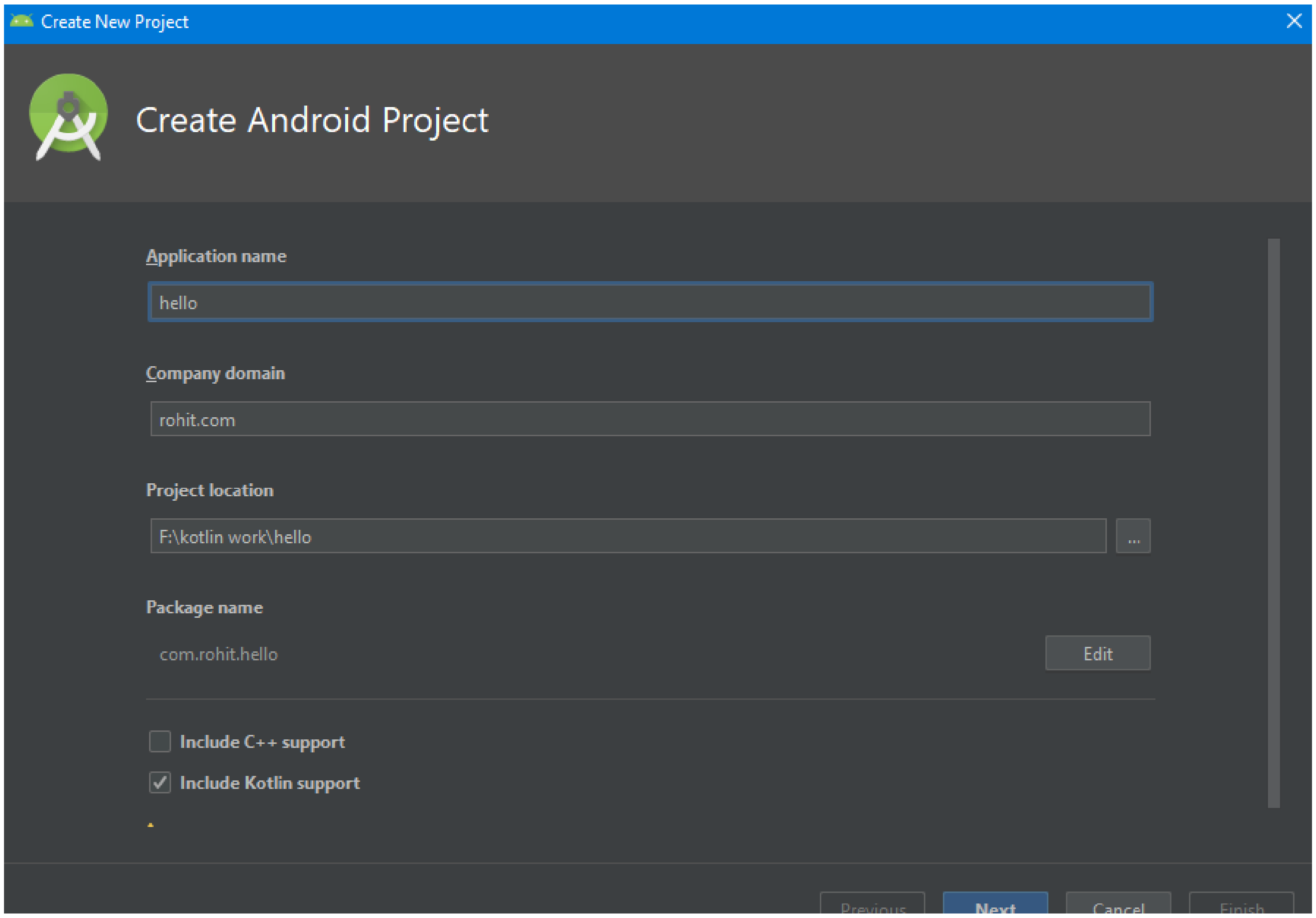
PRACTICAL 1

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.

