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:

Create New Project ✕

Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

☑ **Phone and Tablet**

API 15: Android 4.0.3 (IceCreamSandwich) ▼

By targeting **API 15 and later**, your app will run on approximately **100%** of devices. Help me choose

☐ Include Android Instant App support

☐ **Wear OS**

API 23: Android 6.0 (Marshmallow) ▼

☐ **TV**

API 21: Android 5.0 (Lollipop) ▼

☐ **Android Auto**

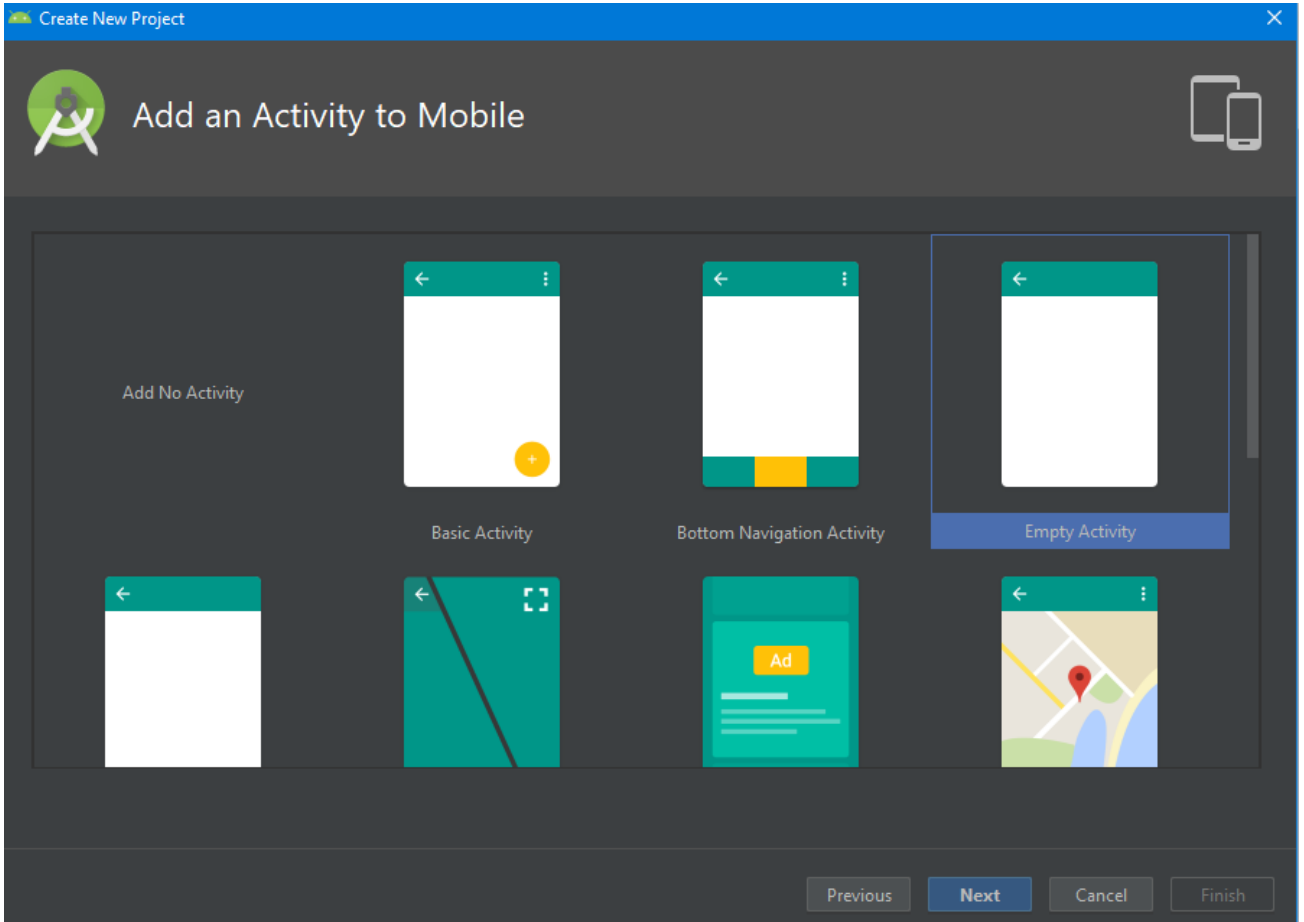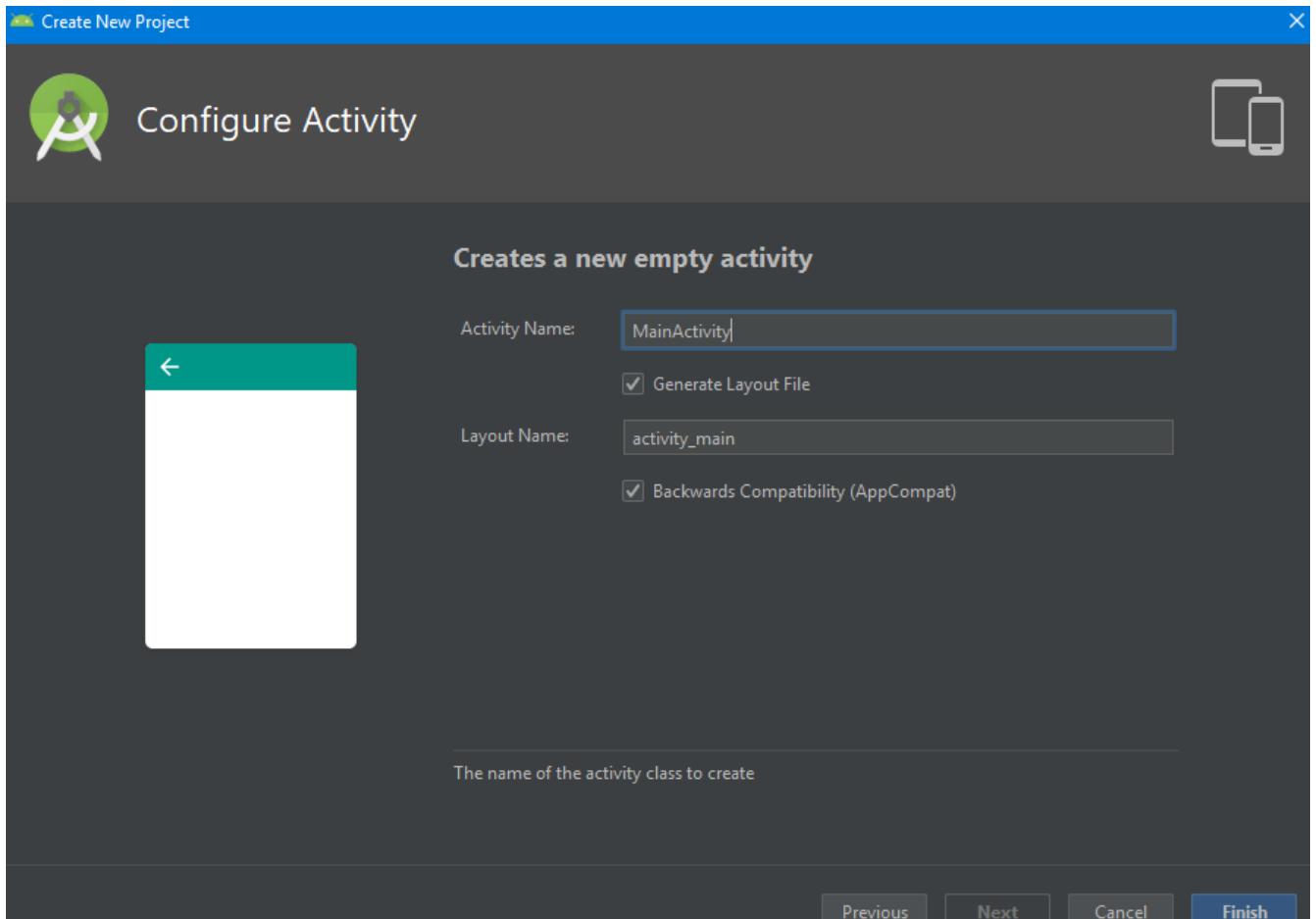☐ **Android Things**

