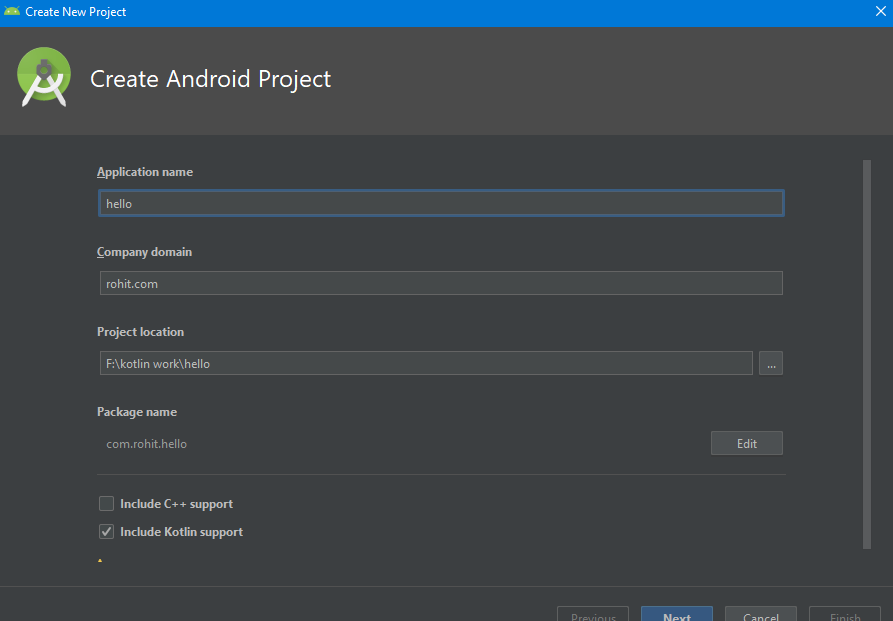
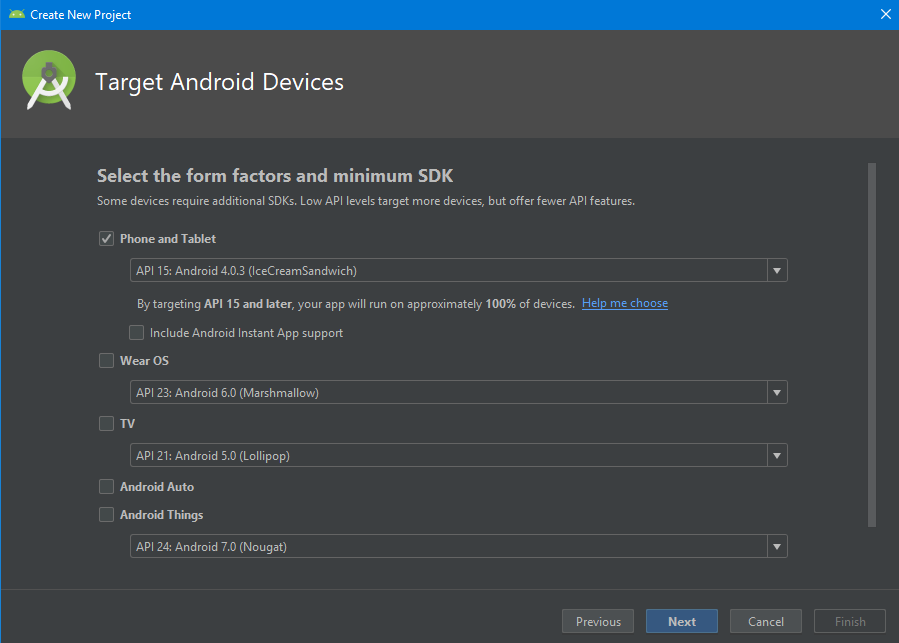
**PRACTICAL 1**

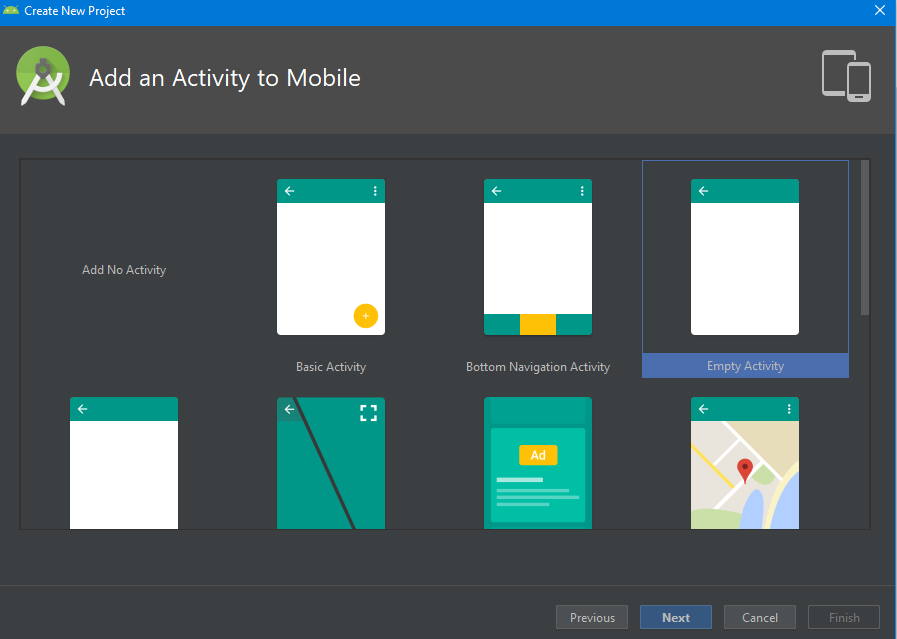
1. Introduction to Android, Introduction to Android Studio IDE, Application Fundamentals: Creating a Project, Android Components, Activities, Services, Content Providers, Broadcast Receivers, Interface overview, Creating Android Virtual device, USB debugging mode, Android Application Overview. Simple “Hello World” program.

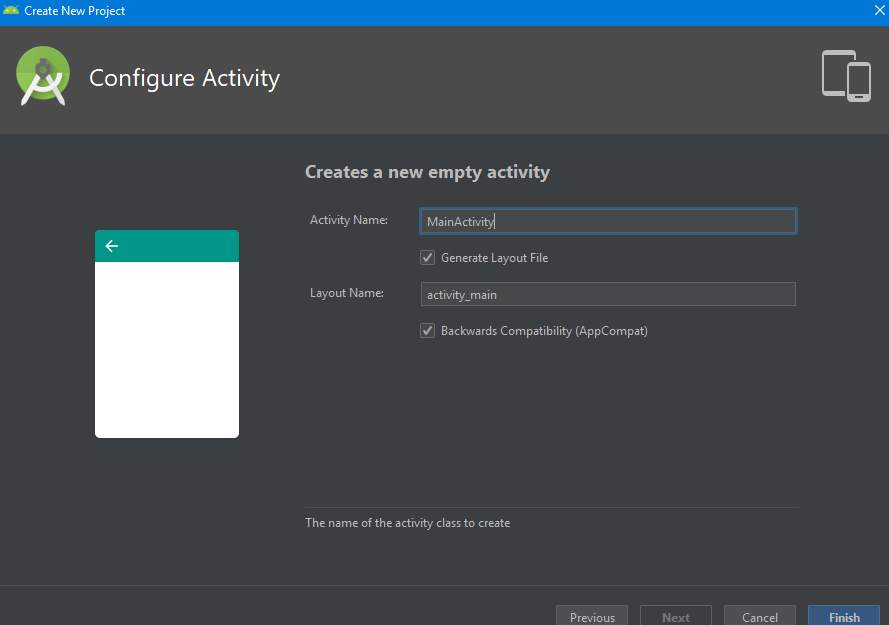
**Solution:**

Creating a project:









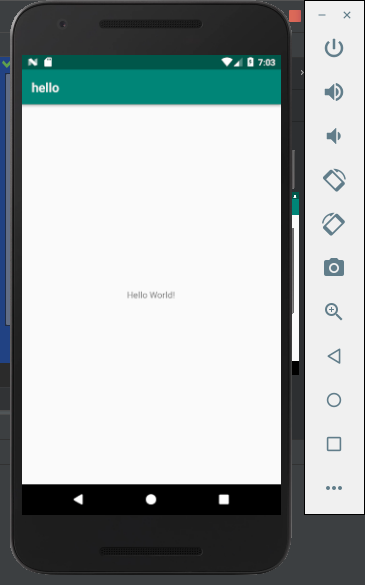
Activity\_Main.Kt

package com.rohit.hello  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 }  
}

Activity\_Main.xml

<?xml version="1.0" encoding="utf-8"?>  
<android.support.constraint.ConstraintLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"/>  
  
</android.support.constraint.ConstraintLayout>

Apk in avd:



**BroadcastActivity:**

How to receiving Broadcast

Apps can receive and android BroadcastReceiver in two ways: through manifest-declared receivers and context-registered receivers. In this example, we are approaching manifest-declared Receiver. Learn step by step to the kotlin broadcast receiver example works.

Step 1. Create an android app, For creating an Android app with kotlin read this tutorial.

Step 2. Creating Broadcast Receiver

Create and extend Subclass and BroadcastReceiver implement. onReceive(Context, Intent) where onReceive method each message is received as an Intent object parameter.

MyReceiver.kt:

import android.content.BroadcastReceiver

import android.content.Context

import android.content.Intent

import android.widget.Toast

class MyReceiver : BroadcastReceiver() {

override fun onReceive(context: Context, intent: Intent) {

// TODO: This method is called when the BroadcastReceiver is receiving

// an Intent broadcast.

Toast.makeText(context, "Broadcast : Flight mode changed.",

Toast.LENGTH\_LONG).show()

}

}

3.Declare a broadcast receiver in the manifest file

add the element<receiver> in your app’s manifest. Here is code snap

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="in.eyehunt.androidbroadcasts">

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/AppTheme">

<activity android:name=".MainActivity">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<receiver

android:name=".MyReceiver"

android:enabled="true"

android:exported="true">

<intent-filter>

<action android:name="android.intent.action.AIRPLANE\_MODE"/>

</intent-filter>

</receiver>

</application>

</manifest>

Note: If the app is not running and broadcast receiver declared in AndroidManifest.xml, then the system will launch your app.

Step 4. MainActivity code, no needs to do anything

MainActivity.kt:

import android.support.v7.app.AppCompatActivity

import android.os.Bundle

class MainActivity : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

}

}

Step 5. Add following code in main\_activity.xml

add <ImageView> and <TextView>widget layout file.

main\_activity.xml:

<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@color/colorPrimary"

tools:context="in.eyehunt.androidbroadcasts.MainActivity">

<ImageView

android:id="@+id/imageView"

android:layout\_width="40dp"

android:layout\_height="40dp"

android:layout\_margin="8dp"

android:layout\_marginTop="16dp"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:srcCompat="@mipmap/baseline\_airplanemode\_active\_white\_24" />

<TextView

android:id="@+id/textView"

android:layout\_width="300dp"

android:layout\_height="36dp"

android:layout\_marginEnd="8dp"

android:layout\_marginStart="8dp"

android:gravity="center\_vertical"

android:text="Flight Mode"

android:textColor="@color/colorWhite"

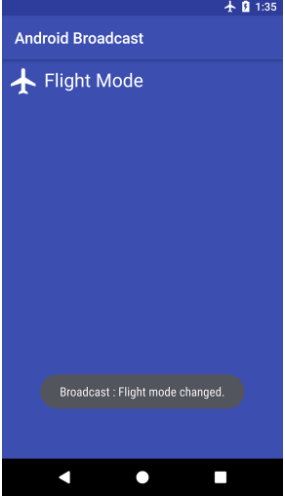
android:textSize="24dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toEndOf="@+id/imageView"

app:layout\_constraintTop\_toTopOf="@+id/imageView" />

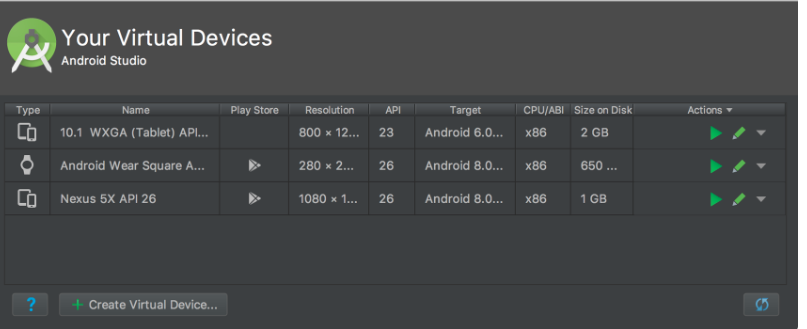
</android.support.constraint.ConstraintLayout>

