implementation("androidx.lifecycle:lifecycle-runtime-ktx:$lifecycle\_version")  implementation("androidx.lifecycle:lifecycle-viewmodel-savedstate:$lifecycle\_version")  implementation("androidx.lifecycle:lifecycle-common-java8:$lifecycle\_version")  //Glide-----------------------------------------------------------------------------------------  def glide\_version = "4.14.2"  implementation "com.github.bumptech.glide:glide:$glide\_version"  annotationProcessor "com.github.bumptech.glide:compiler:$glide\_version"  //Dagger - Hilt---------------------------------------------------------------------------------  implementation 'com.google.dagger:hilt-android:2.45'  kapt 'com.google.dagger:hilt-compiler:2.45'  // For local unit tests  testImplementation 'com.google.dagger:hilt-android-testing:2.45'  kaptTest 'com.google.dagger:hilt-compiler:2.45'  /\* // For local unit tests  testImplementation 'com.google.dagger:hilt-android-testing:2.45'  testAnnotationProcessor 'com.google.dagger:hilt-compiler:2.45'\*/  // for injection view model  implementation 'androidx.activity:activity-ktx:1.7.0'  //Unit Testing ---------------------------------------------------------------------------------  testImplementation 'androidx.arch.core:core-testing:2.2.0'  testImplementation 'org.mockito:mockito-inline:3.9.0'  testImplementation 'org.mockito.kotlin:mockito-kotlin:4.0.0'  //Mockk unit testing  testImplementation "io.mockk:mockk:1.12.0"  //required for scenario  testImplementation 'androidx.test.ext:junit-ktx:1.1.5'  //Espresso  androidTestImplementation('androidx.test.espresso:espresso-core:3.5.1')  androidTestImplementation('androidx.test:runner:1.5.2')  androidTestImplementation('androidx.test:rules:1.5.0')  //for intents testing using espresso  androidTestImplementation 'androidx.test.espresso:espresso-intents:3.5.1'  testImplementation "org.robolectric:robolectric:4.9"  }  -------Gradle.project---------  // Top-level build file where you can add configuration options common to all sub-projects/modules.  buildscript {  repositories {  google()  mavenCentral()  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.45'  }  }  plugins {  id 'com.android.application' version '7.4.1' apply false  id 'com.android.library' version '7.4.1' apply false  id 'org.jetbrains.kotlin.android' version '1.8.0' apply false  }  --------AndroidManifest----------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET"/>  <application  android:name=".App"  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.CleanArchitectureDemo"  tools:targetApi="31">  <activity  android:name=".presentation.activity.MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  --------App---------  @HiltAndroidApp  open class App : Application() {  }  ---------data/entity/Dog---------  data class Dog(  var message: String,  var status: String  )  ---------data/network/ApiService---------  interface ApiService {  @GET("random")  suspend fun getRandomDog(): Dog  }  ---------data/network/Constants---------  object Constants {  const val BASE\_URL = "https://dog.ceo/api/breeds/image/"  }  ---------data/network/Repository---------  interface Repository {  suspend fun getRandomDog(): Flow<Dog>  }  ---------data/network/RepositoryImplementation---------  class RepositoryImplementation @Inject constructor(private val apiService: ApiService) :  Repository {  override suspend fun getRandomDog() = flow {  emit(apiService.getRandomDog())  }  }  --------di/AppRepoModule---------  @InstallIn(SingletonComponent::class)  @Module  class RepositoryModule {  @Provides  fun provideRepository(apiService: ApiService): Repository {  return RepositoryImplementation(apiService)  }  }  @InstallIn(SingletonComponent::class)  @Module  class ApiServiceModule {  @Provides  fun provideApiService(retrofit: Retrofit): ApiService {  return retrofit.create(ApiService::class.java)  }  }  ---------di/NetworkModule---------  @InstallIn(SingletonComponent::class)  @Module  object NetworkModule {  @Singleton  @Provides  fun provideLoggingInterceptor(): HttpLoggingInterceptor {  return HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)  }  @Singleton  @Provides  fun provideConvertorFactory(): Converter.Factory{  return GsonConverterFactory.create()  }  @Singleton  @Provides  fun provideOkhttpClient(httpLoggingInterceptor: HttpLoggingInterceptor): OkHttpClient {  val okHttpClient = OkHttpClient.Builder()  .addInterceptor(httpLoggingInterceptor)  return okHttpClient.build()  }  @Singleton  @Provides  fun provideRetrofit(  converterFactory: Converter.Factory,  okHttpClient: OkHttpClient  ): Retrofit {  val retrofit = Retrofit.Builder()  .baseUrl(BASE\_URL)  .client(okHttpClient)  .addConverterFactory(converterFactory)  return retrofit.build()  }  }  ---------presentation/activity/MainActivity---------  @AndroidEntryPoint  class MainActivity : AppCompatActivity() {  private val viewModel: DogViewModel? by viewModels()  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  observeFlowData()  setUpAPICall()  }  private fun setUpAPICall() {  binding.btnFetch.setOnClickListener {  viewModel?.fetchRandomDog()  }  }  private fun observeFlowData() {  lifecycleScope.launch {  repeatOnLifecycle(Lifecycle.State.STARTED) {  viewModel?.dogResult?.collect { uiState ->  updateUi(uiState)  }  }  }  }  private fun updateUi(uiState: UIState<Dog>) {  when (uiState) {  is UIState.Loading -> {  showProgress()  }  is UIState.Success -> {  hideProgress()  uiState.data?.let { setDataOnUI(it) } ?: showError(null)  }  is UIState.Error -> {  hideProgress()  showError(uiState.error)  }  }  }  private fun setDataOnUI(it: Dog) {  with(binding) {  Glide.with(this@MainActivity)  .load(it.message)  .error(android.R.drawable.ic\_dialog\_alert)  .placeholder(R.drawable.ic\_launcher\_background)  .into(imageOfDog)  }  }  private fun showError(message: String?) = binding.container.showSnackBar(message)  private fun showProgress() = binding.loadingSpinner.visible()  private fun hideProgress() = binding.loadingSpinner.gone()  }  ---------presentation/util/AppExtensions---------  fun View.visible(): View {  if (visibility != View.VISIBLE) {  visibility = View.VISIBLE  }  return this  }  fun View.gone(): View {  if (visibility != View.GONE) {  visibility = View.GONE  }  return this  }  fun View.showSnackBar(message: String? = this.resources.getString(R.string.default\_error)) :Unit{  Snackbar.make(  this,  message ?: this.resources.getString(R.string.default\_error),  Snackbar.LENGTH\_SHORT  ).show()  }  fun View.showToast(context: Context, message: String) {  Toast.makeText(context.applicationContext, message, Toast.LENGTH\_SHORT).show()  }  ---------presentation/util/UIState---------  sealed class UIState<T>(val data: T? = null, val error: String? = null) {  class Success<T>(data: T?) : UIState<T>(data)  class Error<T>(message: String?) : UIState<T>(error = message)  class Loading<T> : UIState<T>()  }  ---------viewmodel/DogViewModel---------  @ExperimentalCoroutinesApi  @HiltViewModel  class DogViewModel @Inject constructor(private val repository: Repository) : ViewModel() {  private val \_dogResult = MutableStateFlow<UIState<Dog>>(UIState.Loading())  val dogResult: StateFlow<UIState<Dog>> = \_dogResult  fun fetchRandomDog() {  viewModelScope.launch {  repository.getRandomDog()  .catch { exception ->  \_dogResult.value = UIState.Error(exception.message)  }.collect { result ->  \_dogResult.value = UIState.Success(result)  }  }  }  }  ---------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:id="@+id/container"  tools:context=".presentation.activity.MainActivity">  <androidx.appcompat.widget.AppCompatImageView  android:id="@+id/imageOfDog"  android:layout\_width="match\_parent"  android:layout\_height="0dp"  android:layout\_margin="10dp"  android:scaleType="centerCrop"  app:layout\_constraintBottom\_toTopOf="@+id/btnFetch"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <com.google.android.material.progressindicator.CircularProgressIndicator  android:id="@+id/loading\_spinner"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="32dp"  android:indeterminate="true"  android:visibility="gone"  app:indicatorColor="#FF0000"  app:indicatorDirectionCircular="counterclockwise"  app:indicatorSize="200dp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:trackColor="@color/purple\_500"  app:trackThickness="20dp" />  <androidx.appcompat.widget.AppCompatButton  android:id="@+id/btnFetch"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:background="#1565C0"  android:text="@string/search"  android:textAllCaps="false"  android:textColor="@color/white"  android:textStyle="bold"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout>  ---------create testing files in (test)-------------  Use @TestInstallIn instead of @InstallIn  Use FakeApiService instead of ApiService  Need FakeApiServiceModule to provide dependency  Provide sample json for fake response  \*\*\*folder structure\*\*\*  Api/FakeApiService.kt  Hilt/FakeApiServiceModule.kt  Util/JsonProvider.kt  Viewmodel/DogViewModelTest  -------------(test)/api/FakeApiService-------------------  class FakeAPiService @Inject constructor() : ApiService {  var failDogApi: Boolean = false  var wrongResponse: Boolean = false  override suspend fun getRandomDog(): Dog {  //Handle Exception  if (failDogApi) throw Exception("Api failed")  //Handle Success  val fakeResponse: Dog = JsonProvider.objectFromJsonFileWithType(DOG\_JSON)  //Handle Error  if (wrongResponse) return fakeResponse.apply {  this.message = ""  this.status = "failed"  }  return fakeResponse  }  companion object {  private const val DOG\_JSON = "/dog.json"  }  }  ----------(test)/hilt/FakeApiServiceModule-----------  @Module  @TestInstallIn(components = [SingletonComponent::class], replaces = [ApiServiceModule::class])  abstract class FakeApiServiceModule {  @Singleton  @Binds  abstract fun provideApiService(fakeAPiService: FakeAPiService): ApiService  }  ----------(test)/util/JsonProvider-----------  object JsonProvider {  inline fun <reified U> objectFromJsonFileWithType(filePath: String): U =  Gson().fromJson(fileAsString(filePath), object : TypeToken<U>() {}.type)  // fun fileAsString(filePath: String): String = this::class.java  // .getResourceAsStream(filePath)!!  // .bufferedReader()  // .use { it.readText() }  fun fileAsString(filePath: String): String {  var content = "no"  this::class.java  .getResourceAsStream(filePath)!!  .bufferedReader()  .use {  content = it.readText()  }  return content  }  }  ----------(test)/viewmodel/DogViewModelTest----------  @HiltAndroidTest  @RunWith(RobolectricTestRunner::class)  @Config(sdk = [25], application = HiltTestApplication::class)  @ExperimentalCoroutinesApi  @LooperMode(LooperMode.Mode.PAUSED)  class DogViewModelTest {  @get:Rule  val hiltRule = HiltAndroidRule(this)  @Inject  lateinit var repository: Repository  @BindValue  @JvmField  val fakeAPiService: FakeAPiService = FakeAPiService()  @Mock  private lateinit var dogViewModel: DogViewModel  @Before  fun setUp() {  hiltRule.inject()  dogViewModel = DogViewModel(repository)  }  @Test  fun `WHEN fetchRandomDog invoked THEN return success result`() = runTest {  dogViewModel.fetchRandomDog()  Shadows.shadowOf(Looper.getMainLooper()).idle()  val value = dogViewModel.dogResult.value  assertTrue(value is UIState.Success)  assertNotNull(value.data)  assertEquals("success", value.data?.status)  assertEquals(  "https://images.dog.ceo/breeds/elkhound-norwegian/n02091467\_6295.jpg",  value.data?.message  )  }  @Test  fun `WHEN fetchRandomDog invoked THEN return failure result`() = runTest {  fakeAPiService.failDogApi = true  dogViewModel.fetchRandomDog()  Shadows.shadowOf(Looper.getMainLooper()).idle()  val value = dogViewModel.dogResult.value  assertTrue(value is UIState.Error)  assertNull(value.data)  }  fun `WHEN fetchRandomDog invoked THEN return exception result`() {  }  } | </end> |
| <hitle> | PDFDownloaderWorkManager | <chare> | 1 | <pext> | 03-30/PDFDownloaderWorkManager  ---------Gradle.Module--------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  }  android {  namespace 'com.example.pdfdownloaderworkmanager'  compileSdk 33  defaultConfig {  applicationId "com.example.pdfdownloaderworkmanager"  minSdk 26  targetSdk 32  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  vectorDrawables {  useSupportLibrary true  }  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures {  compose true  }  composeOptions {  kotlinCompilerExtensionVersion '1.2.0'  }  packagingOptions {  resources {  excludes += '/META-INF/{AL2.0,LGPL2.1}'  }  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'  implementation 'androidx.activity:activity-compose:1.3.1'  implementation "androidx.compose.ui:ui:$compose\_version"  implementation "androidx.compose.ui:ui-tooling-preview:$compose\_version"  implementation 'androidx.compose.material3:material3:1.0.0-alpha11'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  androidTestImplementation "androidx.compose.ui:ui-test-junit4:$compose\_version"  debugImplementation "androidx.compose.ui:ui-tooling:$compose\_version"  debugImplementation "androidx.compose.ui:ui-test-manifest:$compose\_version"  // Retrofit  implementation 'com.squareup.retrofit2:retrofit:2.9.0'  implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  implementation "com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2"  // Work Manager  implementation "androidx.work:work-runtime-ktx:2.7.1"  implementation 'io.github.grizzi91:bouquet:1.0.1'  // Livedata  implementation 'androidx.lifecycle:lifecycle-livedata-ktx:2.4.1'  implementation 'androidx.compose.runtime:runtime-livedata:1.2.0-alpha05'  }  -----------Gradle/Project-------------  buildscript {  ext {  compose\_version = '1.2.0'  }  }// Top-level build file where you can add configuration options common to all sub-projects/modules.  plugins {  id 'com.android.application' version '7.4.1' apply false  id 'com.android.library' version '7.4.