API 24: Android 7.0 (Nougat) ▼

Previous   Next   Cancel   Finish

Create New Project

Add an Activity to Mobile

Add No Activity

Basic Activity

Bottom Navigation Activity

Empty Activity

Previous   Next   Cancel   Finish

Activity_Main.Kt

```kotlin
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
<?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:
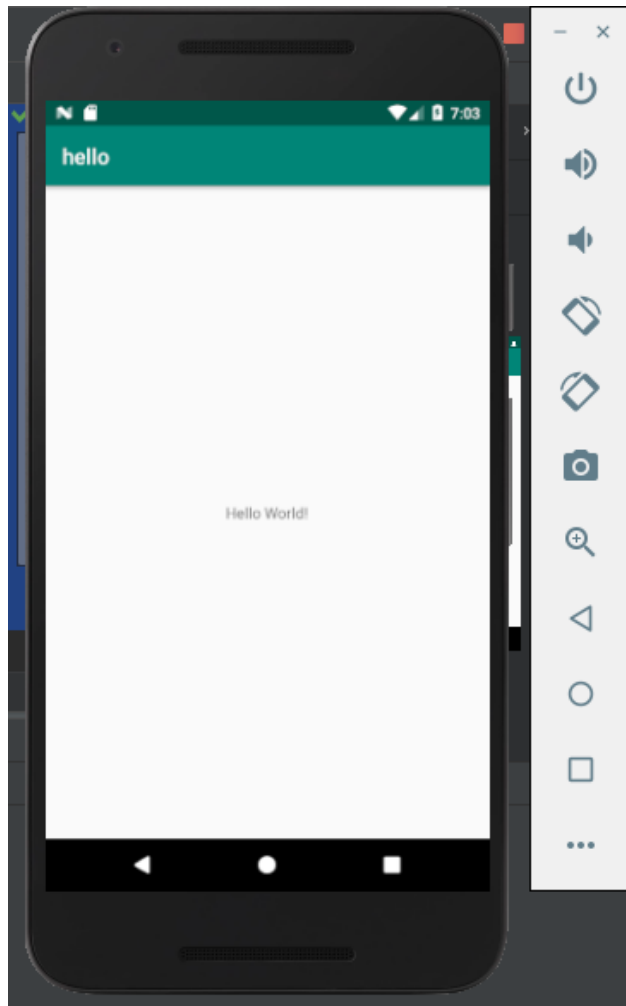
BroadcastActivity:

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:

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

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

    package="in.eyehunt.androidbroadcasts">

    <application
```

```xml
        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:

```kotlin
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
<?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"
```

```xml
            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.

Your Virtual Devices
Android Studio

Type	Name	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions ▼
🔲	10.1 WXGA (Tablet) API...		800 × 12...	23	Android 6.0...	x86	2 GB	▶ ✎ ▼
⌚	Android Wear Square A...	▷	280 × 2...	26	Android 8.0...	x86	650 ...	▶ ✎ ▼
🔲	Nexus 5X API 26	▷	1080 × 1...	26	Android 8.0...	x86	1 GB	▶ ✎ ▼

? + Create Virtual Device...

Select Hardware
Android Studio

Choose a device definition

Category	Name ▼	Play Store	Size	Resolution	Density
TV	Pixel XL		5.5"	1440x2...	560dpi
Wear	Pixel		5.0"	1080x1...	xxhdpi
Phone	Nexus S		4.0"	480x800	hdpi
Tablet	Nexus One		3.7"	480x800	hdpi
	Nexus 6P		5.7"	1440x2...	560dpi
	Nexus 6		5.96"	1440x2...	560dpi
	Nexus 5X	▷	5.2"	1080x1...	420dpi
	Nexus 5	▷	4.95"	1080x1...	xxhdpi
	Nexus 4		4.7"	768x1280	xhdpi

🔲 Nexus 5X

1080px

Size: large
Ratio: long
Density: 420dpi

5.2"   1920px

New Hardware Profile   Import Hardware Profiles

Clone Device...