Creating a project:



Application name: hello

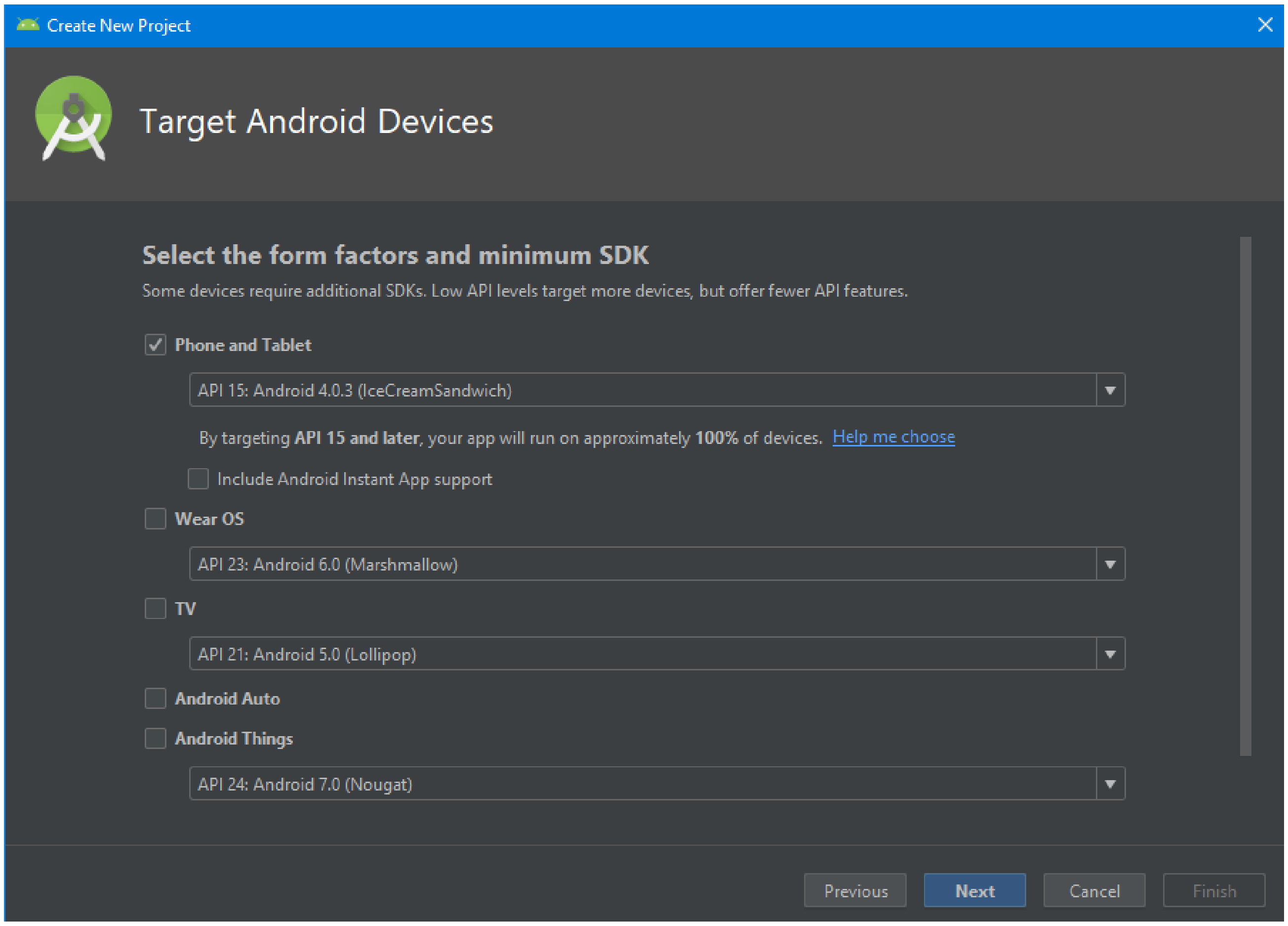
Company domain: rohit.com

Project location: F:\kotlin work\hello

Package name: com.rohit.hello

Include C++ support [ ]

Include Kotlin support [✓]



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)