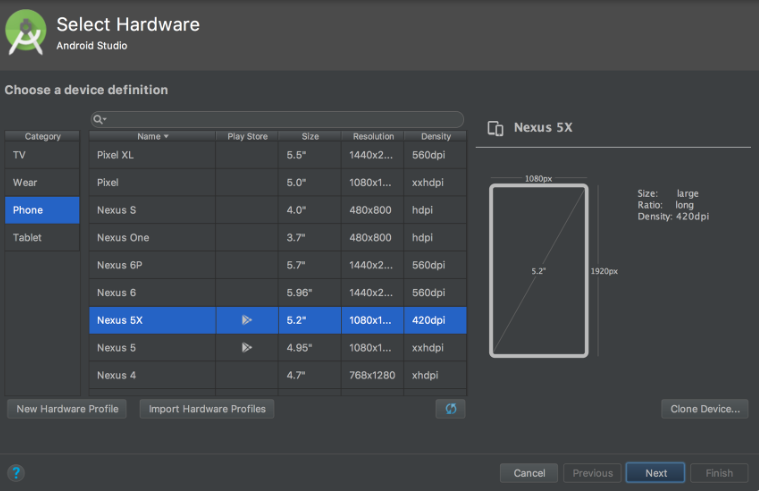


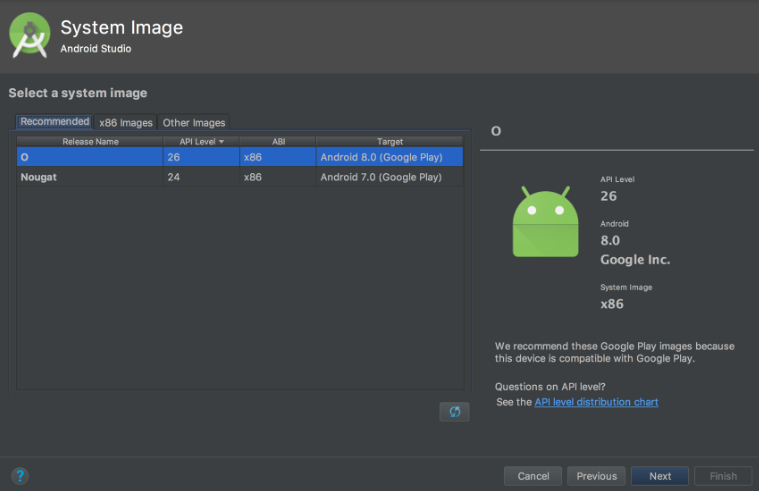
**Create and manage virtual devices:**

To open the AVD Manager, do one of the following:

* Select Tools > AVD Manager.
* Click AVD Manager AVD Manager icon in the toolbar.







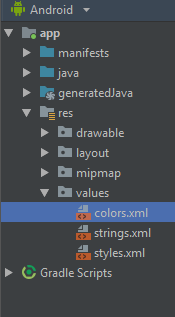


**PRACTICAL 2**

**Programming Resources**

Android Resources: (Color, Theme, String, Drawable, Dimension, Image).

**Color:**

****

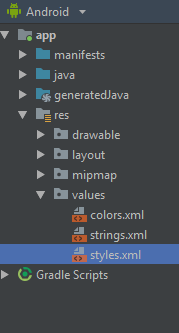
**Color.xml**

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="colorPrimary">#008577</color>  
 <color name="colorPrimaryDark">#00574B</color>  
 <color name="colorAccent">#D81B60</color>  
</resources>

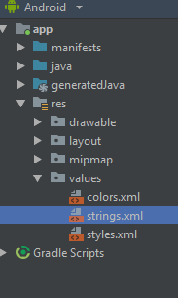
**Theme:**

**Style.xml**

<resources>  
  
 <!-- Base application theme. -->  
 <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">  
 <!-- Customize your theme here. -->  
 <item name="colorPrimary">@color/colorPrimary</item>  
 <item name="colorPrimaryDark">@color/colorPrimaryDark</item>  
 <item name="colorAccent">@color/colorAccent</item>  
 </style>  
  
</resources>

****

**String:**

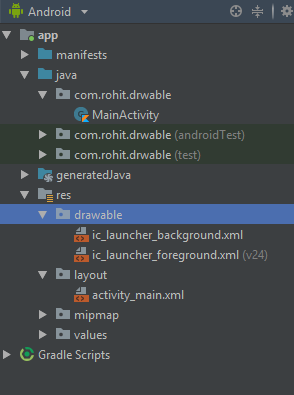
****

**String.xml:**

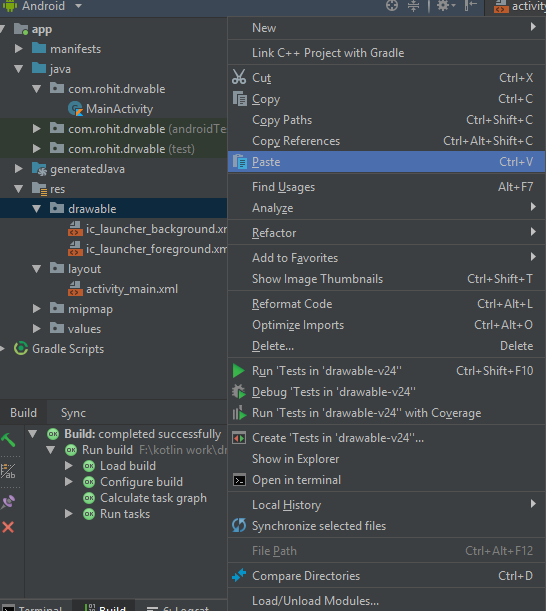
<resources>  
 <string name="app\_name">hello</string>  
 <string name="numbers">  
 <item>1</item>  
 <item>2</item>  
 <item>3</item>  
 </item>  
 </string>  
</resources>

**Drawable:**

1. Right click on drawable folder

****

1. Copy the image if you want to create image drawable
2. Paste that image file inside the drawable folder

****

**Note: to create drawable resource, right click on drawable folder and select drawable resource file.**

**Dimension, Image:**

**Main\_Activity.kt:**

package com.rohit.drwable  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 }  
}

**activity\_main.xml:**

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity"  
 android:background="@drawable/one">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"/>  
  
</LinearLayout>

**Output:**

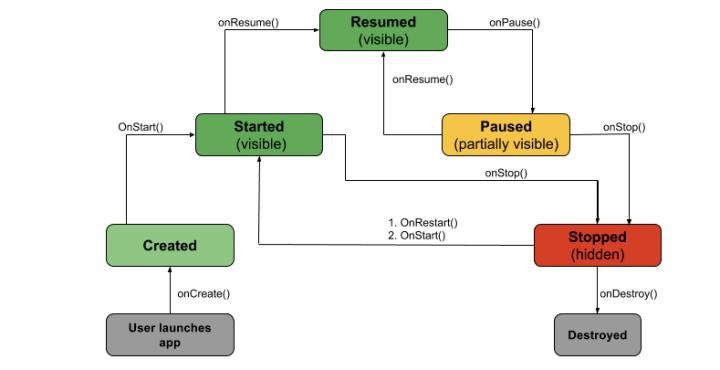
****

**PRACTICAL 3**

**Programming Activities and fragments**

Activity Life Cycle, Activity methods, Multiple Activities, Life Cycle of fragments and multiple fragments.

**Activity Lifecycle:**



• **onCreate():** Called by the OS when the activity is first created. This is

where you initialize any UI elements or data objects. You also have the savedInstanceState of the activity that contains its previously saved state, and you can use it to recreate that state.\

fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.*layout*.*activity\_task\_description*)

• **onStart():** Just before presenting the user with an activity, this method is called. It’s always followed by onResume(). In here, you generally should

start UI animations, audio based content or anything else that requires the activity’s contents to be on screen.

•

• **onResume():** As an activity enters the foreground, this method is called.

Here you have a good place to restart animations, update UI elements, restart camera previews, resume audio/video playback or initialize any components that you release during onPause().

• **onPause():** This method is called before sliding into the background. Here you should stop any visuals or audio associated with the activity such as UI animations, music playback or the camera. This method is followed by

onResume() if the activity returns to the foreground or by onStop() if it becomes hidden.

• **onStop():** This method is called right after onPause(), when the activity is no longer visible to the user, and it’s a good place to save data that you want to commit to the disk. It’s followed by either onRestart(), if this activity is

coming back to the foreground, or onDestroy() if it’s being released from memory.

• **onRestart():** Called after stopping an activity, but just before starting it again. It’s always followed by onStart().

• **onDestroy():** This is the final callback you’ll receive from the OS before the activity is destroyed. You can trigger an activity’s desctruction by calling

finish(), or it can be triggered by the system when the system needs to

recoup memory. If your activity includes any background threads or other long-running resources, destruction could lead to a memory leak if they’re not released, so you need to remember to stop these processes here as well.

EXAMPLE:

import android.os.Bundle

import android.support.design.widget.Snackbar

import android.support.v7.app.AppCompatActivity

import android.view.Menu

import android.view.MenuItem

import android.util.Log

import kotlinx.android.synthetic.main.activity\_state\_change.\*

class StateChangeActivity : AppCompatActivity() {

val TAG = "StateChange"

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.*layout*.*activity\_state\_change*)

setSupportActionBar(toolbar)

fab.setOnClickListener **{** view **->**

Snackbar.make(view, "Replace with your own action",

Snackbar.*LENGTH\_LONG*)

.setAction("Action", null).show()

**}**

Log.i(TAG, "onCreate")

}

}

override fun onStart() {

super.onStart()

Log.i(TAG, "onStart")

}

override fun onResume() {

super.onResume()

Log.i(TAG, "onResume")

}

override fun onPause() {

super.onPause()

Log.i(TAG, "onPause")

}

override fun onStop() {

super.onStop()

Log.i(TAG, "onStop")

}

override fun onRestart() {

super.onRestart()

Log.i(TAG, "onRestart")

}

override fun onDestroy() {

super.onDestroy()

Log.i(TAG, "onDestroy")

}

override fun onSaveInstanceState(outState: Bundle?) {

super.onSaveInstanceState(outState)

Log.i(TAG, "onSaveInstanceState")

}

override fun onRestoreInstanceState(savedInstanceState: Bundle?) {

super.onRestoreInstanceState(savedInstanceState)

Log.i(TAG, "onRestoreInstanceState")

}

**Multiple Activities:**

**activity\_first.xml code:**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

xmlns:android="<http://schemas.android.com/apk/res/android>"

xmlns:tools="<http://schemas.android.com/tools>"

xmlns:app="<http://schemas.android.com/apk/res>-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context="ganeshannt.frist.FristActivity">

<Button

android:id="@+id/button2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:onClick="Ganesh"

android:text="click third activity"

android:textColor="@color/colorPrimary"

app:layout\_constraintTop\_toTopOf="parent"

tools:layout\_editor\_absoluteX="168dp"

android:layout\_alignParentBottom="true"

android:layout\_toEndOf="@+id/text"

android:layout\_marginBottom="196dp" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="This s my first app!"

android:id="@+id/text"

tools:layout\_editor\_absoluteY="8dp"

tools:layout\_editor\_absoluteX="8dp" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/button"

android:text="click second activity"

android:textColor="@color/colorPrimary"

android:onClick="Ganesh"

tools:layout\_editor\_absoluteX="168dp"

app:layout\_constraintTop\_toTopOf="parent"

android:layout\_above="@+id/button2"

android:layout\_alignStart="@+id/button2"

android:layout\_marginBottom="40dp" />

</RelativeLayout>

**activity\_second.xml code:**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>" android:orientation="vertical" android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_margin="20pt"

android:text="second acticity is working...."

android:textAllCaps="true"

android:textColor="@color/colorPrimaryDark"/>

</LinearLayout>

**activity\_third.xml code:**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>" android:orientation="vertical" android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_margin="20pt"

android:text="Third activity is working ........."

android:textAllCaps="true"

android:textColor="@color/colorPrimary"

/>

</LinearLayout>

**Activity\_first.kt**

package rohit.technobeat

import android.content.Intent

import android.support.v7.app.AppCompatActivity

import android.os.Bundle

import kotlinx.android.synthetic.main.activity\_login.\*

import kotlinx.android.synthetic.main.activity\_main.\*

import kotlinx.android.synthetic.main.activity\_register.\*

import rohit.technobeat.R.id.*login*

import rohit.technobeat.R.id.*newaccount*

class MainActivity : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.*activity\_main*)

second.setOnClickListener **{**

val intent = Intent(this, Activity\_second::class.*java*)

// start your next activity

startActivity(intent)

**}**

third.setOnClickListener **{**

val intent = Intent(this, Activity\_third::class.*java*)

// start your next activity

startActivity(intent)

**}**

}

}

**PRACTICAL 4**

**Programs related to different Layouts**

Coordinate, Linear, Relative, Table, Absolute, Frame, List View, Grid View.