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.0' apply false  }  -----------AndroidManifest-----------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"/>  <uses-permission android:name="android.permission.MANAGE\_EXTERNAL\_STORAGE"/>  <application  android:name=".PdfDownloader"  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.PDFDownloaderWorkManager"  tools:targetApi="31">  <activity  android:name=".ui.main\_screen.MainActivity"  android:exported="true"  android:label="@string/app\_name"  android:theme="@style/Theme.PDFDownloaderWorkManager">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  ----------PdfDownloader(App)-----------  const val CHANNEL\_ID = "channelId"  private const val CHANNEL\_NAME = "Worker channel"  private const val CHANNEL\_DESCRIPTION = "notify users about downloading status"  class PdfDownloader: Application() {  override fun onCreate() {  super.onCreate()  createNotification()  }  private fun createNotification() {  val channel = NotificationChannel(  CHANNEL\_ID,  CHANNEL\_NAME,  NotificationManager.IMPORTANCE\_HIGH  ).apply {  description = CHANNEL\_DESCRIPTION  enableLights(true)  lightColor = Color.RED  }  with(getSystemService(NOTIFICATION\_SERVICE) as NotificationManager) {  createNotificationChannel(channel)  }  }  }  --------------data/api/Downloader--------------  interface Downloader {  @Streaming  @GET  suspend fun download(@Url url: String): Response<ResponseBody>  companion object {  val instance: Downloader by lazy {  val loggingInterceptor = HttpLoggingInterceptor().apply {  level = HttpLoggingInterceptor.Level.BODY  }  val client = OkHttpClient.Builder().addInterceptor(loggingInterceptor).build()  Retrofit.Builder()  .client(client)  .baseUrl("http://localhost/")  .build()  .create(Downloader::class.java)  }  }  }  --------------data/worker/DownloadWorker-------------  class DownloadWorkder(private val context: Context, workerParameters: WorkerParameters):  CoroutineWorker(context, workerParameters) {  override suspend fun doWork(): Result {  return withContext(Dispatchers.IO) {  try {  val filename = inputData.getString(PDF\_FILE\_KEY) ?: "pdf-file.pdf"  handleNotification(context, filename)  val pdfFolderPath = "${context.applicationContext.filesDir}/$FOLDER\_NAME"  File(pdfFolderPath).apply { if (exists().not()) mkdir() }  val pdfFile = File(pdfFolderPath, filename)  val pdfUrl = inputData.getString(PDF\_URL\_DOWNLOAD)  val result = Downloader.instance.download(pdfUrl!!).body()?.let { body ->  body.byteStream().use { inputStream ->  FileOutputStream(pdfFile).use { outputStream ->  val totalBytes = body.contentLength()  val data = ByteArray(8\_192)  var progressBytes = 0L  while (true){  val bytes = inputStream.read(data)  if(bytes == -1) {  break  }  outputStream.write(data, 0, bytes)  progressBytes += bytes  updateNotificationProgress(  context,  ((100 \* progressBytes) / totalBytes).toInt(),  filename,  id  )  }  }  }  val output = workDataOf(FILE\_URI\_KEY to pdfFile.toUri().toString())  Result.success(output)  }?: Result.failure(  workDataOf(  WORKER\_ERROR to ("Unknown error")  )  )  result  } catch (e: Exception) {  e.printStackTrace()  Result.failure(  workDataOf(  WORKER\_ERROR to (e.localizedMessage ?: "Unknown error")  )  )  }  }  }  override suspend fun getForegroundInfo(): ForegroundInfo {  val fileName = inputData.getString(PDF\_FILE\_KEY) ?: "pdf-file.pdf"  return ForegroundInfo(  fileName.hashCode(),  getNotification(context, fileName, id).build()  )  }  }  -------------ui/main\_screen/MainActivity---------  class MainActivity : ComponentActivity() {  val mainViewModel: MainViewModel by viewModels { ViewModelFactory(application = application) }  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  MainScreen(mainVM = mainViewModel)  }  }  }  -------------ui/main\_screen/MainScreen-------------  @Composable  fun MainScreen(mainVM: MainViewModel) {  val workInfo = mainVM.workInfoLiveData.observeAsState().value?.find {  mainVM.downloadPDFWorkderId?.let { id ->  id == it.id  } == true  }  var currentWorkState by remember {  mutableStateOf("")  }    val pdfFiles = mainVM.arrayOfFile()    Column {  Button(onClick = { mainVM.doWork() }) {  Text(text = "Download")  }    Box(modifier = Modifier  .fillMaxWidth()  .wrapContentHeight()  .padding(10.dp)) {  if (pdfFiles.isNullOrEmpty()) {  Text(text = "You currently don't have any pdf", modifier = Modifier.align(Alignment.Center))  } else {  LazyVerticalGrid(columns = GridCells.Fixed(3), modifier = Modifier.fillMaxSize()) {  items(pdfFiles) {item ->  PDFItem(item = item)  }  }  }  }  }  when (workInfo?.state) {  WorkInfo.State.RUNNING -> {  currentWorkState = "Your pdf is downloading"  }  WorkInfo.State.SUCCEEDED -> {  currentWorkState = "Your pdf is downloaded"  }  WorkInfo.State.FAILED -> {  currentWorkState = "Your pdf failed to download"  }  WorkInfo.State.CANCELLED -> {  currentWorkState = "Your pdf download is cancelled"  }  else -> Unit  }  if (workInfo?.state?.isFinished == true) {  workInfo.outputData.getString(FILE\_URI\_KEY)?.let { mainVM.setOutputFile(it) }  }  }  @Composable  fun PDFItem(  item: File  ) {  Column {  Image(  painter = painterResource(id = R.drawable.baseline\_picture\_as\_pdf\_24),  contentDescription = "this is pdf"  )  Spacer(modifier = Modifier.height(10.dp))  Text(text = item.nameWithoutExtension)  }  }  ---------------ui/main\_screen/MainViewModel---------  class MainViewModel(private val application: Application): ViewModel() {  var urlText by mutableStateOf("")  private set  private var workManager = WorkManager.getInstance(application)  var downloadPDFWorkderId: UUID? = null  private set  private var outputFile: File? = null  val workInfoLiveData: LiveData<MutableList<WorkInfo>>  get() = workManager.getWorkInfosByTagLiveData(DOWNLOAD\_PDF\_WORK)  fun doWork() = viewModelScope.launch {  val constraints = Constraints.Builder()  .setRequiredNetworkType(NetworkType.CONNECTED)  .build()  val pdfDownloadWork = OneTimeWorkRequest.Builder(DownloadWorkder::class.java)  .addTag(DOWNLOAD\_PDF\_WORK)  .setInputData(  Data.Builder()  .putString(PDF\_FILE\_KEY, urlText.getFileName())  .putString(PDF\_URL\_DOWNLOAD, TEST\_URL2)  .build()  )  .setExpedited(OutOfQuotaPolicy.RUN\_AS\_NON\_EXPEDITED\_WORK\_REQUEST)  .setConstraints(constraints)  .build()  workManager.enqueueUniqueWork(  urlText.getFileName(),  ExistingWorkPolicy.KEEP,  pdfDownloadWork  )  }  fun setOutputFile(fileUri: String) {  outputFile = File(URI.create(fileUri))  }  fun arrayOfFile(): Array<File>? {  val pdfFolderPath = File("${application.filesDir}/${FOLDER\_NAME}")  val allPdf = pdfFolderPath.listFiles()  return if (pdfFolderPath.exists().not()) {  null  } else {  allPdf  }  }  }  ----------ui/main\_screen/ViewModelFactory----------  class ViewModelFactory(private val application: Application): ViewModelProvider.Factory {  override fun <T : ViewModel> create(modelClass: Class<T>): T {  return when {  modelClass.isAssignableFrom(MainViewModel::class.java) -> MainViewModel(application) as T  else -> throw IllegalArgumentException("Unknown View Model")  }  }  }  ---------util/Constants------  object Constants {  //Testing URLs  const val TEST\_URL1 = "https://www.africau.edu/images/default/sample.pdf"  const val TEST\_URL2 = "https://www.clickdimensions.com/links/TestPDFfile.pdf"  const val TEST\_URL3 = "https://www.orimi.com/pdf-test.pdf"  const val FILE\_URI\_KEY = "a pdf file"  const val PDF\_FILE\_KEY = "file\_key"  const val PDF\_URL\_DOWNLOAD = "url of a pdf to download"  const val FOLDER\_NAME = "App-pdfs"  const val WORKER\_ERROR = "error worker"  const val DOWNLOAD\_PDF\_WORK = "downloading pdf work"  }  ---------util/WorkerUtils----------  suspend fun CoroutineWorker.handleNotification(  context: Context,  fileName: String  ) {  yield()  setForeground(  ForegroundInfo(  fileName.hashCode(),  getNotification(  context,  fileName,  id  ).build()  )  )  }  fun getNotification(  context: Context,  fileName: String,  workderId: UUID  ): NotificationCompat.Builder {  val cancelIntent = WorkManager.getInstance(context).createCancelPendingIntent(workderId)  return NotificationCompat.Builder(context, CHANNEL\_ID)  .setSmallIcon(R.drawable.ic\_launcher\_background)  .setContentTitle("Downloading")  .setContentText("Started download $fileName file")  .setAutoCancel(true)  .setOnlyAlertOnce(true)  .setOngoing(true)  .setAutoCancel(false)  .setOnlyAlertOnce(true)  .addAction(R.drawable.baseline\_cancel\_24, "cancel", cancelIntent)  .setPriority(NotificationCompat.PRIORITY\_HIGH)  }  @SuppressLint("MissingPermission")  fun CoroutineScope.updateNotificationProgress(  context: Context,  progress: Int,  fileName: String,  workderId: UUID  ) {  with(NotificationManagerCompat.from(context)) {  if(isActive) {  notify(  fileName.hashCode(),  getNotification(context, fileName, workderId).setProgress(100, progress, false)  .build()  )  } else {  with(context.getSystemService(Context.NOTIFICATION\_SERVICE) as NotificationManager) {  cancel(fileName.hashCode())  }  }  }  }  fun String.getFileName() = URLDecoder.decode(this, "UTF-8").substringAfterLast("/", "just\_pdf.pdf") | </end> |
| <hitle> | map() vs flatMap | <chare> | 1 | <pext> | class ShoppingBag(val items: List<String>)  val groceryShopping = listOf(  ShoppingBag(listOf("Mango", "Apple", "Banana")),  ShoppingBag(listOf("LadyFinger", "Broccoli", "Onion")),  ShoppingBag(listOf("Spices", "Knife", "Plates"))  )  val clothesShopping = listOf(  ShoppingBag(listOf("Pant", "Shirt", "Tie")),  ShoppingBag(listOf("Socks", "Shoes", "Jeans")),  ShoppingBag(listOf("Jackets", "Caps", "Muffler"))  )  val grocery = groceryShopping.flatMap {  it.items  }  println(grocery)  val clothes = clothesShopping.map {  it.items  }  println(clothes)  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  [Mango, Apple, Banana, LadyFinger, Broccoli, Onion, Spices, Knife, Plates]  [[Pant, Shirt, Tie], [Socks, Shoes, Jeans], [Jackets, Caps, Muffler]] | </end> |
| <hitle> | QuotesPaginationPagingSourceDemo | <chare> | 1 | <pext> | 03-31/QuotesPaginationPagingSourceDemo  ------------Gradle.module---------  plugins {  id 'com.android.application'  id 'org.jetbrains.kotlin.android'  id 'kotlin-kapt'  id 'dagger.hilt.android.plugin'  }  android {  namespace 'com.example.pagingdemo'  compileSdk 33  defaultConfig {  applicationId "com.example.pagingdemo"  minSdk 24  targetSdk 33  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  compileOptions {  sourceCompatibility JavaVersion.VERSION\_1\_8  targetCompatibility JavaVersion.VERSION\_1\_8  }  kotlinOptions {  jvmTarget = '1.8'  }  buildFeatures{  viewBinding true  }  }  dependencies {  implementation 'androidx.core:core-ktx:1.7.0'  implementation 'androidx.appcompat:appcompat:1.6.1'  implementation 'com.google.android.material:material:1.8.0'  implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'androidx.test.ext:junit:1.1.5'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  def hilt\_version="2.45"  implementation "com.google.dagger:hilt-android:$hilt\_version"  kapt "com.google.dagger:hilt-compiler:$hilt\_version"  def lifecycle\_version = "2.6.1"  implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:$lifecycle\_version"  implementation "androidx.lifecycle:lifecycle-livedata-ktx:$lifecycle\_version"  implementation 'androidx.activity:activity-ktx:1.7.0'  def retrofit\_version = "2.9.0"  implementation "com.squareup.retrofit2:retrofit:$retrofit\_version"  implementation "com.squareup.retrofit2:converter-gson:$retrofit\_version"  implementation 'com.squareup.okhttp3:logging-interceptor:5.0.0-alpha.2'  def room\_version = "2.5.1"  implementation "androidx.room:room-runtime:$room\_version"  implementation "androidx.room:room-ktx:$room\_version"  kapt "androidx.room:room-compiler:$room\_version"  def coroutines\_version = "1.6.1"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-core:$coroutines\_version"  implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:$coroutines\_version"  def paging\_version = "3.1.1"  implementation "androidx.paging:paging-runtime:$paging\_version"  }  ------------Gradle.project-----------  // Top-level build file where you can add configuration options common to all sub-projects/modules.  buildscript {  repositories {  google()  mavenCentral()  }  dependencies {  classpath 'com.google.dagger:hilt-android-gradle-plugin:2.45'  }  }  plugins {  id 'com.android.application' version '7.4.1' apply false  id 'com.android.library' version '7.4.1' apply false  id 'org.jetbrains.kotlin.android' version '1.8.10' apply false  }  task clean(type: Delete) {  delete rootProject.buildDir  }  ------------AndroidManifest------------  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET"/>  <application  android:name=".QuotesApp"  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.PagingDemo"  tools:targetApi="31">  <activity  android:name=".presentation.MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  ------------HiltAndroidApp------------  @HiltAndroidApp  class QuotesApp : Application() {  }  ------------util/Constants------------  object Constants {  const val BASE\_URL = "https://quotable.io/"  const val END\_POINT = "quotes"  }  ------------data/api/QuoteApi------------  interface QuoteAPI {  @GET(END\_POINT)  suspend fun getQuotes(@Query("page") page:Int) :Quotes  }  ------------data/api/models/data classes------------  data class Quote(  val \_id: String,  val author: String,  val authorSlug: String,  val content: String,  val dateAdded: String,  val dateModified: String,  val length: Int,  val tags: List<String>  )  data class Quotes(  val count: Int,  val lastItemIndex: Int,  val page: Int,  val results: List<Quote>,  val totalCount: Int,  val totalPages: Int  )  ------------data/paging/QuotePagingSource------------  class QuotePagingSource(private val quoteAPI: QuoteAPI) : PagingSource<Int, Quote>() {  override suspend fun load(params: LoadParams<Int>): LoadResult<Int, Quote> {  return try {  val position = params.key ?: 1  val response = quoteAPI.getQuotes(position)  return LoadResult.Page(  data = response.results,  prevKey = if (position == 1) null else position.minus(1),  nextKey = if (position == response.totalPages) null else position.plus(1)  )  } catch (e: Exception) {  LoadResult.Error(e)  }  }  override fun getRefreshKey(state: PagingState<Int, Quote>): Int? {  return state.anchorPosition?.let { anchorPosition ->  val anchorPage = state.closestPageToPosition(anchorPosition)  anchorPage?.prevKey?.plus(1) ?: anchorPage?.nextKey?.minus(1)  }  }  }  ------------data/paging/QuoteRepository------------  class QuoteRepository @Inject constructor(private val quoteAPI: QuoteAPI) {  fun getQuotes() = Pager(PagingConfig(pageSize = 20, maxSize = 100),  pagingSourceFactory = { QuotePagingSource(quoteAPI) }).liveData  }  ------------di/NetworkModule------------  @Module  @InstallIn(SingletonComponent::class)  object NetworkModule {  @Singleton  @Provides  fun provideLoggingInterceptor(): HttpLoggingInterceptor{  return HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)  }  @Singleton  @Provides  fun provideOkhttpClient(httpLoggingInterceptor: HttpLoggingInterceptor): OkHttpClient {  val okHttpClient = OkHttpClient.Builder()  .addInterceptor(httpLoggingInterceptor)  return okHttpClient.build()  }  @Singleton  @Provides  fun getRetrofit(okHttpClient: OkHttpClient): Retrofit {  return Retrofit.Builder().baseUrl(BASE\_URL).client(okHttpClient)  .addConverterFactory(GsonConverterFactory.create()).build()  }  @Singleton  @Provides  fun getQuotesAPI(retrofit: Retrofit): QuoteAPI {  return retrofit.create(QuoteAPI::class.java)  }  }  ------------presentation/QuotesAdapter------------  class QuotesAdapter : PagingDataAdapter<Quote, QuotesAdapter.QuoteViewHolder>(COMPARATOR) {  override fun onBindViewHolder(holder: QuoteViewHolder, position: Int) {  val item = getItem(position)  item?.let {  holder.quote.text = item.content  holder.quoteAuthor.text = item.author  }  }  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): QuoteViewHolder {  val view = LayoutInflater.from(parent.context).inflate(R.layout.quote\_item, parent, false)  return QuoteViewHolder(view)  }  class QuoteViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  val quote = itemView.findViewById<TextView>(R.id.txtQuote)  val quoteAuthor = itemView.findViewById<TextView>(R.id.txtAuthor)  }  ------------COMPARATOR is important------------  companion object {  private val COMPARATOR = object : DiffUtil.ItemCallback<Quote>() {  override fun areItemsTheSame(oldItem: Quote, newItem: Quote): Boolean {  return oldItem.\_id == newItem.\_id  }  override fun areContentsTheSame(oldItem: Quote, newItem: Quote): Boolean {  return oldItem == newItem  }  }  }  }  ------------presentation/QuoteViewModel------------  @HiltViewModel  class QuoteViewModel @Inject constructor(repository: QuoteRepository) : ViewModel() {  val list = repository.getQuotes().cachedIn(viewModelScope)  }  ------------presentation/MainActivity------------  @AndroidEntryPoint  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private val viewModel: QuoteViewModel by viewModels()  private lateinit var quotesAdapter: QuotesAdapter  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  setUpRecyclerView()  observeData()  }  private fun setUpRecyclerView() {  quotesAdapter = QuotesAdapter()  binding.quoteRecyclerView.apply {  layoutManager = LinearLayoutManager(this@MainActivity)  setHasFixedSize(true)  adapter = quotesAdapter  }  }  private fun observeData() {  viewModel.list.observe(this) {  quotesAdapter.submitData(lifecycle, it)  }  }  }  ------------quote\_item.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="wrap\_content"  android:orientation="vertical"  android:padding="2dp">  <com.google.android.material.floatingactionbutton.FloatingActionButton  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:layout\_margin="10dp"  android:layout\_marginEnd="20dp"  android:backgroundTint="@color/white"  android:clickable="true"  android:contentDescription="@string/app\_name"  android:elevation="20dp"  android:focusable="true"  android:src="@drawable/baseline\_share\_24"  android:tint="@color/white"  app:layout\_anchor="@id/card\_view"  app:layout\_anchorGravity="center"  android:id="@+id/shareQuote"  app:layout\_constraintBottom\_toBottomOf="@+id/card\_view"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/card\_view" />  <androidx.cardview.widget.CardView  android:id="@+id/card\_view"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:padding="10dp"  app:cardBackgroundColor="#1D2B3A"  app:cardCornerRadius="20dp"  app:cardElevation="5dp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical">  <TextView  android:id="@+id/txtQuote"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:fontFamily="cursive"  android:text="@string/my\_quote"  android:textColor="@color/white"  android:textSize="45sp"  android:textStyle="italic" />  <TextView  android:id="@+id/txtAuthor"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="10dp"  android:fontFamily="monospace"  android:gravity="bottom|start"  android:text="@string/my\_quote\_author"  android:textColor="@color/white"  android:textSize="15sp"  android:textStyle="italic" />  </LinearLayout>  </androidx.cardview.widget.CardView>  </androidx.constraintlayout.widget.ConstraintLayout>  ------------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"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".presentation.MainActivity">  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/quoteRecyclerView"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout> | </end> |
| <hitle> | AlgorithmVisualization visualgositelink | <chare> | 1 | <pext> | AlgorithmVisualization  https://visualgo.net/en/sorting | </end> |
| <hitle> | xxxxxxxxxxxxxx | <chare> | 1 | <pext> | xxxxxxxxxxxxxx xxxxxxxxxxxxxx | </end> |
| <hitle> | GooglePreperation AbhishekTeacher KotlinBasics | <chare> | 1 | <pext> | 04-07/GooglePreperation AbhishekTeacher KotlinBasics  https://github.com/cheetahmail007/GooglePreperation | </end> |
| <hitle> | ToastyStartAndBoundService | <chare> | 1 | <pext> | 04-12/ ToastyStartAndBoundService  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.StartAndBoundService"  tools:targetApi="31">  <activity  android:name=".MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  <service android:name=".ToastyService" />  </application>  </manifest>  interface CanBeToasty {  fun shoutToast()  }  -----------ToastyServie--------------  class ToastyService: Service() {  private val binder = ToastyBinder()  private lateinit var customJob: Job  private var toastStarted = false  inner class ToastyBinder: Binder() {  private var toastActivity: CanBeToasty? = null  fun getService() = this@ToastyService  fun setTarget(activity: CanBeToasty) { toastActivity = activity }  fun clearTarget() { toastActivity = null }  fun shoutToast() = toastActivity?.shoutToast()  }  override fun onBind(intent: Intent?) = binder.also { customJob = Job() }  override fun onUnbind(intent: Intent?): Boolean {  customJob.cancel()  return super.onUnbind(intent)  }  fun startToast() {  if (toastStarted) return  toastStarted = true  CoroutineScope(customJob).launch(Dispatchers.Default) {  while(customJob.isActive) {  withContext(Dispatchers.Main) {  binder.shoutToast()  }  delay(1000)  }  }  }  fun stopToast() {  customJob.cancel()  toastStarted = false  }  }  ---------MainActivity--------------  class MainActivity : AppCompatActivity(), CanBeToasty {  private var counter = 0  private lateinit var binding: ActivityMainBinding  private var boundService: ToastyService? = null  private var boundbinder: ToastyService.ToastyBinder? = null  private val connection = object : ServiceConnection {  override fun onServiceConnected(name: ComponentName?, service: IBinder?) {  boundbinder = service as ToastyService.ToastyBinder  boundService = boundbinder?.getService()  boundbinder?.setTarget(this@MainActivity)  }  override fun onServiceDisconnected(name: ComponentName?) {  boundService = null  boundbinder = null  }  }  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnStartService.setOnClickListener {  boundService?.startToast()  }  binding.btnStopService.setOnClickListener {  boundService?.stopToast()  }  }  override fun onResume() {  super.onResume()  bindService(  Intent(this, ToastyService::class.java),  connection,  Context.BIND\_AUTO\_CREATE  )  }  override fun onPause() {  super.onPause()  boundbinder?.clearTarget()  unbindService(connection)  }  override fun shoutToast() {  counter++  Toast.makeText(this, counter.toString(), Toast.LENGTH\_SHORT).show()  }  } | </end> |