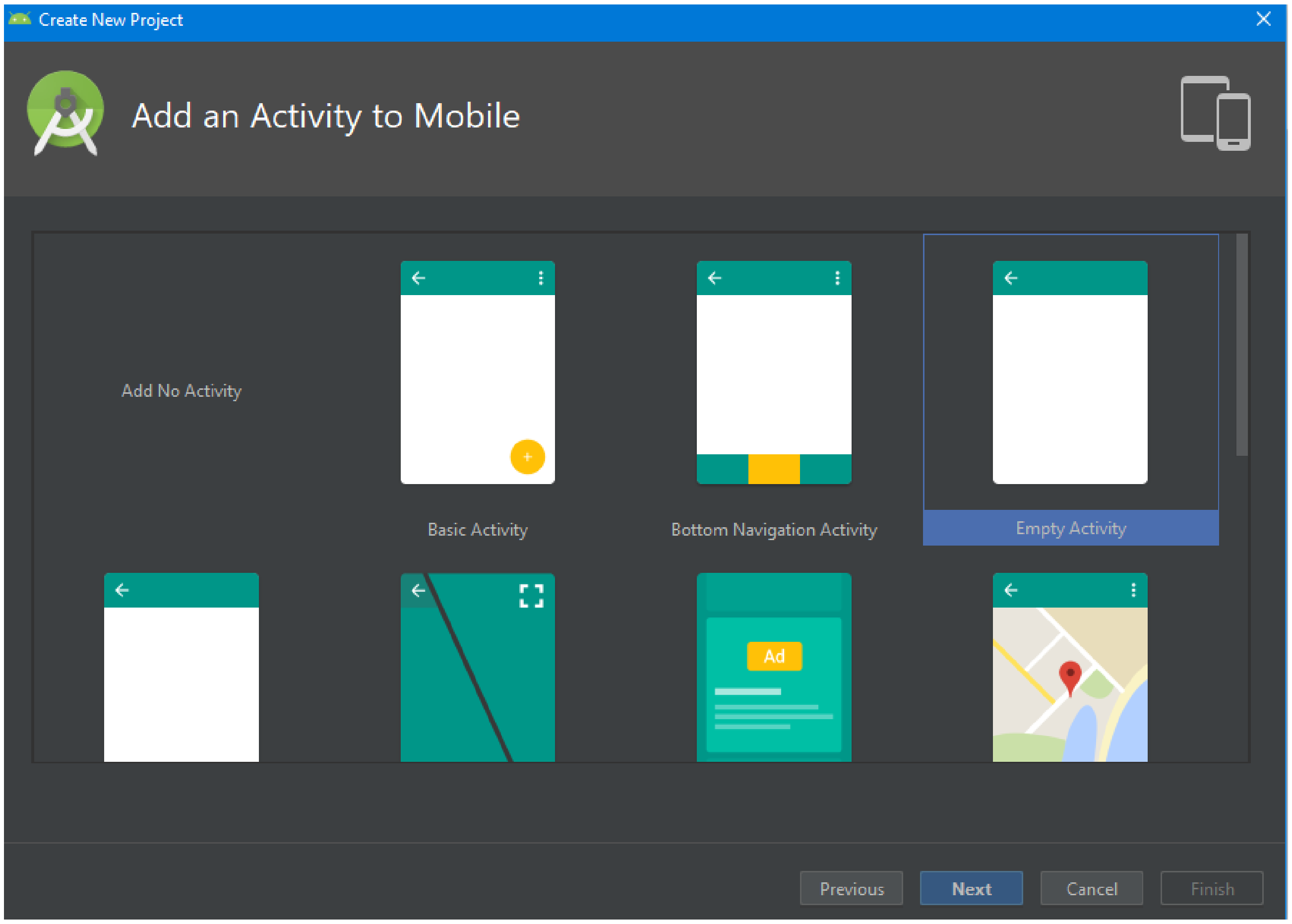
[ ] TV

API 21: Android 5.0 (Lollipop)