1. linear layout:

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
android:layout\_width="fill\_parent"  
android:layout\_height="fill\_parent"  
android:orientation="vertical" >  
  
<Button android:id="@+id/btnStartService"  
android:layout\_width="270dp"  
android:layout\_height="wrap\_content"  
android:text="start\_service"/>  
  
<Button android:id="@+id/btnPauseService"  
android:layout\_width="270dp"  
android:layout\_height="wrap\_content"  
android:text="pause\_service"/>  
  
<Button android:id="@+id/btnStopService"  
android:layout\_width="270dp"  
android:layout\_height="wrap\_content"  
android:text="stop\_service"/>  
  
</LinearLayout>

1. Relative:

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
android:layout\_width="fill\_parent"  
android:layout\_height="fill\_parent"  
android:paddingLeft="16dp"  
android:paddingRight="16dp" >  
  
<EditText  
android:id="@+id/name"  
android:layout\_width="fill\_parent"  
android:layout\_height="wrap\_content"  
android:hint="@string/reminder" />  
  
<LinearLayout  
android:orientation="vertical"  
android:layout\_width="fill\_parent"  
android:layout\_height="fill\_parent"  
android:layout\_alignParentStart="true"  
android:layout\_below="@+id/name">  
  
<Button  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:text="New Button"  
android:id="@+id/button" />  
  
<Button  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:text="New Button"  
android:id="@+id/button2" />  
  
</LinearLayout>  
  
</RelativeLayout>

1. Table:

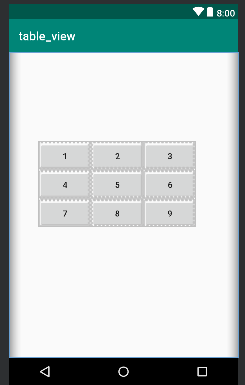
Activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <TableLayout android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="50dp"  
 android:layout\_marginTop="150dp">  
 <TableRow>  
 <Button  
 android:id="@+id/btn1"  
 android:text="1"  
 android:layout\_gravity="center"  
 />  
 <Button  
 android:id="@+id/btn2"  
 android:text="2"  
 android:layout\_gravity="center"  
 />  
 <Button  
 android:id="@+id/btn3"  
 android:text="3"  
 android:layout\_gravity="center"  
 />  
 </TableRow>  
 <TableRow>  
 <Button  
 android:id="@+id/btn4"  
 android:text="4"  
 android:layout\_gravity="center"  
 />  
 <Button  
 android:id="@+id/btn5"  
 android:text="5"  
 android:layout\_gravity="center"  
 /><Button  
 android:id="@+id/btn6"  
 android:text="6"  
 android:layout\_gravity="center"  
 />  
 </TableRow>  
 <TableRow>  
 <Button  
 android:id="@+id/btn7"  
 android:text="7"  
 android:layout\_gravity="center"  
 />  
 <Button  
 android:id="@+id/btn8"  
 android:text="8"  
 android:layout\_gravity="center"  
 /><Button  
 android:id="@+id/btn9"  
 android:text="9"  
 android:layout\_gravity="center"  
 />  
 </TableRow>  
 </TableLayout>  
  
</LinearLayout>

Activity\_main.kt

package com.r.table\_view  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import kotlinx.android.synthetic.main.activity\_main.\*  
import org.jetbrains.anko.toast  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 btn1.setOnClickListener **{** *toast*("1")  
 **}** btn2.setOnClickListener **{** *toast*("2")  
 **}** btn3.setOnClickListener **{** *toast*("3")  
 **}** btn4.setOnClickListener **{** *toast*("4")  
 **}** btn5.setOnClickListener **{** *toast*("5")  
 **}** btn6.setOnClickListener **{** *toast*("6")  
 **}** btn7.setOnClickListener **{** *toast*("7")  
 **}** btn8.setOnClickListener **{** *toast*("8")  
 **}** btn9.setOnClickListener **{** *toast*("9")  
 **}** }  
}

output:



1. Frame:

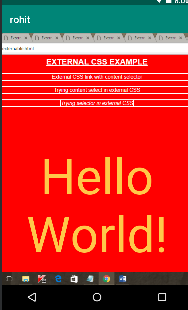
Activity\_main.xml

<?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=".MainActivity">  
  
 <ImageView android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:src="@drawable/red"  
 android:scaleType="centerCrop"/>  
 <TextView  
 android:textSize="100dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!"  
 android:gravity="center"  
 android:textColor="@color/rohit"  
 android:layout\_marginTop="220dp"  
 />  
  
</FrameLayout>

Activity\_main.kt

package com.rohit.frame\_layout  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 }  
}

output:



1. List View:

Activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <Button android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/btn"  
 android:text="Click me to view list"  
 android:layout\_marginTop="200dp"  
 android:layout\_marginLeft="90dp"/>  
</LinearLayout>

String.xml

<resources>  
 <string name="app\_name">list</string>  
  
 <array name="insert\_list">  
 <item>one</item>  
 <item>two</item>  
 <item>three</item>  
 <item>four</item>  
 <item>five</item>  
 <item>six</item>  
 <item>seven</item>  
 <item>eight</item>  
 <item>nine</item>  
 <item>ten</item>  
 </array>  
</resources>

Activity\_list\_view.xml:

<?xml version="1.0" encoding="utf-8"?>  
<ListView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".list\_view" android:entries="@array/insert\_list">  
  
</ListView>

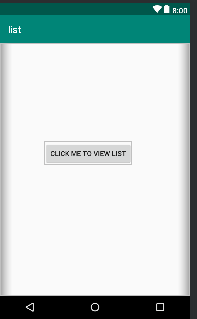
List\_view.kt:

package com.rohit.list  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
  
class list\_view : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_list\_view*)  
 }  
}

main\_Activity.kt

package com.rohit.list  
  
import android.content.Intent  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import kotlinx.android.synthetic.main.activity\_main.\*  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 btn.setOnClickListener **{** val intent =Intent(this, list\_view::class.*java*)  
 startActivity(intent)  
 **}** }  
}

output:

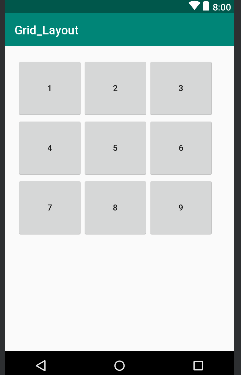


1. Grid layout:
2. <?xml version="1.0" encoding="utf-8"?>  
   <GridLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:tools="http://schemas.android.com/tools"  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"  
    tools:context=".MainActivity"  
    android:rowCount="3"  
    android:columnCount="3"  
    android:padding="20dp">  
     
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="1"/>  
     
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="2"/>  
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="3"/>  
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="4"/>  
     
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="5"/>  
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="6"/>  
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="7"/>  
     
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="8"/>  
    <Button  
    android:layout\_width="110dp"  
    android:layout\_height="100dp"  
    android:text="9"/>  
     
   </GridLayout>

mainActvity.kt:

package com.rohit.grid\_layout  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 }  
}

output:



**PRACTICAL 5**

**Programming UI elements**

**Design App With UI:**

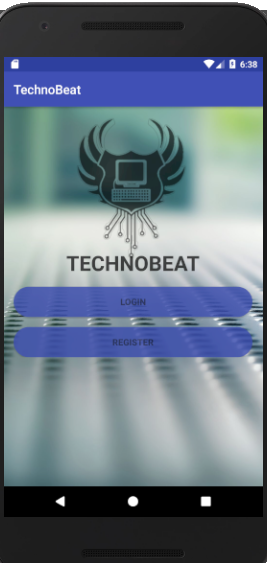
mainActivity.kt:

package rohit.technobeat  
  
import android.content.Intent  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import kotlinx.android.synthetic.main.activity\_login.\*  
import kotlinx.android.synthetic.main.activity\_main.\*  
import kotlinx.android.synthetic.main.activity\_register.\*  
import rohit.technobeat.R.id.login  
import rohit.technobeat.R.id.newaccount  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.activity\_main)  
 login.setOnClickListener **{** val intent = Intent(this, LoginActivity::class.java)  
 // start your next activity  
 startActivity(intent)  
 **}** newaccount.setOnClickListener **{** val intent = Intent(this, RegisterActivity::class.java)  
 // start your next activity  
 startActivity(intent)  
 **}** }  
}

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:gravity="center\_horizontal"  
 android:orientation="vertical"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 android:background="@drawable/home"  
 tools:context=".MainActivity">  
  
 <ScrollView  
 android:id="@+id/login\_form"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:gravity="center">  
  
  
 <android.support.v7.widget.AppCompatTextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="210dp"  
 android:alpha="0.7"  
 android:text="TECHNOBEAT"  
 android:textColor="#000000"  
 android:textSize="33dp"  
 android:textStyle="bold"  
 tools:layout\_marginLeft="85dp" />  
  
  
 <Button  
 android:id="@+id/login"  
 style="?android:textAppearanceSmall"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="16dp"  
 android:text="Login"  
 android:background="@drawable/round\_button"  
 android:alpha="0.8"  
 android:textStyle="bold" />  
 <Button  
 android:id="@+id/newaccount"  
 style="?android:textAppearanceSmall"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="16dp"  
 android:text="REGISTER"  
 android:background="@drawable/round\_button"  
 android:alpha="0.8"  
 android:textStyle="bold" />  
  
 </LinearLayout>  
 </ScrollView>  
</LinearLayout>

Output:



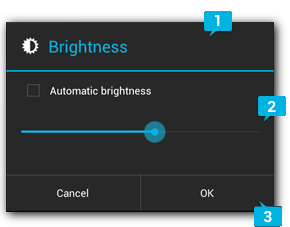
**PRACTICAL 6**

**Programming menus, dialog, dialog fragments**

**Alert:**

val *alertDialog*: AlertDialog? = activity?.let **{** val builder = AlertDialog.Builder(it)  
 builder.apply **{** setPositiveButton(R.*string*.*ok*,  
 DialogInterface.OnClickListener **{** dialog, id **->** // User clicked OK button  
 **}**)  
 setNegativeButton(R.*string*.*cancel*,  
 DialogInterface.OnClickListener **{** dialog, id **->** // User cancelled the dialog  
 **}**)  
 **}** // Set other dialog properties  
 ...  
  