? Cancel Previous Next Finish

System Image
Android Studio

Select a system image

Recommended | x86 Images | Other Images

Release Name	API Level ▼	ABI	Target
O	26	x86	Android 8.0 (Google Play)
Nougat	24	x86	Android 7.0 (Google Play)

O

API Level
26

Android
8.0
Google Inc.

System Image
x86

We recommend these Google Play images because this device is compatible with Google Play.

Questions on API level?
See the API level distribution chart

? Cancel Previous Next Finish

PRACTICAL 2

Programming Resources

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

Color:
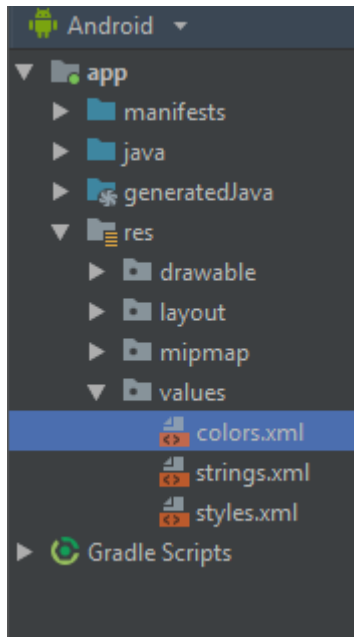
Color.xml

```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

```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>
```
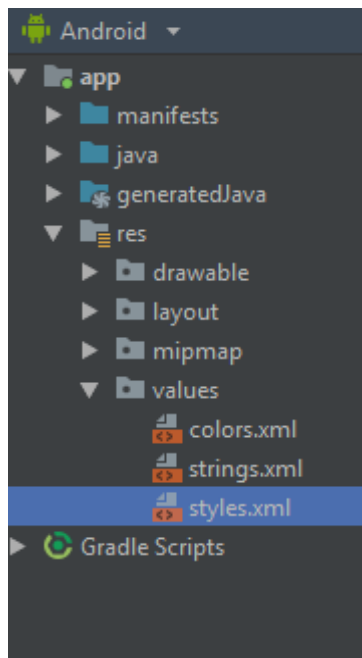
String:
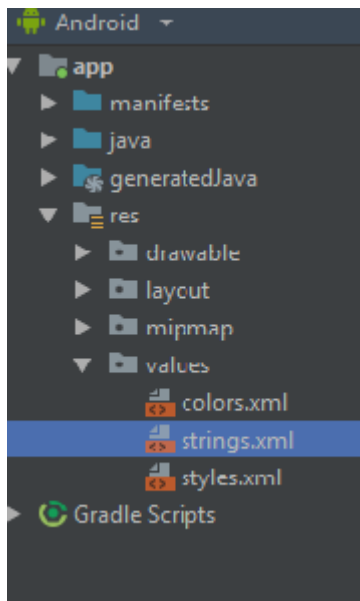
String.xml:

```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
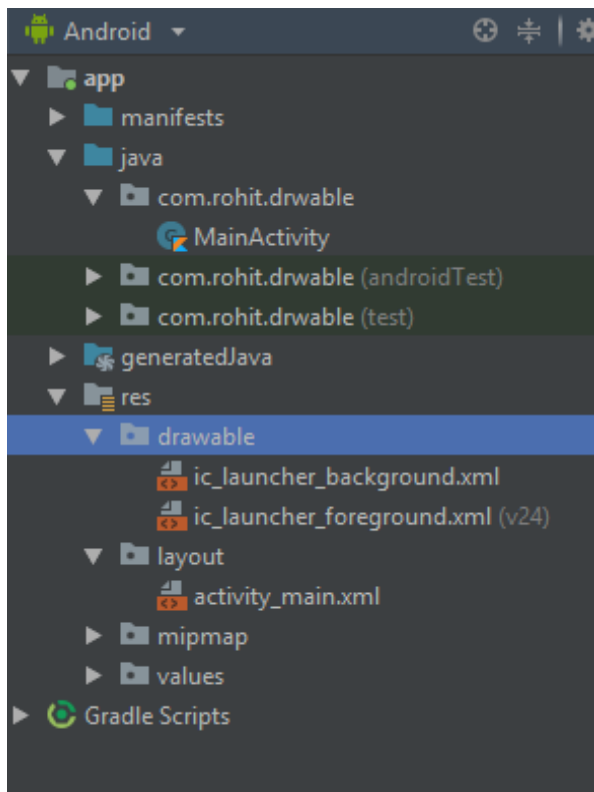
2. Copy the image if you want to create image drawable
3. Paste that image file inside the drawable folder
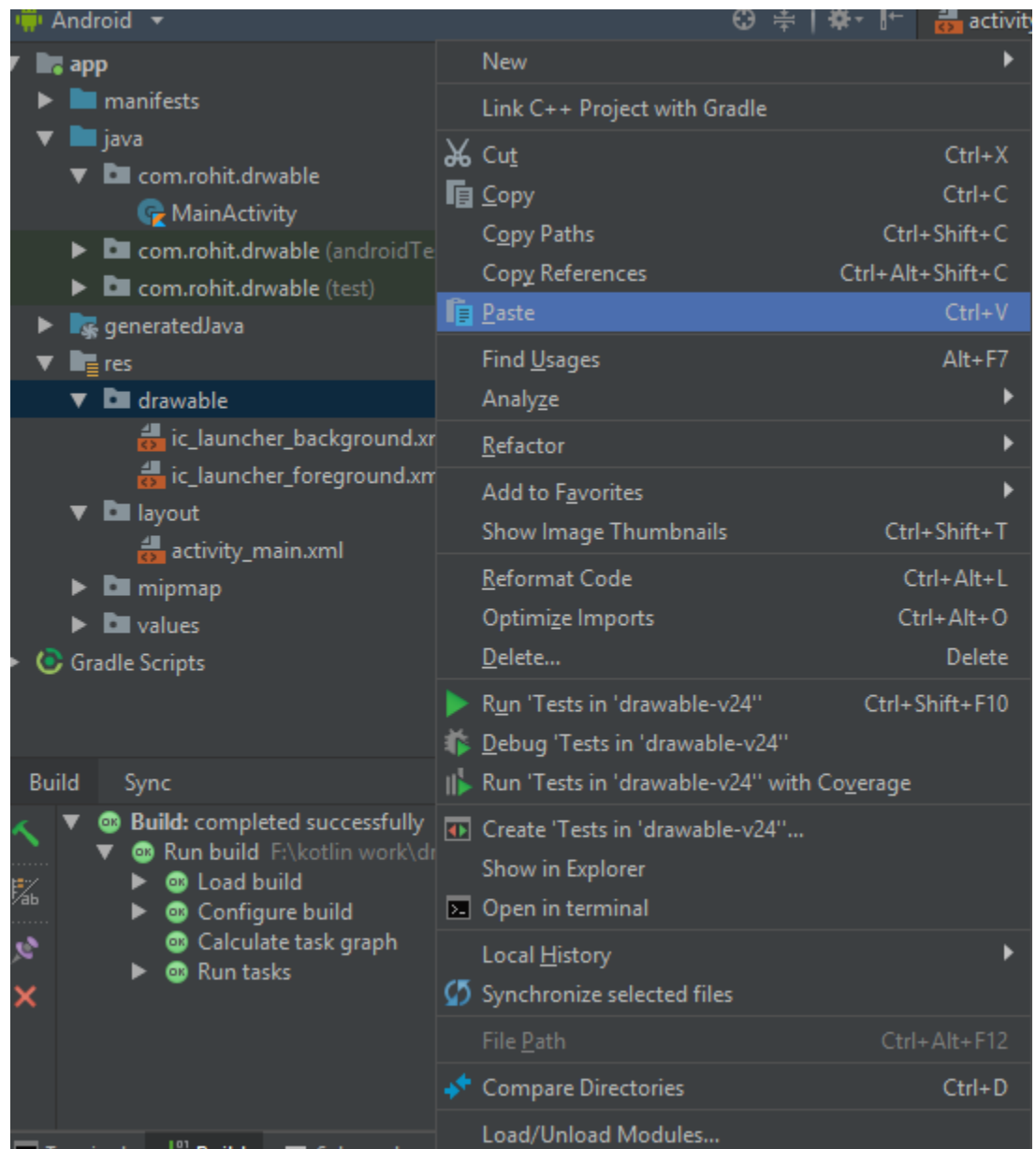
Note: to create drawable resource, right click on drawable folder and select drawable resource file.