| <hitle> | StudentsContentProvider | <chare> | 1 | <pext> | 04-13/StudentsContentProvider  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.StudentsContentProvider"  tools:targetApi="31">  <activity  android:name=".MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  <provider android:name=".StudentsProvider"  android:authorities="com.example.studentscontentprovider.StudentsProvider"/>  </application>  </manifest>  -----------StudentsProvider-------------  class StudentsProvider : ContentProvider() {  /\*\*  \* Database specific constant declarations  \*/  private lateinit var db: SQLiteDatabase  private val studentsMap: HashMap<String, String>? = null  private val uriMatcher = UriMatcher(UriMatcher.NO\_MATCH).apply {  this.addURI(PROVIDER\_NAME, "students", STUDENTS)  this.addURI(PROVIDER\_NAME, "students/#", STUDENT\_ID)  }  /\*\*  \* Helper class that actually creates and manages  \* the provider's underlying data repository.  \*/  private class DatabaseHelper(context: Context?) :  SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {  override fun onCreate(db: SQLiteDatabase) {  db.execSQL(CREATE\_DB\_TABLE)  }  override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {  db.execSQL("DROP TABLE IF EXISTS $STUDENTS\_TABLE\_NAME")  onCreate(db)  }  }  override fun onCreate(): Boolean {  val dbHelper = DatabaseHelper(context)  /\*\*  \* Create a writeable database which will trigger its  \* creation if it doesn't already exist.  \*/  db = dbHelper.writableDatabase  return true  }  override fun insert(uri: Uri, values: ContentValues?): Uri? {  /\*\*  \* Add a new student record  \*/  val rowID = db.insert(STUDENTS\_TABLE\_NAME, "", values)  /\*\*  \* If record is added successfully  \*/  if (rowID > 0) {  val uri = ContentUris.withAppendedId(CONTENT\_URI, rowID)  context?.contentResolver?.notifyChange(uri, null)  return uri  }  throw SQLException("Failed to add a record into $uri")  }  override fun query(  uri: Uri, projection: Array<String>?,  selection: String?, selectionArgs: Array<String>?, sortOrder: String?  ): Cursor? {  val sortOrder = sortOrder?: NAME  val qb = SQLiteQueryBuilder()  qb.tables = STUDENTS\_TABLE\_NAME  when (uriMatcher.match(uri)) {  STUDENTS -> qb.projectionMap = studentsMap  STUDENT\_ID -> qb.appendWhere(PRIMARY\_ID + "=" + uri.pathSegments[1])  else -> {}  }  val cursor = qb.query(  db, projection, selection,  selectionArgs, null, null, sortOrder  )  /\*\*  \* register to watch a content URI for changes  \*/  cursor.setNotificationUri(context!!.contentResolver, uri)  return cursor  }  override fun delete(uri: Uri, selection: String?, selectionArgs: Array<String>?): Int {  var count = 0  count = when (uriMatcher.match(uri)) {  STUDENTS -> db.delete(  STUDENTS\_TABLE\_NAME,  selection,  selectionArgs  )  STUDENT\_ID -> {  val id = uri.pathSegments[1]  db.delete(  STUDENTS\_TABLE\_NAME,  "$PRIMARY\_ID = $id" + if (selection?.isNotEmpty() == true) " AND ($selection)" else "",  selectionArgs  )  }  else -> throw IllegalArgumentException("Unknown URI $uri")  }  context?.contentResolver?.notifyChange(uri, null)  return count  }  override fun update(  uri: Uri, values: ContentValues?,  selection: String?, selectionArgs: Array<String>?  ): Int {  val count = when (uriMatcher.match(uri)) {  STUDENTS -> db.update(  STUDENTS\_TABLE\_NAME,  values,  selection,  selectionArgs  )  STUDENT\_ID -> db.update(  STUDENTS\_TABLE\_NAME,  values,  "$PRIMARY\_ID = ${uri.pathSegments[1]}" + if (selection?.isNotEmpty() == true) " AND ($selection)" else "",  selectionArgs  )  else -> throw IllegalArgumentException("Unknown URI $uri")  }  context!!.contentResolver.notifyChange(uri, null)  return count  }  override fun getType(uri: Uri): String? {  return when (uriMatcher.match(uri)) {  STUDENTS -> "vnd.android.cursor.dir/vnd.example.students"  STUDENT\_ID -> "vnd.android.cursor.item/vnd.example.students"  else -> throw IllegalArgumentException("Unsupported URI: $uri")  }  }  companion object {  const val PROVIDER\_NAME = "com.example.studentscontentprovider.StudentsProvider"  private const val URL = "content://$PROVIDER\_NAME/students"  val CONTENT\_URI: Uri = Uri.parse(URL)  const val PRIMARY\_ID = "\_id"  const val NAME = "name"  const val GRADE = "grade"  const val STUDENTS = 1  const val STUDENT\_ID = 2  const val DATABASE\_NAME = "College"  const val STUDENTS\_TABLE\_NAME = "students"  const val DATABASE\_VERSION = 1  const val CREATE\_DB\_TABLE =  "CREATE TABLE $STUDENTS\_TABLE\_NAME (\_id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT NOT NULL, grade TEXT NOT NULL);"  }  }  ------------MainActivity-------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnAdd.setOnClickListener {  addName()  }  binding.btnRetrive.setOnClickListener {  retrieveStudents()  }  }  private fun addName() {  val values = ContentValues()  values.put(  StudentsProvider.NAME,  binding.editText2.text.toString()  )  values.put(  StudentsProvider.GRADE,  binding.editText3.text.toString()  )  val uri = contentResolver.insert(StudentsProvider.CONTENT\_URI, values)  Toast.makeText(this, uri.toString(), Toast.LENGTH\_SHORT).show()  }  @SuppressLint("Range")  private fun retrieveStudents() {  val url = "content://${StudentsProvider.PROVIDER\_NAME}"  val students = Uri.parse(url)  val cursor = contentResolver.query(students, null, null, null, "name")  cursor?.let {  if (cursor.moveToFirst()) {  do {  Toast.makeText(  this,  cursor.getString(cursor.getColumnIndex( StudentsProvider.PRIMARY\_ID )) + ", " +  cursor.getString(cursor.getColumnIndex( StudentsProvider.NAME )) + ", " +  cursor.getString(cursor.getColumnIndex( StudentsProvider.GRADE )),  Toast.LENGTH\_SHORT  ).show()  } while (cursor.moveToNext())  }  }  }  } | </end> |
| <hitle> | ScrollableComposeLogin KeyboardShowBottomButton | <chare> | 1 | <pext> | 04-13/ScrollableComposeLogin KeyboardShowBottomButton  class MyViewModel: ViewModel() {  var loginUsername by mutableStateOf("")  var loginPassword by mutableStateOf("")  private var loginScrollDelta = 0.0f  val loginScrollState = ScrollableState { loginScrollDelta += it; loginScrollDelta }  }  class MainActivity : ComponentActivity() {  private val vm: MyViewModel by viewModels()  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContent {  ScrollableComposeLoginTheme {  // A surface container using the 'background' color from the theme  Surface(  modifier = Modifier.fillMaxSize(),  color = MaterialTheme.colorScheme.background  ) {  LoginPage(vm)  }  }  }  }  }  @Composable  fun LoginPage(  viewModel: MyViewModel  ) {  Column(  Modifier  .fillMaxSize()  .verticalScroll(rememberScrollState()),  verticalArrangement = Arrangement.SpaceBetween  ) {  Column(  Modifier.padding(16.dp),  verticalArrangement = Arrangement.spacedBy(16.dp),  horizontalAlignment = Alignment.CenterHorizontally  ) {  Image(  painterResource(R.drawable.ic\_launcher\_background),"",  Modifier.size(100.dp)  )  Text("Login", fontSize = 32.sp, fontWeight = FontWeight.Bold)  TextField(  value = viewModel.loginUsername,  onValueChange = {  viewModel.loginUsername = it  },  modifier = Modifier.fillMaxWidth(),  placeholder = { Text("Username") }  )  TextField(  value = viewModel.loginPassword,  onValueChange = {  viewModel.loginPassword = it  },  modifier = Modifier.fillMaxWidth(),  placeholder = { Text("Password") }  )  Spacer(modifier = Modifier.size(10.dp))  Button(onClick = { /\*TODO\*/ }) {  Text("Login")  }  }  Column(  Modifier  .fillMaxWidth(),  horizontalAlignment = Alignment.CenterHorizontally  ) {  TextButton(  onClick = { /\*TODO\*/ }  ) {  Text("Register")  }  }  }  }  @Composable  @Preview(showBackground = true)  private fun PreviewE4LoginPage() {  LoginPage(viewModel = MyViewModel())  }  @Composable  fun Greeting(name: String) {  Text(text = "Hello $name!")  }  @Preview(showBackground = true)  @Composable  fun DefaultPreview() {  ScrollableComposeLoginTheme {  Greeting("Android")  }  } | </end> |