 // Create the AlertDialog  
 builder.create()  
**}**

**output:**

****

**Menu:**

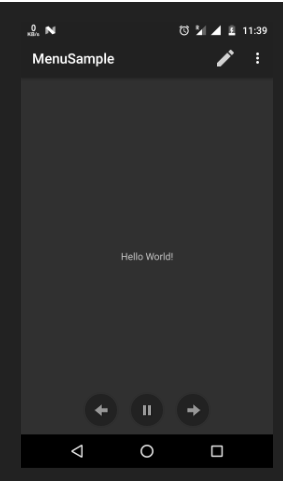
menu.xml:

<?xml version=”1.0″ encoding=”utf-8″?>  
<menu xmlns:android=”http://schemas.android.com/apk/res/android&#8221;  
xmlns:app=”http://schemas.android.com/apk/res-auto”&gt;  
  
<item  
android:id=”@+id/menu\_1″  
android:icon=”@drawable/ic\_menu\_1″  
android:title=”Menu 1″  
app:showAsAction=”always” />  
  
<item  
android:id=”@+id/menu\_2″  
android:icon=”@drawable/ic\_menu\_2″  
android:title=”Menu 2″ />  
  
<item  
android:id=”@+id/menu\_3″  
android:icon=”@drawable/ic\_menu\_3″  
android:title=”Menu 3″ />  
  
<item  
android:id=”@+id/menu\_4″  
android:icon=”@drawable/ic\_menu\_4″  
android:title=”Menu 4″ />  
  
</menu>

MainActivity.kt:

package rohit.com  
  
import android.os.Bundle  
import android.support.v7.app.AppCompatActivity  
import android.view.Menu  
import android.view.MenuItem  
import android.widget.Toast  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.*layout*.*activity\_main*)  
 }  
  
 override fun onCreateOptionsMenu(menu: Menu): Boolean {  
 *menuInflater*.inflate(R.*menu*.*main*, menu)  
 return true  
 }  
  
 override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 when (item.*itemId*) {  
 R.*id*.*menu\_1* -> {  
 Toast.makeText(this, “Menu 1 is selected”, Toast.LENGTH\_SHORT).show()  
 return true  
 }  
 R.*id*.*menu\_2* -> {  
 Toast.makeText(this, “Menu 2 is selected”, Toast.LENGTH\_SHORT).show()  
 return true  
 }  
 R.*id*.*menu\_3* -> {  
 Toast.makeText(this, “Menu 3 is selected”, Toast.LENGTH\_SHORT).show()  
 return true  
 }  
 R.*id*.*menu\_4* -> {  
 Toast.makeText(this, “Menu 4 is selected”, Toast.LENGTH\_SHORT).show()  
 return true  
 }  
 else -> return super.onOptionsItemSelected(item)  
 }  
 }  
}

**Output:**

****

**Practical 7**

**Programs on Services, notification and broadcast receivers**

1. **Programs on Services:**

**Services are commands which are used by kotlin in functions to execute the task. They are :** **IntentService, onStartCommand(),onHandleIntent() etc.**

1. **notification and broadcast receivers:**

**Step 1. Create an android app, For creating an Android app with kotlin read this tutorial.**

**Step 2. Creating Broadcast Receiver Create and extend Subclass and BroadcastReceiver implement.onReceive(Context, Intent) where onReceive method each message is received as an Intent object parameter.**

**MyReceiver.kt:**

package `in`.eyehunt.androidbroadcasts  
  
import android.content.BroadcastReceiver  
import android.content.Context  
import android.content.Intent  
import android.widget.Toast  
  
class MyReceiver : BroadcastReceiver() {  
  
 override fun onReceive(context: Context, intent: Intent) {  
 // *TODO: This method is called when the BroadcastReceiver is receiving* // an Intent broadcast.  
 Toast.makeText(context, "Broadcast : Flight mode changed.",  
 Toast.*LENGTH\_LONG*).show()  
 }  
}

**Step 3. Declare a broadcast receiver in the manifest file add the element<receiver> in your app’s manifest. Here is code snap**

**AndroidManifest.xml:**

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
package="in.eyehunt.androidbroadcasts">  
  
<application  
android:allowBackup="true"  
android:icon="@mipmap/ic\_launcher"  
android:label="@string/app\_name"  
android:roundIcon="@mipmap/ic\_launcher\_round"  
android:supportsRtl="true"  
android:theme="@style/AppTheme">  
<activity android:name=".MainActivity">  
<intent-filter>  
<action android:name="android.intent.action.MAIN" />  
  
<category android:name="android.intent.category.LAUNCHER" />  
</intent-filter>  
</activity>  
  
<receiver  
android:name=".MyReceiver"  
android:enabled="true"  
android:exported="true">  
<intent-filter>  
<action android:name="android.intent.action.AIRPLANE\_MODE"/>  
</intent-filter>  
</receiver>  
</application>  
  
</manifest>

**Note: If the app is not running and broadcast receiver declared in AndroidManifest.xml, then the system will launch your app.**

**Step 4. MainActivity code, no needs to do anything**

**MainActivity.kt:**

package `in`.eyehunt.androidbroadcasts  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.*layout*.*activity\_main*)  
 }  
}

**Step 5. Add following code in main\_activity.xml**

**add <ImageView> and <TextView>widget layout file.**

**main\_activity.xml:**

<?xml version="1.0" encoding="utf-8"?>  
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
xmlns:app="http://schemas.android.com/apk/res-auto"  
xmlns:tools="http://schemas.android.com/tools"  
android:layout\_width="match\_parent"  
android:layout\_height="match\_parent"  
android:background="@color/colorPrimary"  
tools:context="in.eyehunt.androidbroadcasts.MainActivity">  
  
<ImageView  
android:id="@+id/imageView"  
android:layout\_width="40dp"  
android:layout\_height="40dp"  
android:layout\_margin="8dp"  
android:layout\_marginTop="16dp"  
app:layout\_constraintStart\_toStartOf="parent"  
app:layout\_constraintTop\_toTopOf="parent"  
app:srcCompat="@mipmap/baseline\_airplanemode\_active\_white\_24" />  
  
<TextView  
android:id="@+id/textView"  
android:layout\_width="300dp"  
android:layout\_height="36dp"  
android:layout\_marginEnd="8dp"  
android:layout\_marginStart="8dp"  
android:gravity="center\_vertical"  
android:text="Flight Mode"  
android:textColor="@color/colorWhite"  
android:textSize="24dp"  
app:layout\_constraintEnd\_toEndOf="parent"  
app:layout\_constraintStart\_toEndOf="@+id/imageView"  
app:layout\_constraintTop\_toTopOf="@+id/imageView" />  
</android.support.constraint.ConstraintLayout>

**Output:**

****

**PRACTICAL 8**

**Database Programming with SQLite**

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="wrap\_content"  
android:orientation="vertical"  
android:gravity="center"  
tools:context="com.tutorialkart.sqlitetutorial.MainActivity">  
  
<TextView  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:text="SQLite Tutorial - User Management"  
android:textSize="20dp"  
android:padding="10dp" />  
  
<LinearLayout  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content"  
android:orientation="vertical">  
<EditText  
android:id="@+id/edittext\_userid"  
android:hint="User ID"  
android:gravity="center"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content" />  
<EditText  
android:id="@+id/edittext\_name"  
android:hint="User Name"  
android:gravity="center"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content" />  
<EditText  
android:id="@+id/edittext\_age"  
android:hint="User Age"  
android:gravity="center"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content" />  
</LinearLayout>  
  
<LinearLayout  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content"  
android:orientation="horizontal">  
<Button  
android:id="@+id/button\_add\_user"  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:layout\_weight="1"  
android:onClick="addUser"  
android:text="Add" />  
  
<Button  
android:id="@+id/button\_delete\_user"  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:layout\_weight="1"  
android:onClick="deleteUser"  
android:text="Delete" />  
  
<Button  
android:id="@+id/button\_show\_all"  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:layout\_weight="1"  
android:onClick="showAllUsers"  
android:text="Show All" />  
</LinearLayout>  
<TextView  
android:id="@+id/textview\_result"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content" />  
<LinearLayout  
android:id="@+id/ll\_entries"  
android:padding="15dp"  
android:orientation="vertical"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content"></LinearLayout>  
</LinearLayout>

UserModel.kt:

package com.tutorialkart.sqlitetutorial  
  
class UserModel(val userid: String, val name: String, val age: String)

DBContract.kt

package com.tutorialkart.sqlitetutorial  
  
import android.provider.BaseColumns  
  
object DBContract {  
  
 /\* Inner class that defines the table contents \*/  
 class UserEntry : BaseColumns {  
 companion object {  
 val TABLE\_NAME = "users"  
 val COLUMN\_USER\_ID = "userid"  
 val COLUMN\_NAME = "name"  
 val COLUMN\_AGE = "age"  
 }  
 }  
}

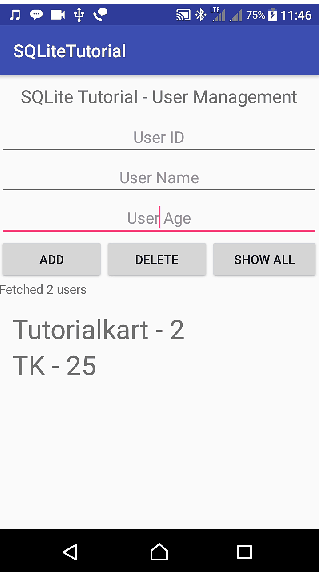
UserDBHelper.kt:

package com.tutorialkart.sqlitetutorial  
  
import android.content.ContentValues  
import android.content.Context  
import android.database.Cursor  
import android.database.sqlite.SQLiteConstraintException  
import android.database.sqlite.SQLiteDatabase  
import android.database.sqlite.SQLiteException  
import android.database.sqlite.SQLiteOpenHelper  
  
import java.util.ArrayList  
  
class UsersDBHelper(context: Context) : SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {  
 override fun onCreate(db: SQLiteDatabase) {  
 db.execSQL(SQL\_CREATE\_ENTRIES)  
 }  
  
 override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {  
 // This database is only a cache for online data, so its upgrade policy is  
 // to simply to discard the data and start over  
 db.execSQL(SQL\_DELETE\_ENTRIES)  
 onCreate(db)  
 }  
  
 override fun onDowngrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {  
 onUpgrade(db, oldVersion, newVersion)  
 }  
  
 @Throws(SQLiteConstraintException::class)  
 fun insertUser(user: UserModel): Boolean {  
 // Gets the data repository in write mode  
 val db = *writableDatabase* // Create a new map of values, where column names are the keys  
 val values = ContentValues()  
 values.put(DBContract.*UserEntry*.*COLUMN\_USER\_ID*, user.*userid*)  
 values.put(DBContract.UserEntry.COLUMN\_NAME, user.name)  
 values.put(DBContract.UserEntry.COLUMN\_AGE, user.age)  
  