Dimension, Image:

Main_Activity.kt:

```kotlin
package com.rohit.drwable

import android.support.v7.app.AppCompatActivity
import android.os.Bundle

class MainActivity : AppCompatActivity() {
```

```kotlin
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

activity_main.xml:

```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:
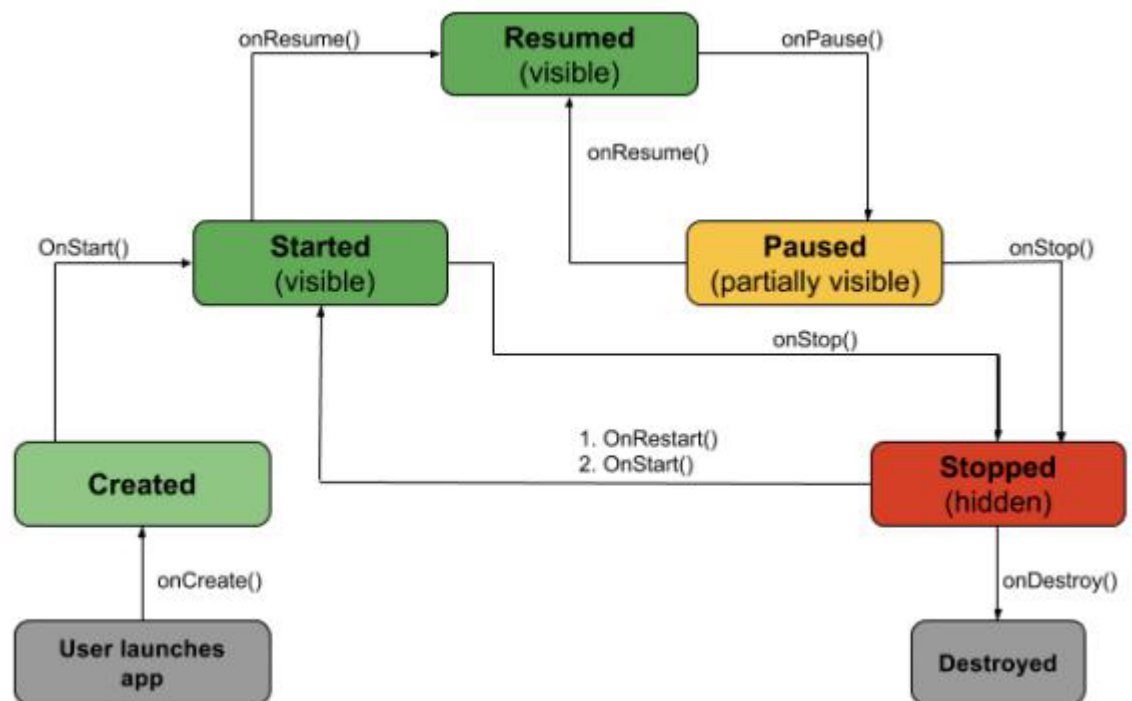
- **onCreate():** Called by the OS when the activity is first created. This is where you initialize any UI elements or data objects. You also have the savedInstanceState of the activity that contains its previously saved state, and you can use it to recreate that state.\

```
fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
```

- **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:

```kotlin
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
```

```kotlin
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")
}
```

Multiple Activities:

activity_first.xml code:

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

activity_second.xml code:

```xml
<?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"
```

```
</LinearLayout>
```

activity_third.xml code:

```xml
<?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 ........."
```

Activity_first.kt

```kotlin
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)

        }
```

PRACTICAL 4

Programs related to different Layouts

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

1. linear layout:

```xml
<?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>
```

2. Relative:

```xml
<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>
```

```xml
</RelativeLayout>
```

3. Table:

Activity_main.xml

```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"
```

```xml
                android:layout_gravity="center"
            />
        </TableRow>
    </TableLayout>

</LinearLayout>
```

Activity_main.kt

```kotlin
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:

4. Frame:

Activity_main.xml