| <hitle> | NoInternetBroadcastReceiver | <chare> | 1 | <pext> | 04-14/NoInternetBroadcastReceiver  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.NoInternetBroadcast"  tools:targetApi="31">  <activity  android:name=".MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  ------------ConnectivityReceiver---------------  class ConnectivityReceiver: BroadcastReceiver() {  override fun onReceive(context: Context?, intent: Intent?) {  if (connectivityReceiverListener != null) {  connectivityReceiverListener!!.onNetworkConnectionChanged(isConnectedOrConnecting(context!!))  }  }  private fun isConnectedOrConnecting(context: Context): Boolean {  var result = false  val connectivityManager =  context.getSystemService(Context.CONNECTIVITY\_SERVICE) as ConnectivityManager  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {  val networkCapabilities = connectivityManager.activeNetwork ?: return false  val actNw =  connectivityManager.getNetworkCapabilities(networkCapabilities) ?: return false  result = when {  actNw.hasTransport(NetworkCapabilities.TRANSPORT\_WIFI) -> true  actNw.hasTransport(NetworkCapabilities.TRANSPORT\_CELLULAR) -> true  actNw.hasTransport(NetworkCapabilities.TRANSPORT\_ETHERNET) -> true  else -> false  }  } else {  connectivityManager.run {  connectivityManager.activeNetworkInfo?.run {  result = when (type) {  ConnectivityManager.TYPE\_WIFI -> true  ConnectivityManager.TYPE\_MOBILE -> true  ConnectivityManager.TYPE\_ETHERNET -> true  else -> false  }  }  }  }  return result  }  interface ConnectivityReceiverListener {  fun onNetworkConnectionChanged(isConnected: Boolean)  }  companion object {  var connectivityReceiverListener: ConnectivityReceiverListener? = null  }  }  -----------MainActivity---------------  class MainActivity : AppCompatActivity(),  ConnectivityReceiver.ConnectivityReceiverListener {  private lateinit var binding: ActivityMainBinding  private var snackBar: Snackbar? = null  private var testConnection = false;  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  registerReceiver(ConnectivityReceiver(), IntentFilter(ConnectivityManager.CONNECTIVITY\_ACTION))  binding.btnCheck.setOnClickListener {  // val testSnackbar = Snackbar.make(this, binding.root, "hey this is snack bar", Snackbar.LENGTH\_LONG)  // testSnackbar.apply {  // setAction("Retry") {  // Toast.makeText(this@MainActivity, "Hello", Toast.LENGTH\_SHORT).show()  // }  // }  // testSnackbar.show()  testConnection = !testConnection  showNetworkMessage(testConnection)  }  }  override fun onResume() {  super.onResume()  ConnectivityReceiver.connectivityReceiverListener = this  }  override fun onNetworkConnectionChanged(isConnected: Boolean) {  showNetworkMessage(isConnected)  }  private fun showNetworkMessage(isConnected: Boolean) {  if (!isConnected) {  snackBar = Snackbar.make(this@MainActivity, binding.root, "No internet!!", Snackbar.LENGTH\_LONG) //Assume "rootLayout" as the root layout of every activity.  snackBar?.duration = BaseTransientBottomBar.LENGTH\_INDEFINITE  snackBar?.setAction("Retry") {  }  snackBar?.show()  } else {  snackBar?.dismiss()  }  }  } | </end> |
| <hitle> | AlarmBroadcastReceiverPractice | <chare> | 1 | <pext> | 04-18/AlarmBroadcastReceiverPractice  <receiver android:name=".MyAlarmReceiver" />  </application>  </manifest>  class MyAlarmReceiver: BroadcastReceiver() {  override fun onReceive(context: Context?, intent: Intent?) {  Toast.makeText(context, "Alarm Triggered", Toast.LENGTH\_SHORT).show()  val mediaPlayer = MediaPlayer.create(context, Settings.System.DEFAULT\_ALARM\_ALERT\_URI)  mediaPlayer.isLooping = true  mediaPlayer.start()  }  }  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var alarmManager: AlarmManager  private lateinit var pendingIntent: PendingIntent  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  initViews()  }  private fun setAlarmManager(hour: Int, minute: Int) {  alarmManager = getSystemService(ALARM\_SERVICE) as AlarmManager  val intent = Intent(this, MyAlarmReceiver::class.java)  pendingIntent = PendingIntent.getBroadcast(  this,  REQUEST\_CODE,  intent,  PendingIntent.FLAG\_IMMUTABLE  )  val calendar = Calendar.getInstance()  calendar.timeInMillis = System.currentTimeMillis()  calendar.set(Calendar.HOUR\_OF\_DAY, hour)  calendar.set(Calendar.MINUTE, minute)  alarmManager.setRepeating(  AlarmManager.RTC,  calendar.timeInMillis,  AlarmManager.INTERVAL\_DAY,  pendingIntent  )  }  private fun initViews() {  binding.btnTimePicker.setOnClickListener {  val calendar = Calendar.getInstance()  val hour = calendar.get(Calendar.HOUR)  val minute = calendar.get(Calendar.MINUTE)  val timePicker = TimePickerDialog(  this,  { \_, selectedHour, selectedMinute ->  setAlarmManager(selectedHour, selectedMinute)  }, hour, minute, true  )  timePicker.show()  }  }  override fun onDestroy() {  super.onDestroy()  if(this::alarmManager.isInitialized) {  alarmManager.cancel(pendingIntent)  }  }  companion object {  const val REQUEST\_CODE = 100  }  } | </end> |
| <hitle> | NewEditpoint | <chare> | 1 | <pext> | https://github.com/cheetahmail007/Whatsapp-Chatting-Sample\_Work | </end> |
| <hitle> | lasteditpoint | <chare> | 1 | <pext> | Share | </end> |
| <hitle> | lasteditpoint | <chare> | 1 | <pext> | Share | </end> |
| <hitle> | FBNotesDBPractice Firebase Realtime Database | <chare> | 1 | <pext> | 04-24/FBNotesDBPractice Firebase Realtime Database  data class Note (  val id: String = "",  val title: String = "",  val body: String = ""  )  class NotesAdapter(private val context: Context, private val notes: List<Note>): RecyclerView.Adapter<NotesAdapter.NotesViewHolder>() {  private lateinit var binding: ItemNoteBinding  override fun getItemCount() = notes.size  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NotesViewHolder {  val layoutInflater = LayoutInflater.from(parent.context)  binding = ItemNoteBinding.inflate(layoutInflater, parent, false)  return NotesViewHolder(binding.root)  }  override fun onBindViewHolder(holder: NotesViewHolder, position: Int) {  holder.bind(notes[position])  }  inner class NotesViewHolder(val view: View): RecyclerView.ViewHolder(view) {  fun bind(note: Note) {  binding.txtId.text = note.id  binding.txtTitle.text = note.title  binding.txtBody.text = note.body  binding.btnDelete.setOnClickListener {  val parent = context as MainActivity?  