 // Insert the new row, returning the primary key value of the new row  
 val newRowId = db.insert(DBContract.UserEntry.TABLE\_NAME, null, values)  
  
 return true  
 }  
  
 @Throws(SQLiteConstraintException::class)  
 fun deleteUser(userid: String): Boolean {  
 // Gets the data repository in write mode  
 val db = writableDatabase  
 // Define 'where' part of query.  
 val selection = DBContract.UserEntry.COLUMN\_USER\_ID + " LIKE ?"  
 // Specify arguments in placeholder order.  
 val selectionArgs = arrayOf(userid)  
 // Issue SQL statement.  
 db.delete(DBContract.UserEntry.TABLE\_NAME, selection, selectionArgs)  
  
 return true  
 }  
  
 fun readUser(userid: String): ArrayList<UserModel> {  
 val users = ArrayList<UserModel>()  
 val db = writableDatabase  
 var cursor: Cursor? = null  
 try {  
 cursor = db.rawQuery("select \* from " + DBContract.UserEntry.TABLE\_NAME + " WHERE " + DBContract.UserEntry.COLUMN\_USER\_ID + "='" + userid + "'", null)  
 } catch (e: SQLiteException) {  
 // if table not yet present, create it  
 db.execSQL(SQL\_CREATE\_ENTRIES)  
 return ArrayList()  
 }  
  
 var name: String  
 var age: String  
 if (cursor!!.moveToFirst()) {  
 while (cursor.isAfterLast == false) {  
 name = cursor.getString(cursor.getColumnIndex(DBContract.UserEntry.COLUMN\_NAME))  
 age = cursor.getString(cursor.getColumnIndex(DBContract.UserEntry.COLUMN\_AGE))  
  
 users.add(UserModel(userid, name, age))  
 cursor.moveToNext()  
 }  
 }  
 return users  
 }  
  
 fun readAllUsers(): ArrayList<UserModel> {  
 val users = ArrayList<UserModel>()  
 val db = writableDatabase  
 var cursor: Cursor? = null  
 try {  
 cursor = db.rawQuery("select \* from " + DBContract.UserEntry.TABLE\_NAME, null)  
 } catch (e: SQLiteException) {  
 db.execSQL(SQL\_CREATE\_ENTRIES)  
 return ArrayList()  
 }  
  
 var userid: String  
 var name: String  
 var age: String  
 if (cursor!!.moveToFirst()) {  
 while (cursor.*isAfterLast* == false) {  
 userid = cursor.getString(cursor.getColumnIndex(DBContract.*UserEntry*.*COLUMN\_USER\_ID*))  
 name = cursor.getString(cursor.getColumnIndex(DBContract.*UserEntry*.*COLUMN\_NAME*))  
 age = cursor.getString(cursor.getColumnIndex(DBContract.*UserEntry*.*COLUMN\_AGE*))  
  
 users.add(UserModel(userid, name, age))  
 cursor.moveToNext()  
 }  
 }  
 return users  
 }  
  
 companion object {  
 // If you change the database schema, you must increment the database version.  
 val DATABASE\_VERSION = 1  
 val DATABASE\_NAME = "FeedReader.db"  
  
 private val SQL\_CREATE\_ENTRIES =  
 "CREATE TABLE " + DBContract.*UserEntry*.*TABLE\_NAME* + " (" +  
 DBContract.*UserEntry*.*COLUMN\_USER\_ID* + " TEXT PRIMARY KEY," +  
 DBContract.*UserEntry*.*COLUMN\_NAME* + " TEXT," +  
 DBContract.*UserEntry*.*COLUMN\_AGE* + " TEXT)"  
  
 private val SQL\_DELETE\_ENTRIES = "DROP TABLE IF EXISTS " + DBContract.*UserEntry*.*TABLE\_NAME* }  
  
}

MainActivity.kt:

package com.tutorialkart.sqlitetutorial  
  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import android.view.View  
import android.widget.TextView  
import kotlinx.android.synthetic.main.activity\_main.\*  
  
class MainActivity : AppCompatActivity() {  
  
 lateinit var usersDBHelper : UsersDBHelper  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.activity\_main)  
  
 usersDBHelper = UsersDBHelper(this)  
 }  
  
 fun addUser(v:View){  
 var userid = this.edittext\_userid.text.toString()  
 var name = this.edittext\_name.text.toString()  
 var age = this.edittext\_age.text.toString()  
 var result = usersDBHelper.insertUser(UserModel(userid = userid,name = name,age = age))  
 //clear all edittext s  
 this.edittext\_age.setText("")  
 this.edittext\_name.setText("")  
 this.edittext\_userid.setText("")  
 this.textview\_result.*text* = "Added user : "+result  
 this.ll\_entries.removeAllViews()  
 }  
  
 fun deleteUser(v:View){  
 var userid = this.edittext\_userid.text.toString()  
 val result = usersDBHelper.deleteUser(userid)  
 this.textview\_result.text = "Deleted user : "+result  
 this.ll\_entries.removeAllViews()  
 }  
  
 fun showAllUsers(v:View){  
 var users = usersDBHelper.readAllUsers()  
 this.ll\_entries.removeAllViews()  
 users.forEach **{** var tv\_user = TextView(this)  
 tv\_user.textSize = 30F  
 tv\_user.text = it.name.toString() + " - " + it.age.toString()  
 this.ll\_entries.addView(tv\_user)  
 **}** this.textview\_result.text = "Fetched " + users.size + " users"  
 }  
}

output:



**PRACTICAL 9**

**Programming Security and permissions**

**Extra Packages requied in ManagePermission.kt (Class File)**

**import** android.app.Activity  
**import** android.content.pm.PackageManager  
**import** android.support.v4.app.ActivityCompat  
**import** android.support.v4.content.ContextCompat  
**import** android.support.v7.app.AlertDialog

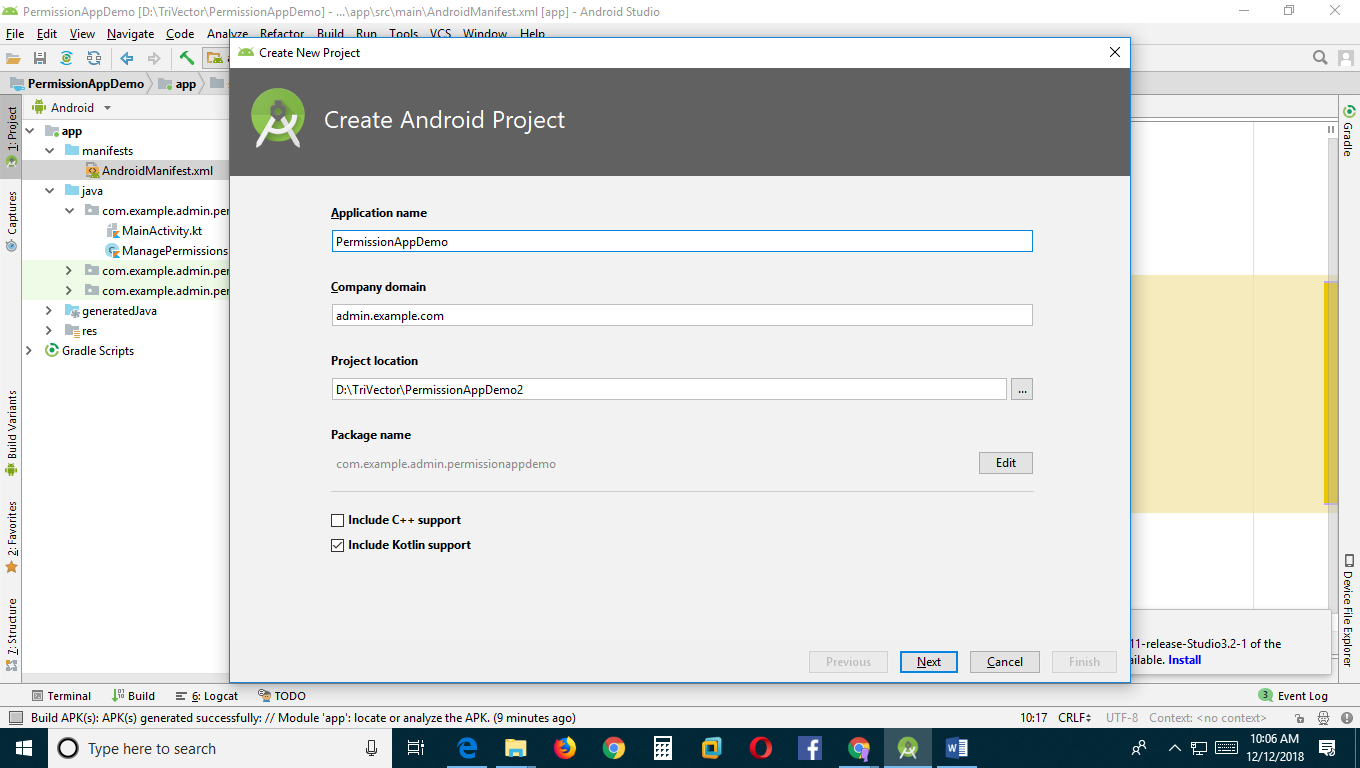
**Extra Packages requied in MainActivity.kt**

**import** android.Manifest  
**import** android.content.Context  
**import** android.os.Build  
**import** android.widget.Toast  
**import** kotlinx.android.synthetic.main.activity\_main.\*

**For Multple Permission Access,need to add following line in class MainActivity**

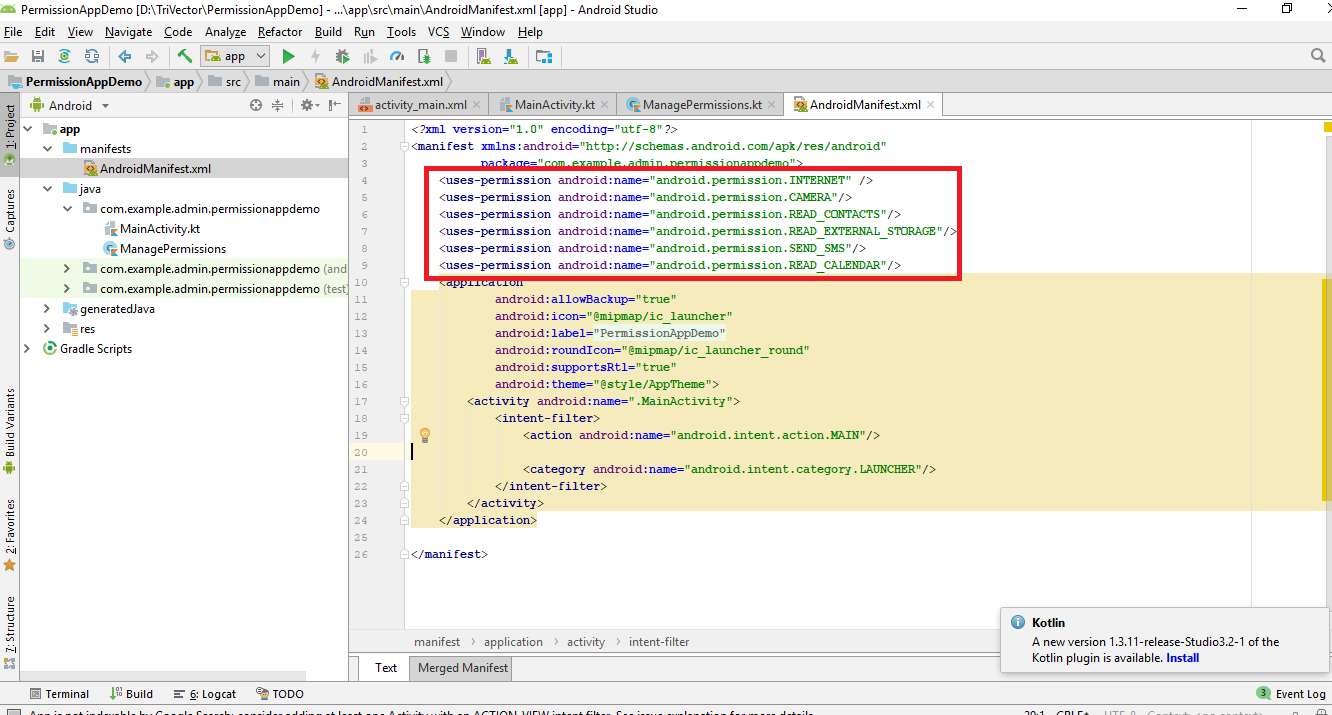
**private val PermissionsRequestCode = 123**

1. **Create a new project in android studio**



1. **An app must publicize the permissions it requires by including [<uses-permission>](https://developer.android.com/guide/topics/manifest/uses-permission-element.html) tags in the [app manifest](https://developer.android.com/guide/topics/manifest/manifest-intro.html).**

<**uses-permission android:name="android.permission.INTERNET"** />  
<**uses-permission android:name="android.permission.CAMERA"**/>  
<**uses-permission android:name="android.permission.READ\_CONTACTS"**/>  
<**uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"**/>  
<**uses-permission android:name="android.permission.SEND\_SMS"**/>  
<**uses-permission android:name="android.permission.READ\_CALENDAR"**/>