```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>
```
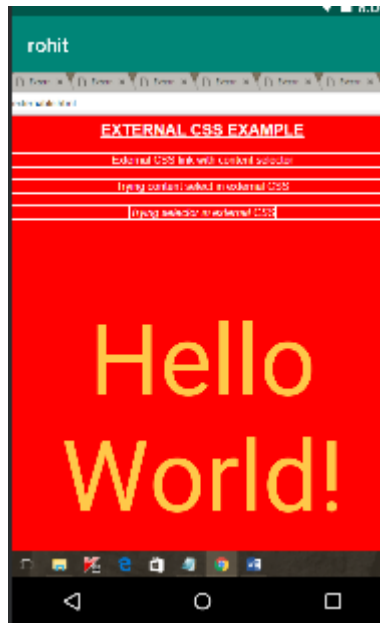
Activity_main.kt

```
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:

5. List View:

Activity_main.xml

```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

```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
<?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:

```kotlin
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

```kotlin
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:

6. Grid layout:

```xml
7. <?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"
```

```xml
            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:

```kotlin
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:

```kotlin
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
<?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"
```

```xml
        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>
```
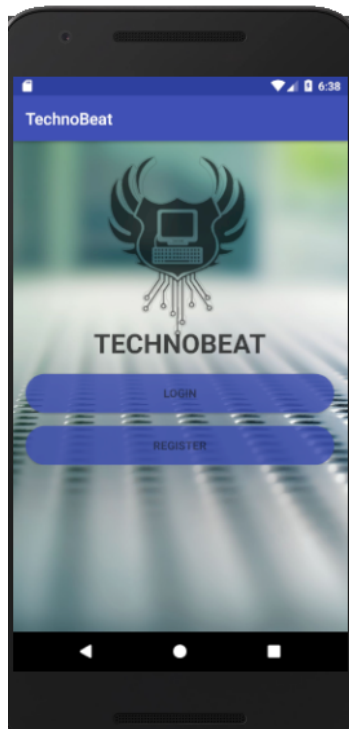
Output:

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:
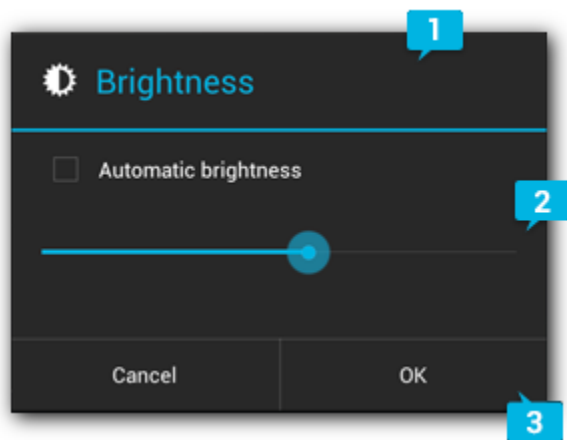
Menu:

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:

```kotlin
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)
        }
    }
}
```
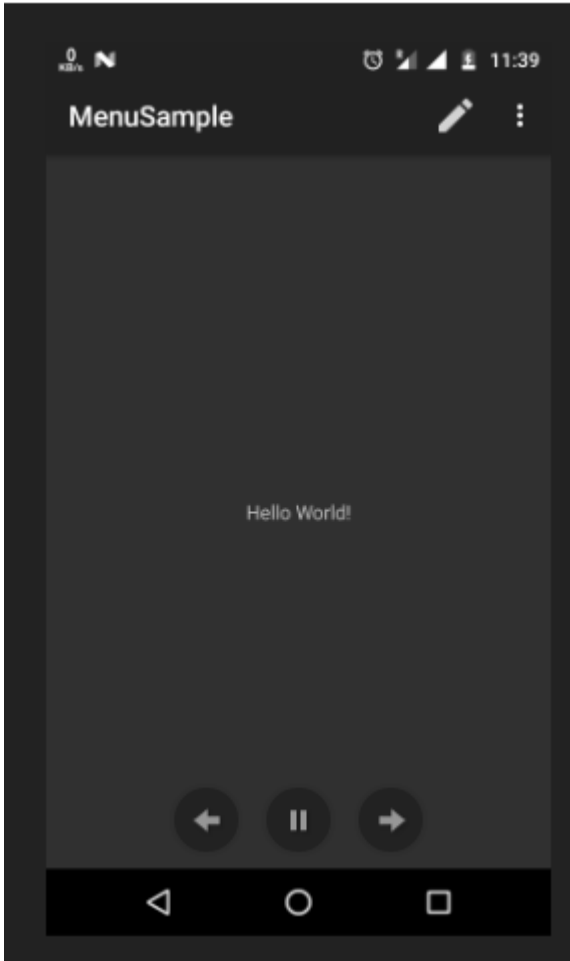
Practical 7

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.

2. 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:

```kotlin
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
<?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
<?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"
```

```xml
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:

```kotlin
package com.tutorialkart.sqlitetutorial

class UserModel(val userid: String, val name: String, val age: String)
```

DBContract.kt

```kotlin
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:

```kotlin
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
```

```kotlin
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))
```

```kotlin
                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:

```kotlin
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)
```

```kotlin
        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"
    }
}
```
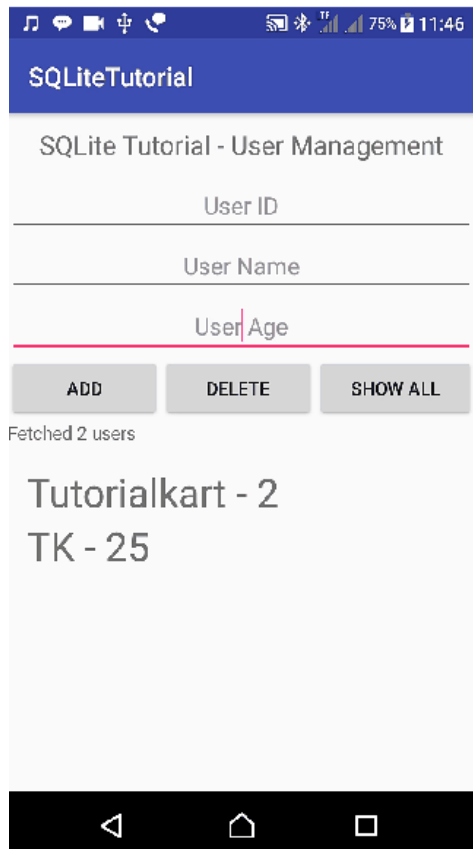
output:

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
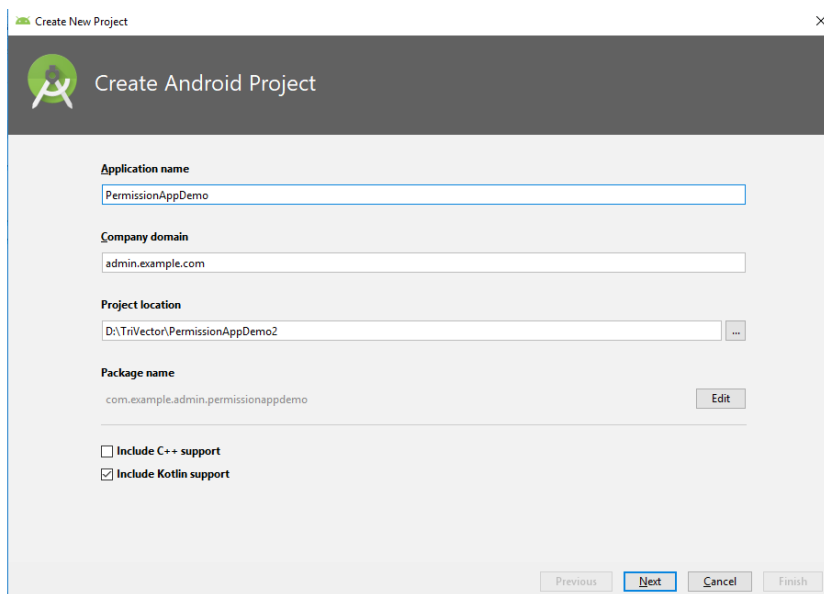import android.support.v7.app.AlertDialog

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

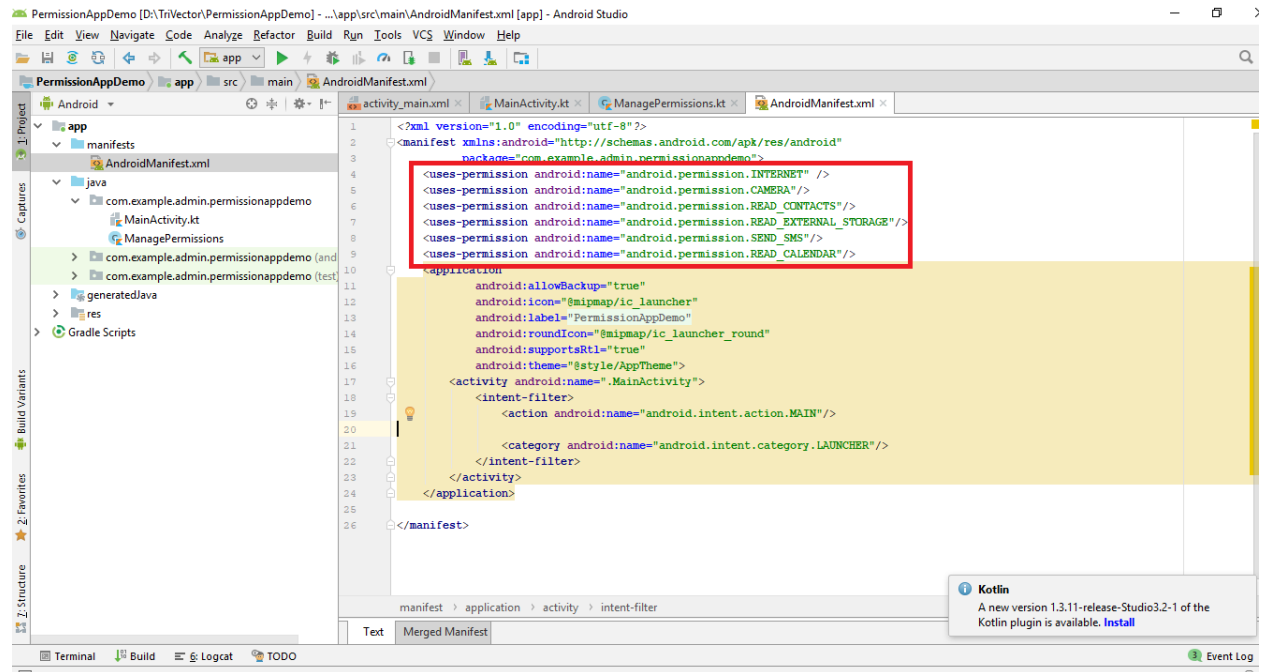1. Create a new project in android studio

2. An app must publicize the permissions it requires by including <uses

-permission> tags in the app manifest.

```xml
<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"/>
```


3. MainActivity.kt