parent?.deleteNote(note.id)  }  }  }  }  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:orientation="vertical"  android:padding="2dp"  android:layout\_margin="2dp"  android:background="#ECECEC"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="horizontal"  >  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:id="@+id/txtId"  android:text="1"  />  <TextView  android:layout\_marginLeft="4dp"  android:layout\_marginRight="4dp"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:id="@+id/txtTitle"  android:text="First Title"  android:textSize="20sp"  android:layout\_weight="1"  />  <ImageButton  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:backgroundTint="@android:color/transparent"  android:id="@+id/btnDelete"  android:src="@drawable/baseline\_delete\_24"  />  </LinearLayout>  <TextView  android:id="@+id/txtBody"  android:layout\_marginTop="4dp"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="First Body"  />  </LinearLayout>  \*\*\*MainActivity\*\*\*  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private lateinit var dbReference: DatabaseReference  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  -------get the instance of FirebaseDatabase----------  -------get the reference of table/node/child----------  dbReference = FirebaseDatabase.getInstance().getReference("Notes")  -------register value change event listener-----------  dbReference.addValueEventListener(object: ValueEventListener {  override fun onDataChange(snapshot: DataSnapshot) {  onDBChanged(snapshot)  }  override fun onCancelled(error: DatabaseError) {  }  })  -------register table change event listener-----------  dbReference.orderByKey().addValueEventListener( object: ValueEventListener {  override fun onDataChange(snapshot: DataSnapshot) {  updateUIFromDB(snapshot)  }  override fun onCancelled(error: DatabaseError) {  }  })  binding.rvNoteList.layoutManager = LinearLayoutManager(this)  binding.btnApply.setOnClickListener { applyNote() }  }  private fun applyNote() {  val id = binding.editId.text.toString()  if (id.isNotEmpty()) {  val title = binding.editTitle.text.toString()  val body = binding.editBody.text.toString()  --------add/set value for given node------------  dbReference.child(id).setValue(Note(id, title, body))  } else {  Toast.makeText(this, "ID is invalid!", Toast.LENGTH\_SHORT).show()  }  }  fun onDBChanged(snapshot: DataSnapshot) {  Toast.makeText(this, "DB changed!", Toast.LENGTH\_SHORT).show()  }  private fun updateUIFromDB(snapshot: DataSnapshot) {  -----------DataSnapshot convert to Class Object List-------  val notes = mutableListOf<Note>()  snapshot.children.forEach {  val note = it.getValue(Note::class.java)  note?.let {  notes.add(note)  }  }  binding.rvNoteList.adapter = NotesAdapter(this, notes)  }  fun deleteNote(id: String) {  --------------Delete note/child/table----------  dbReference.child(id).removeValue()  }  }  \*\*\*activity\_main.xml\*\*\*  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout 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"  android:orientation="vertical"  tools:context=".MainActivity">  <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical"  >  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editId"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="4dp"  android:hint="ID"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editTitle"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="4dp"  android:hint="Title"  />  <androidx.appcompat.widget.AppCompatEditText  android:id="@+id/editBody"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_margin="4dp"  android:hint="Body"  />  <Button  android:id="@+id/btnApply"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:textSize="24sp"  android:text="Apply"  />  </LinearLayout>  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/rvNoteList"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:layout\_weight="1"  />  </LinearLayout> | </end> |
| <hitle> | ContentProviderImageMedia /Git | <chare> | 1 | <pext> | 04-26/ContentProviderImageMedia  <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">  <uses-permission android:name="android.permission.READ\_CONTACTS" />  <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />  <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />  <uses-permission android:name="android.permission.READ\_MEDIA\_IMAGES" />  <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:supportsRtl="true"  android:theme="@style/Theme.ContentProviderImage"  tools:targetApi="31">  <activity  android:name=".MainActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  </manifest>  -----------MainActivity---------------  class MainActivity : AppCompatActivity() {  private lateinit var binding: ActivityMainBinding  private val readImagePermission = if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.TIRAMISU) Manifest.permission.READ\_MEDIA\_IMAGES else Manifest.permission.READ\_EXTERNAL\_STORAGE  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  binding = ActivityMainBinding.inflate(layoutInflater)  setContentView(binding.root)  binding.btnFetch.setOnClickListener {  loadImages()  }  }  private fun loadImages() {  if(ContextCompat.checkSelfPermission(this, readImagePermission) == PackageManager.PERMISSION\_GRANTED){  getImages()  } else {  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.TIRAMISU) {  requestPermissions(  arrayOf(Manifest.permission.READ\_MEDIA\_IMAGES),  REQUEST\_READ\_PERMISSION  )  } else {  requestPermissions(  arrayOf(Manifest.permission.READ\_EXTERNAL\_STORAGE),  REQUEST\_READ\_PERMISSION  )  }  }  }  private fun getImages() {  val cursor = if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.TIRAMISU) {  contentResolver.query(MediaStore.Images.Media.getContentUri( MediaStore.VOLUME\_EXTERNAL ), null, null, null, null)  } else {  contentResolver.query(MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI, null, null, null, null)  }  var absolutePathOfImage = ""  cursor?.let {  if (it.count > 0) {  it.moveToNext()  val columnIndexData = it.getColumnIndexOrThrow(MediaStore.MediaColumns.DATA);  absolutePathOfImage = it.getString(columnIndexData)  }  }  val myBitmap = BitmapFactory.decodeFile(absolutePathOfImage)  binding.imageView.setImageBitmap(myBitmap)  }  companion object {  const val REQUEST\_READ\_PERMISSION = 112  }  }  --------------activity\_main.xml-------------  <?xml version="1.0" encoding="utf-8"?>  <LinearLayout 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"  android:orientation="vertical"  tools:context=".MainActivity">  <TextView  android:id="@+id/txtContacts"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Hello World!"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <Button  android:id="@+id/btnFetch"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Fetch"  />  <ImageView  android:id="@+id/imageView"  android:layout\_width="match\_parent"  android:layout\_height="225dp"  app:srcCompat="@drawable/ic\_launcher\_background" />  </LinearLayout> | </end> |
| <hitle> | FinalJetpackComposeAllOneGuideWeather/Git | <chare> | 1 | <pext> | 05-13/ FinalJetpackComposeAllOneGuideWeather/Git  https://github.com/cheetahmail007/Jetpack-Compose-All-In-One-Guide | </end> |
| <hitle> | MyTomGroceryProject /Git | <chare> | 1 | <pext> | 05-13/ MyTomGroceryProject/Git  https://github.com/Howdyev/TomGrocery | </end> |
| <hitle> | xxxxxxxxxxxxxx | <chare> | 1 | <pext> | xxxxxxxxxxxxxxxxxxxxxxxxx | </end> |