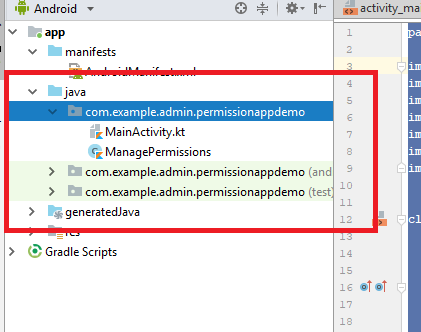


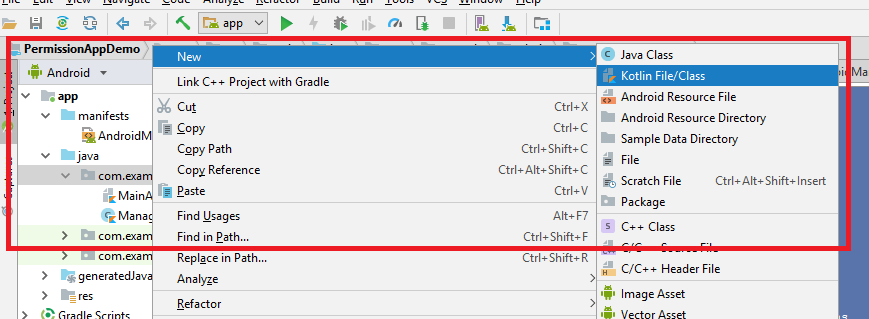
1. **MainActivity.kt**

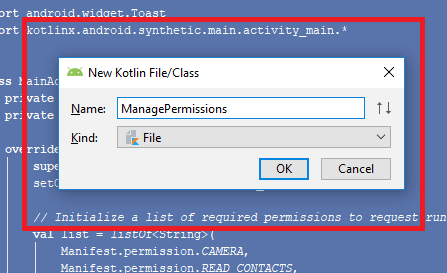
**package** com.example.admin.permissionappdemo  
  
**import** android.Manifest  
**import** android.content.Context  
**import** android.os.Build  
**import** android.support.v7.app.AppCompatActivity  
**import** android.os.Bundle  
**import** android.widget.Toast  
**import** kotlinx.android.synthetic.main.activity\_main.\*  
  
  
**class** MainActivity : AppCompatActivity() {  
 **private val PermissionsRequestCode** = 123  
 **private lateinit var managePermissions**: ManagePermissions  
  
 **override fun** onCreate(savedInstanceState: Bundle?) {  
 **super**.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 *// Initialize a list of required permissions to request runtime* **val** list = *listOf*<String>(  
 Manifest.permission.*CAMERA*,  
 Manifest.permission.*READ\_CONTACTS*,  
 Manifest.permission.*READ\_EXTERNAL\_STORAGE*,  
 Manifest.permission.*SEND\_SMS*,  
 Manifest.permission.*READ\_CALENDAR* )  
  
 *// Initialize a new instance of ManagePermissions class* **managePermissions** = ManagePermissions(**this**,list,**PermissionsRequestCode**)  
  
 *// Button to check permissions states* button.setOnClickListener**{  
 if** (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*)  
 **managePermissions**.checkPermissions()  
 **}** }  
  
  
 *// Receive the permissions request result* **override fun** onRequestPermissionsResult(requestCode: Int, permissions: Array<String>,  
 grantResults: IntArray) {  
 **when** (requestCode) {  
 **PermissionsRequestCode** ->{  
 **val** isPermissionsGranted = **managePermissions** .processPermissionsResult(requestCode,permissions,grantResults)  
  
 **if**(isPermissionsGranted){  
 *// Do the task now  
 toast*(**"Permissions granted."**)  
 }**else**{  
 *toast*(**"Permissions denied."**)  
 }  
 **return** }  
 }  
 }  
}  
  
  
*// Extension function to show toast message***fun** Context.toast(message: String) {  
 Toast.makeText(**this**, message, Toast.*LENGTH\_SHORT*).show()  
}

1. **Create a New Kotlin Class**

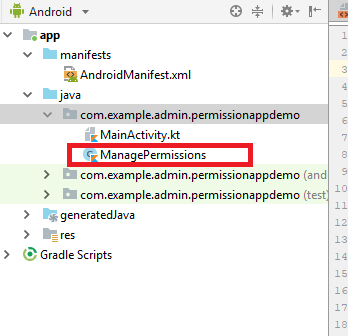
**app->src->main->java->com.example.admin.permissionappdemo**

****

****

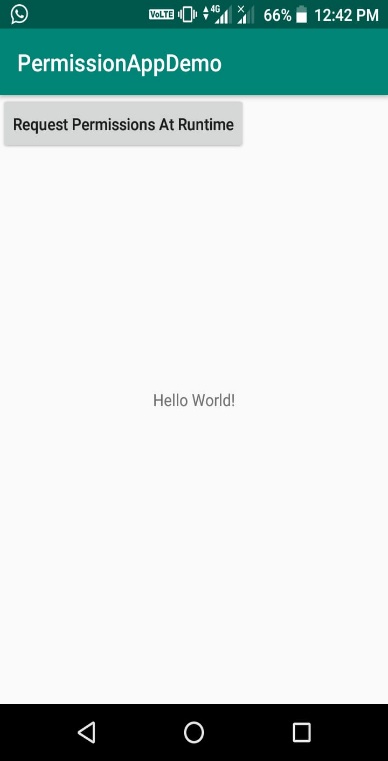
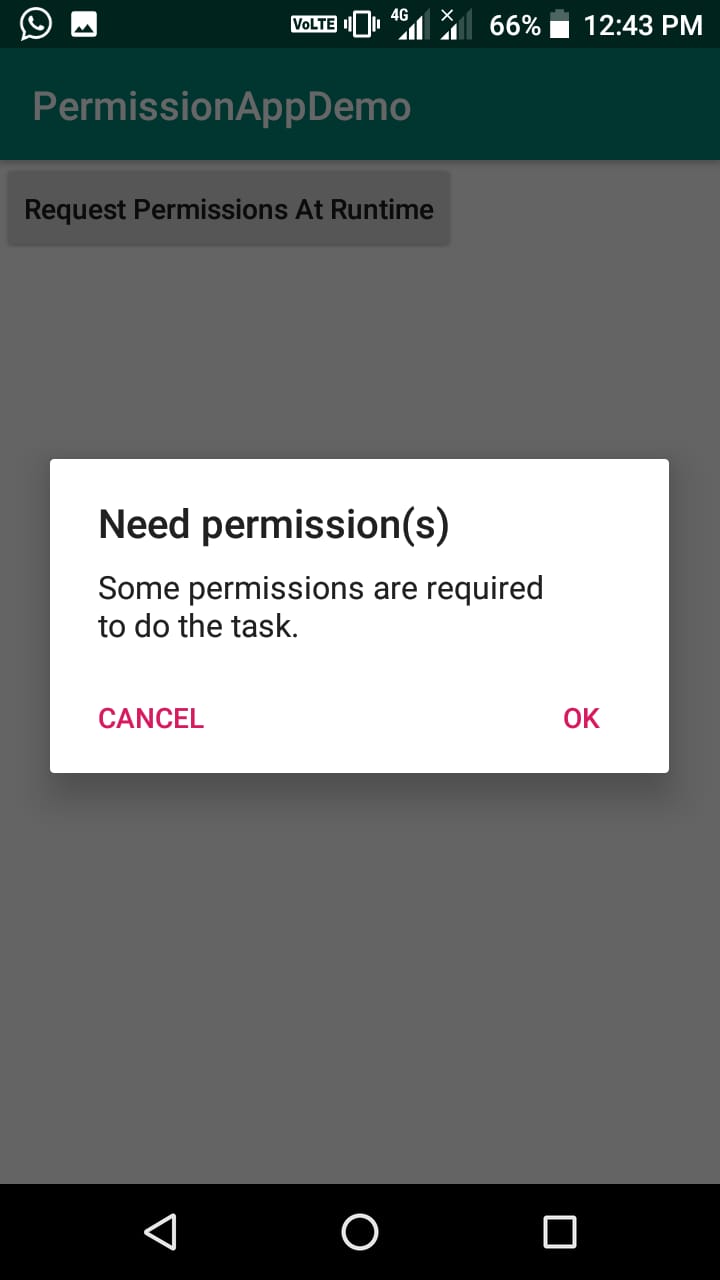
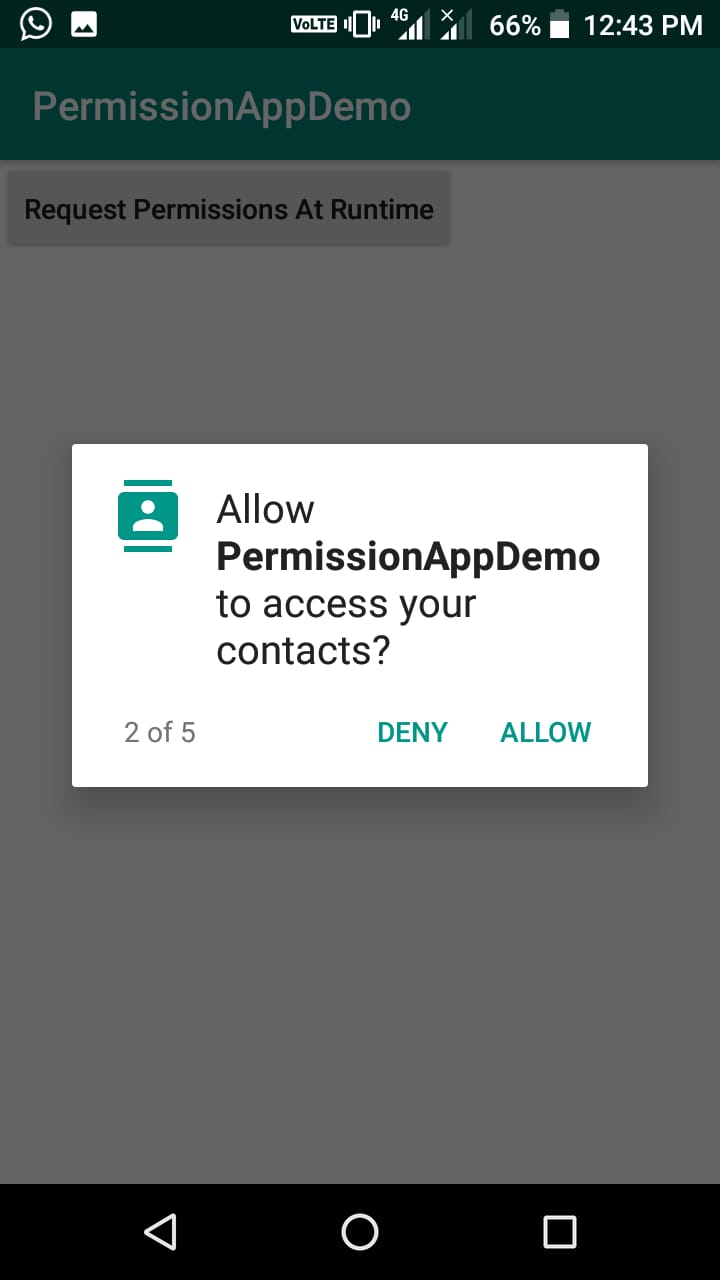
****