[ ] Android Auto

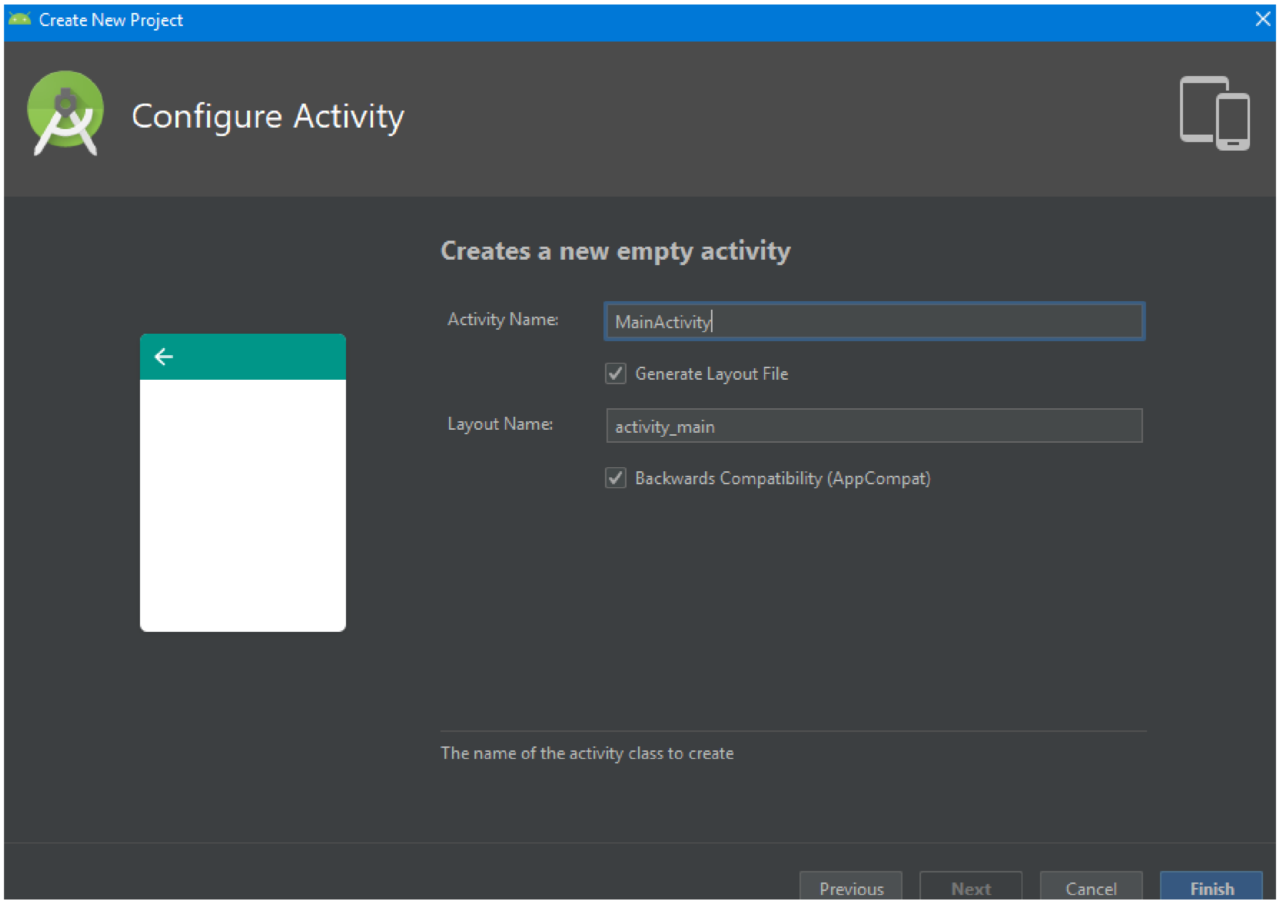
[ ] Android Things

API 24: Android 7.0 (Nougat)



Create New Project

Add an Activity to Mobile


Create New Project

Configure Activity

Creates a new empty activity

Activity Name: MainActivity

✓ Generate Layout File

Layout Name: activity_main

✓ Backwards Compatibility (AppCompat)