```kotlin
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,
```

```kotlin
            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()
}
```
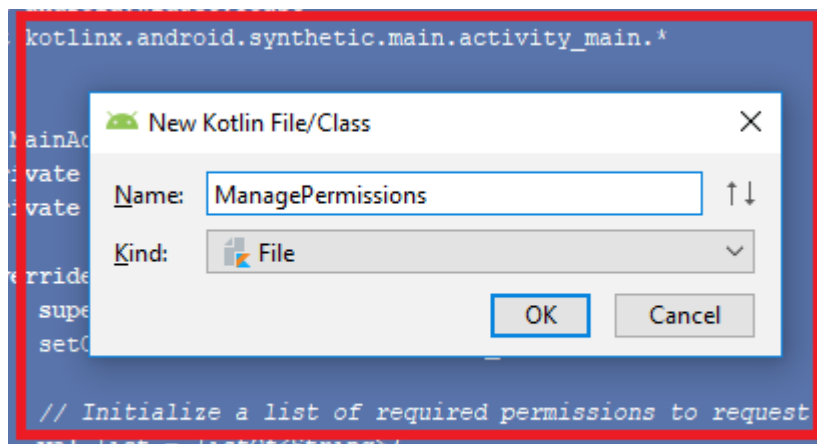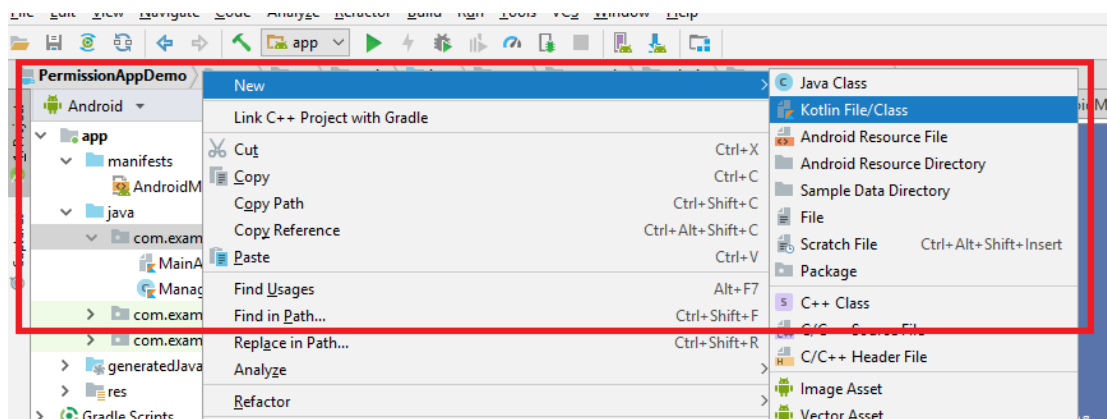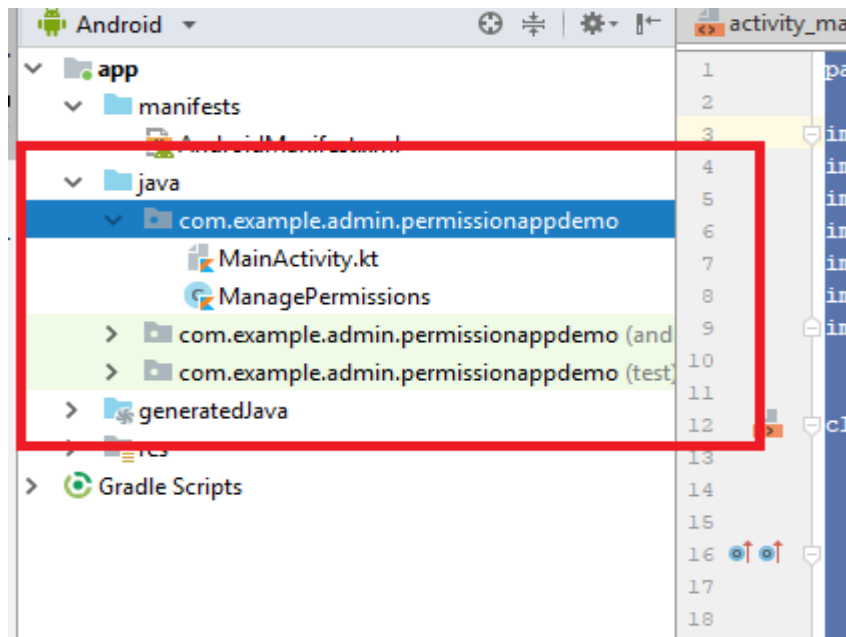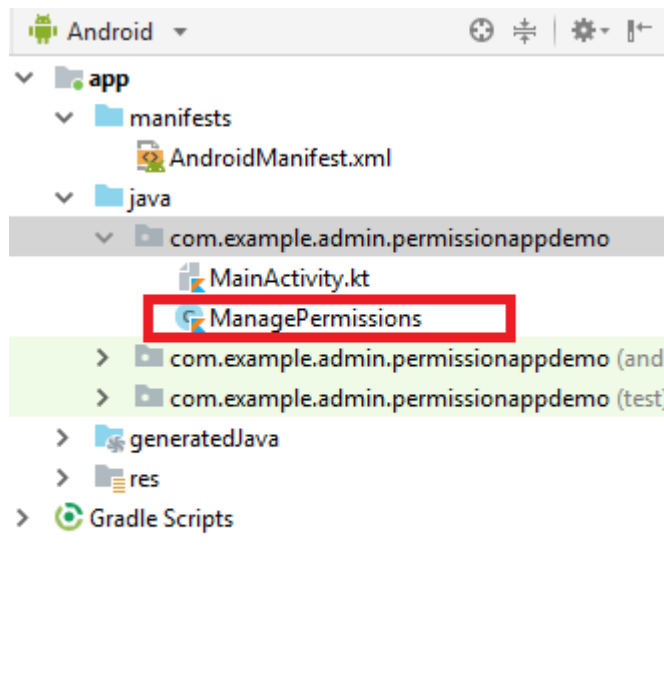
4. Create a New Kotlin Class

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

Class file is generated

5. Write the following code in the Class File