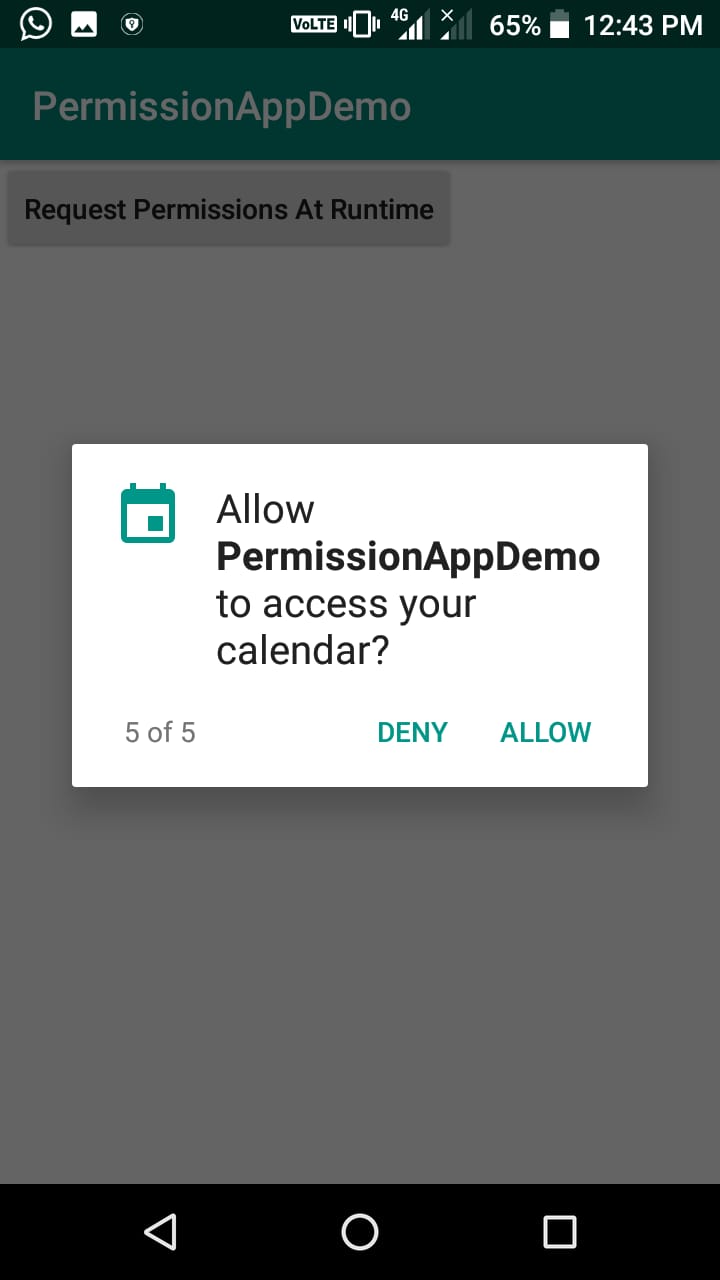
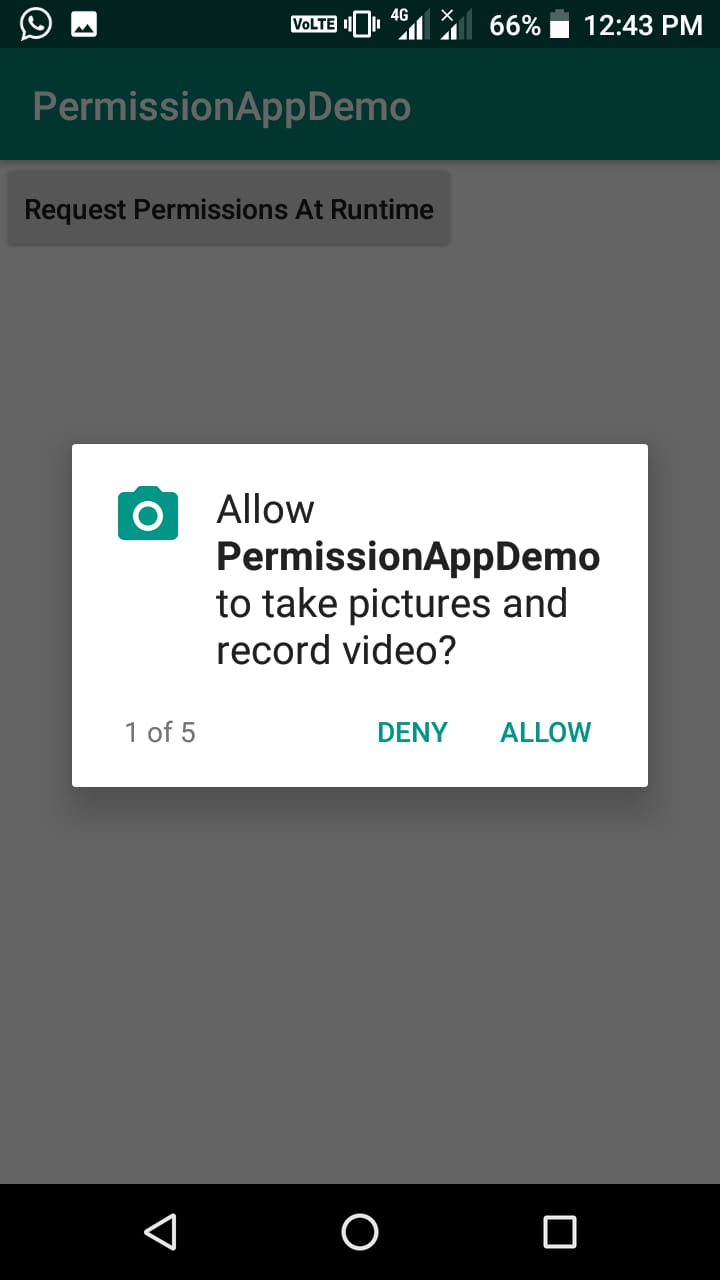
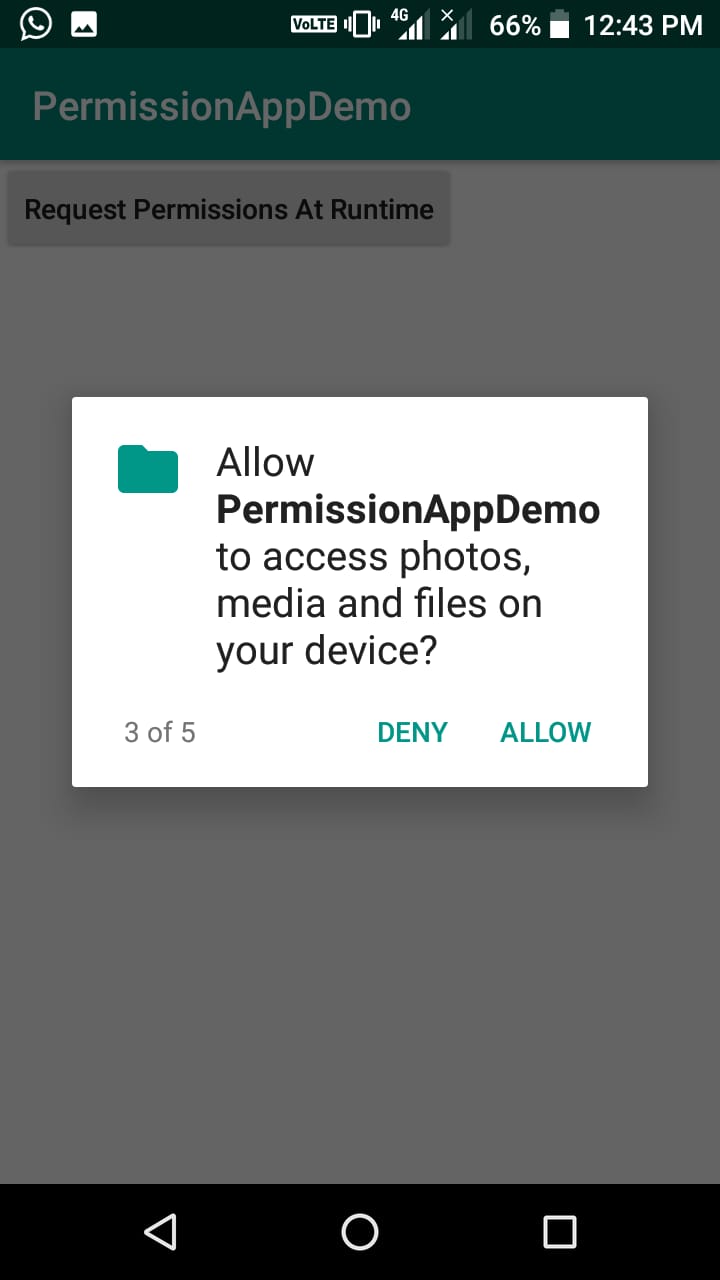
**Class file is generated**

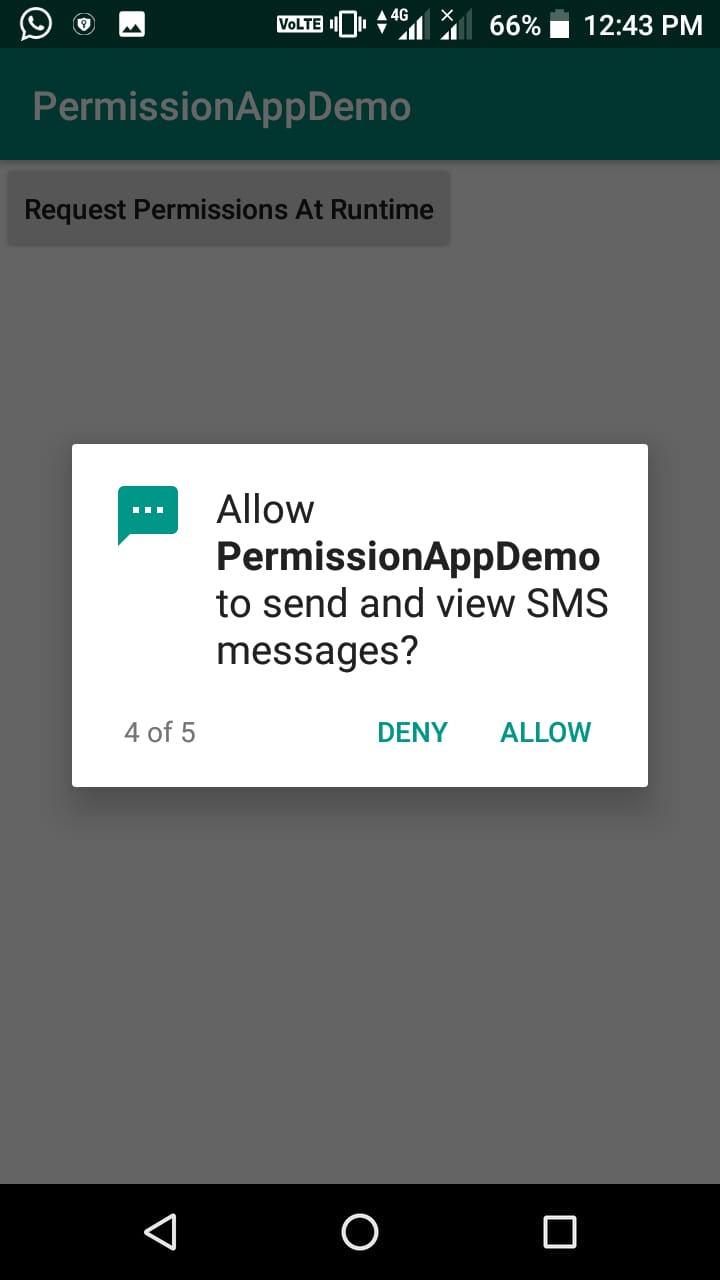
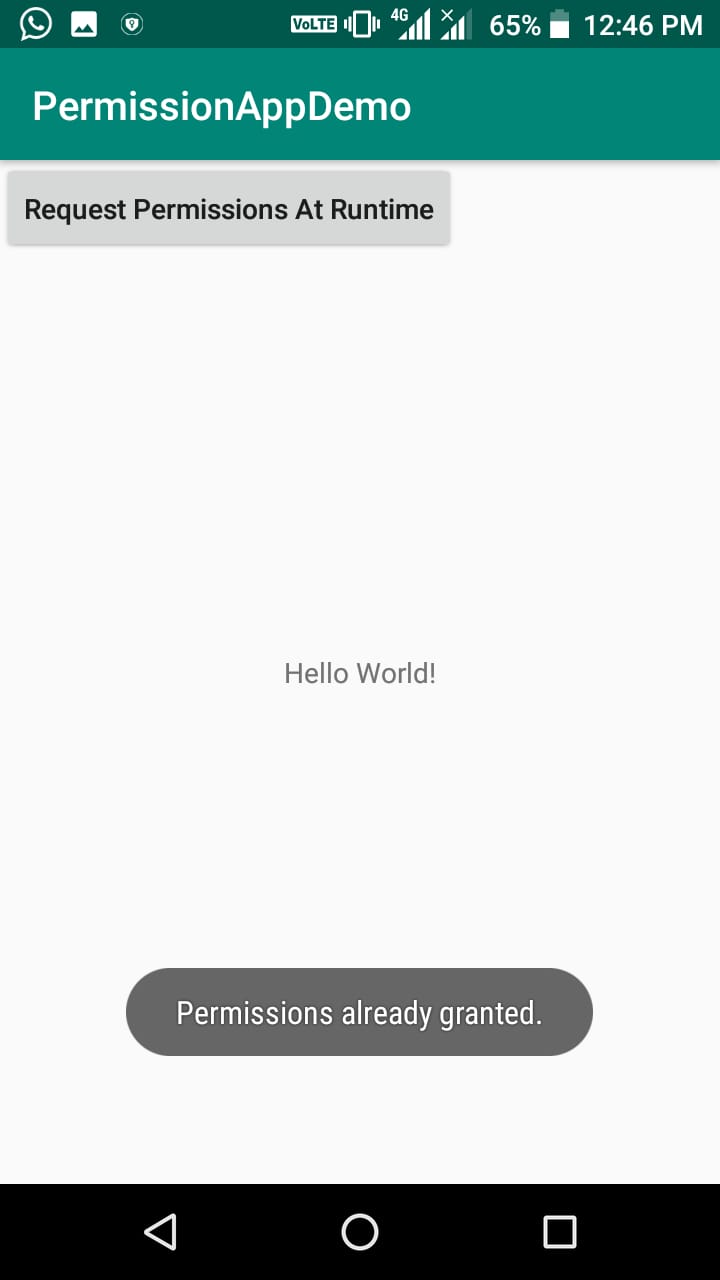
****