```kotlin
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
        }
```
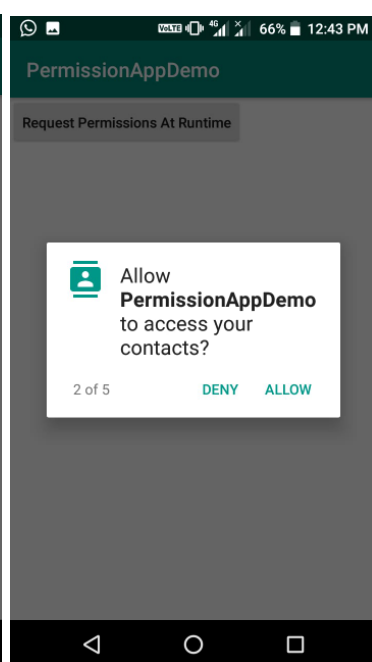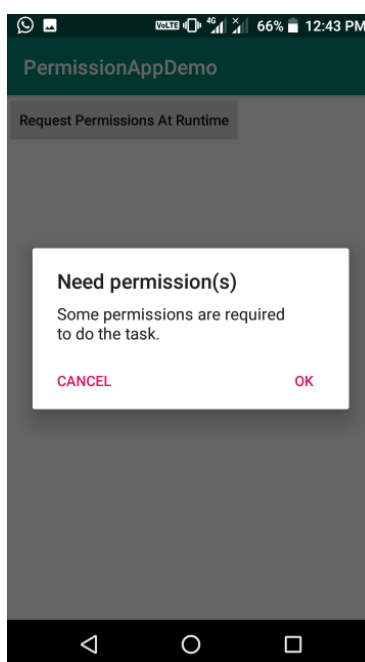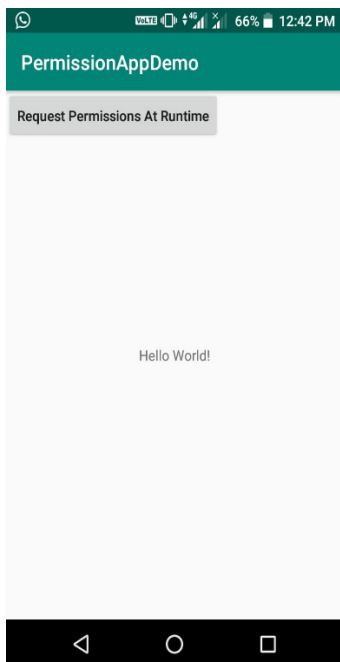
```kotlin
        return ""
    }


    // 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
    }
}
```
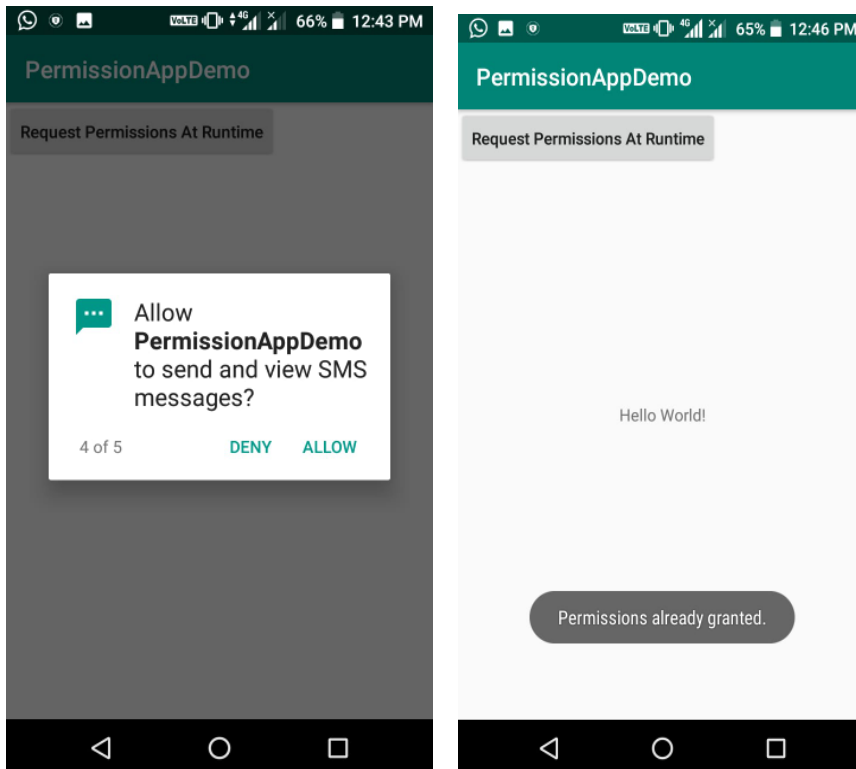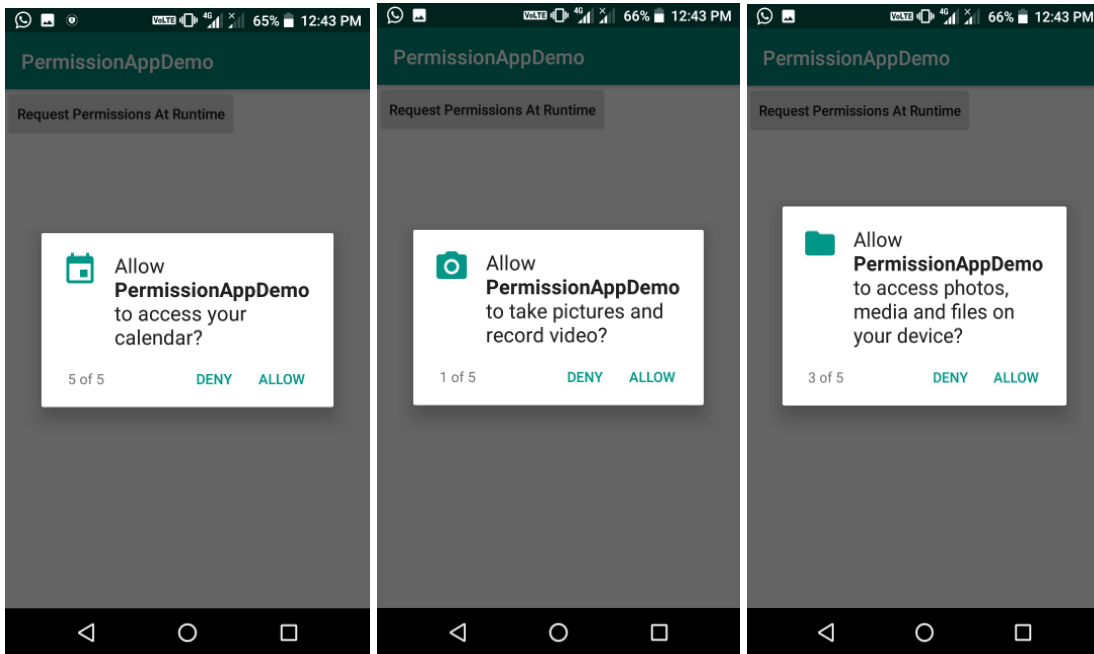
PRACTICAL 10

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:

```xml
    <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:

```kotlin
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:

```kotlin
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
<?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:

```kotlin
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)
    }
}
```