1. **Write the following code in the Class File**

**import** android.app.Activity  
**import** android.content.pm.PackageManager  
**import** android.support.v4.app.ActivityCompat  
**import** android.support.v4.content.ContextCompat  
**import** android.support.v7.app.AlertDialog  
  
  
**class** ManagePermissions(**val activity**: Activity,**val list**: List<String>,**val code**:Int) {  
  
 *// Check permissions at runtime* **fun** checkPermissions() {  
 **if** (isPermissionsGranted() != PackageManager.*PERMISSION\_GRANTED*) {  
 showAlert()  
 } **else** {  
 **activity**.*toast*(**"Permissions already granted."**)  
 }  
 }  
  
  
 *// Check permissions status* **private fun** isPermissionsGranted(): Int {  
 *// PERMISSION\_GRANTED : Constant Value: 0  
 // PERMISSION\_DENIED : Constant Value: -1* **var** counter = 0;  
 **for** (permission **in list**) {  
 counter += ContextCompat.checkSelfPermission(**activity**, permission)  
 }  
 **return** counter  
 }  
  
  
 *// Find the first denied permission* **private fun** deniedPermission(): String {  
 **for** (permission **in list**) {  
 **if** (ContextCompat.checkSelfPermission(**activity**, permission)  
 == PackageManager.*PERMISSION\_DENIED*) **return** permission  
 }  
 **return ""** }  
  
  
 *// Show alert dialog to request permissions* **private fun** showAlert() {  
 **val** builder = AlertDialog.Builder(**activity**)  
 builder.setTitle(**"Need permission(s)"**)  
 builder.setMessage(**"Some permissions are required to do the task."**)  
 builder.setPositiveButton(**"OK"**, **{** dialog, which **->** requestPermissions() **}**)  
 builder.setNeutralButton(**"Cancel"**, **null**)  
 **val** dialog = builder.create()  
 dialog.show()  
 }  
  
  
 *// Request the permissions at run time* **private fun** requestPermissions() {  
 **val** permission = deniedPermission()  
 **if** (ActivityCompat.shouldShowRequestPermissionRationale(**activity**, permission)) {  
 *// Show an explanation asynchronously* **activity**.*toast*(**"Should show an explanation."**)  
 } **else** {  
 ActivityCompat.requestPermissions(**activity**, **list**.*toTypedArray*(), **code**)  
 }  
 }  
  
  
 *// Process permissions result* **fun** processPermissionsResult(requestCode: Int, permissions: Array<String>,  
 grantResults: IntArray): Boolean {  
 **var** result = 0  
 **if** (grantResults.*isNotEmpty*()) {  
 **for** (item **in** grantResults) {  
 result += item  
 }  
 }  
 **if** (result == PackageManager.*PERMISSION\_GRANTED*) **return true  
 return false** }  
}

**  **

**  **

** **

**PRACTICAL 10**

**Programming Network Communications and Services (JSON)**

1. **Handling connectivity errors in Android apps with Kotlin:**

**Open your build.gradle file and add the following dependencies:**

**implementation 'com.android.support:design:27.1.1'**

**implementation 'com.squareup.retrofit2:retrofit:2.3.0'**

**implementation 'com.squareup.retrofit2:converter-scalars:2.3.0'**

**Open your AndroidManifest.xml file and add the permissions like so:**

<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
package="com.example.android.internetconnectivity">  
  
<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"/>  
<uses-permission android:name="android.permission.INTERNET"/>  
  
[...]  
  
</manifest>

**When there is a network connection, we will fetch data from an API. Let’s set up an interface to hold the endpoints we will access. Create a new Kotlin file named ApiService and paste this:**

import retrofit2.Call  
import retrofit2.http.GET  
  
interface ApiService {  
 @GET(".")  
 fun getFeeds(): Call<String>  
}

**For this demo, we are only going to access one endpoint, which is equivalent to our base URL. It’s for this reason we used a dot instead of the usual /some-url in the @GET annotation.**

**When these items are fetched, we will display the items in a list. We, therefore, need a RecyclerView in the layout and a matching adapter. Create a new Kotlin file named RecyclerAdapter and paste this:**

import android.support.v7.widget.RecyclerView  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.TextView  
  
class RecyclerAdapter : RecyclerView.Adapter<RecyclerAdapter.ViewHolder>() {  
  
 private var list = ArrayList<String>()  
  
 fun setItems(newList: ArrayList<String>){  
 this.list = newList  
 this.notifyDataSetChanged()  
 }  
  
 override fun getItemCount() = list.size  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {  
 val view = LayoutInflater.from(parent.context)  
 .inflate(android.R.layout.simple\_list\_item\_1, parent, false)  
  
 return ViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: ViewHolder, position: Int) {  
 holder.textView.text = list[position]  
 }  
  
 inner class ViewHolder(itemView: View?): RecyclerView.ViewHolder(itemView) {  
 var textView: TextView = itemView!!.findViewById(android.R.id.text1)  
 }  
  
}

**he adapter handles the display of items on a list. It has some overridden methods like:**

**getItemCount – to tell the size of the list to be populated.**

**onCreateViewHolder – used to choose a layout for a list row.**

**onBindViewHolder – to bind data to each row depending on the position, etc.**

**Next, we will update the layout of our MainActivity‘s activity\_main.xml file like so:**

<?xml version="1.0" encoding="utf-8"?>  
<android.support.constraint.ConstraintLayout  
xmlns:android="http://schemas.android.com/apk/res/android"  
xmlns:app="http://schemas.android.com/apk/res-auto"  
xmlns:tools="http://schemas.android.com/tools"  
android:layout\_width="match\_parent"  
android:layout\_height="match\_parent"  
tools:context=".MainActivity">  
  
<android.support.v7.widget.RecyclerView  
android:layout\_width="match\_parent"  
android:layout\_height="match\_parent"  
android:id="@+id/recyclerView"  
app:layout\_constraintBottom\_toBottomOf="parent"  
app:layout\_constraintLeft\_toLeftOf="parent"  
app:layout\_constraintRight\_toRightOf="parent"  
app:layout\_constraintTop\_toTopOf="parent" />  
  
<ImageView  
android:id="@+id/imageView"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content"  
android:src="@drawable/no\_internet\_connection" />  
  
</android.support.constraint.ConstraintLayout>

**The layout contains a RecyclerView for our list items and an ImageView to show an error message.**

**We also need an error message image. Once you have an image, rename the file to no\_internet\_connection and save it to your drawable folder: NameOfProject/app/src/main/res/drawable.**

**For us to monitor when the connectivity changes, we need broadcast receivers. Broadcast receivers are components that allow you to register and listen to Android system and application events. Usually, the Android system sends broadcast events when various system events occur and your app needs to register to get these events.**

**Let’s register a listener to be triggered when the internet connection is online or offline. Open your MainActivity file and paste the following code:**

import android.content.BroadcastReceiver  
import android.content.Context  
import android.content.Intent  
import android.content.IntentFilter  
import android.net.ConnectivityManager  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import android.support.v7.widget.LinearLayoutManager  
import android.util.Log  
import android.view.View  
import kotlinx.android.synthetic.main.activity\_main.\*  
import okhttp3.OkHttpClient  
import org.json.JSONObject  
import retrofit2.Call  
import retrofit2.Callback  
import retrofit2.Response  
import retrofit2.Retrofit  
import retrofit2.converter.scalars.ScalarsConverterFactory  
  
  
class MainActivity : AppCompatActivity() {  
  
 private val arrayList = ArrayList<String>()  
 private val adapter = RecyclerAdapter()  
 private val retrofit = Retrofit.Builder()  
 .baseUrl("https://api.reddit.com/")  
 .addConverterFactory(ScalarsConverterFactory.create())  
 .client(OkHttpClient.Builder().build())  
 .build()  
  
 private var broadcastReceiver: BroadcastReceiver = object : BroadcastReceiver() {  
 override fun onReceive(context: Context, intent: Intent) {  
 val notConnected = intent.getBooleanExtra(ConnectivityManager  
 .*EXTRA\_NO\_CONNECTIVITY*, false)  
 if (notConnected) {  
 disconnected()  
 } else {  
 connected()  
 }  
 }  
 }  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.*layout*.*activity\_main*)  
 setupRecyclerView()  
 }  
  
}

**Above, we initialized some variables:**

**arrayList – we will add fetched items to this list.**

**adapter – this is the instance of the adapter class.**

**retrofit – a Retrofit instance.**

**broadcastReciever – this instance implements the onRecieve callback. This callback method is called when the system has notified us of a change in the network connection. In the callback, we then check to know the connectivity status thereby calling either a private connected or disconnected function.**

**After creating the broadcast receiver, we have to register it to get updates and unregister if there are no more activities. To do this, add the following functions to the code above in the**

**MainActivity:**

override fun onStart() {  
 super.onStart()  
 registerReceiver(broadcastReceiver, IntentFilter(ConnectivityManager.*CONNECTIVITY\_ACTION*))  
}  
  
override fun onStop() {  
 super.onStop()  
 unregisterReceiver(broadcastReceiver)  
}

**In the onCreate function, we set up our RecyclerView by calling the setupRecyclerView. Create a private function in the MainActivity class and set it up like this:**

private fun setupRecyclerView(){  
 *with*(recyclerView)**{** *layoutManager* = LinearLayoutManager(this@MainActivity)  
 *adapter* = this@MainActivity.*adapter* **}**}

**Remember we mentioned the connected and disconnected functions earlier in this post. We will now add them to the class. Add them to the MainActivity file like so:**

private fun disconnected() {  
 recyclerView.*visibility* = View.*INVISIBLE* imageView.*visibility* = View.*VISIBLE*}  
  
private fun connected() {  
 recyclerView.*visibility* = View.*VISIBLE* imageView.*visibility* = View.*INVISIBLE* fetchFeeds()  
}

**The disconnected function is called when there is no network connection. It hides the RecyclerView and shows the ImageView. The connected function is called when there is an active internet connection. It shows the RecyclerView, hides the ImageView, and finally calls the fetchFeeds function.**

**Next, in the same file, paste the following code:**

private fun fetchFeeds() {  
 retrofit.create(ApiService::class.*java*)  
 .getFeeds()  
 .enqueue(object : Callback<String> {  
 override fun onFailure(call: Call<String>, t: Throwable) {  
 Log.e("MainActivityTag", t.message)  
 }  
  
 override fun onResponse(call: Call<String>?, response: Response<String>) {  
 addTitleToList(response.body()!!)  
 }  
  
 })  
}

**This function calls the API to get data. When the call is successful, we have another function that helps us add the title of the posts gotten from the endpoint to our list and then to our adapter. Create a function named addTitleToList and set it up like so:**

private fun addTitleToList(response: String) {  
 val jsonObject = JSONObject(response).getJSONObject("data")  
 val children = jsonObject.getJSONArray("children")  
  
 for (i in 0..(children.length()-1)) {  
 val item = children.getJSONObject(i).getJSONObject("data").getString("title")  
 arrayList.add(item)  
 adapter.setItems(arrayList)  
 }  
}