The name of the activity class to create

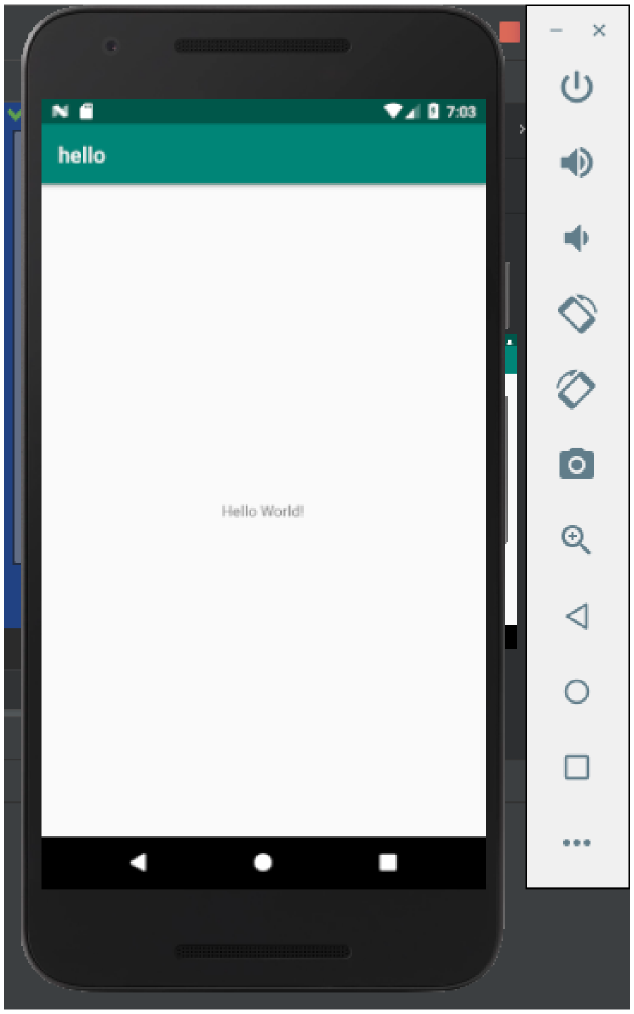
Activity_Main.kt

```
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>
```

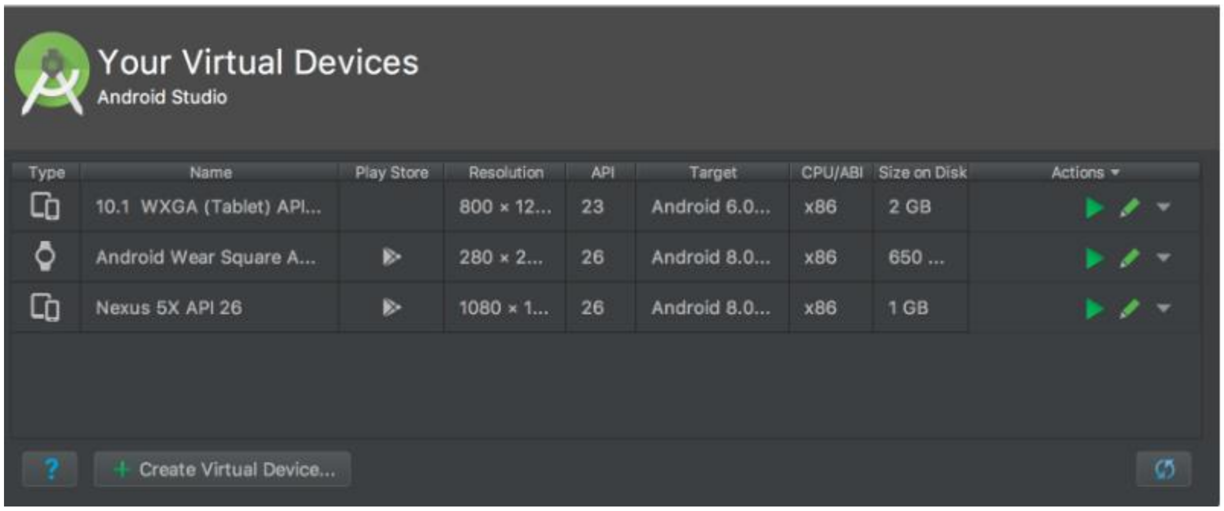
Application in AVD:



Create and manage virtual devices:

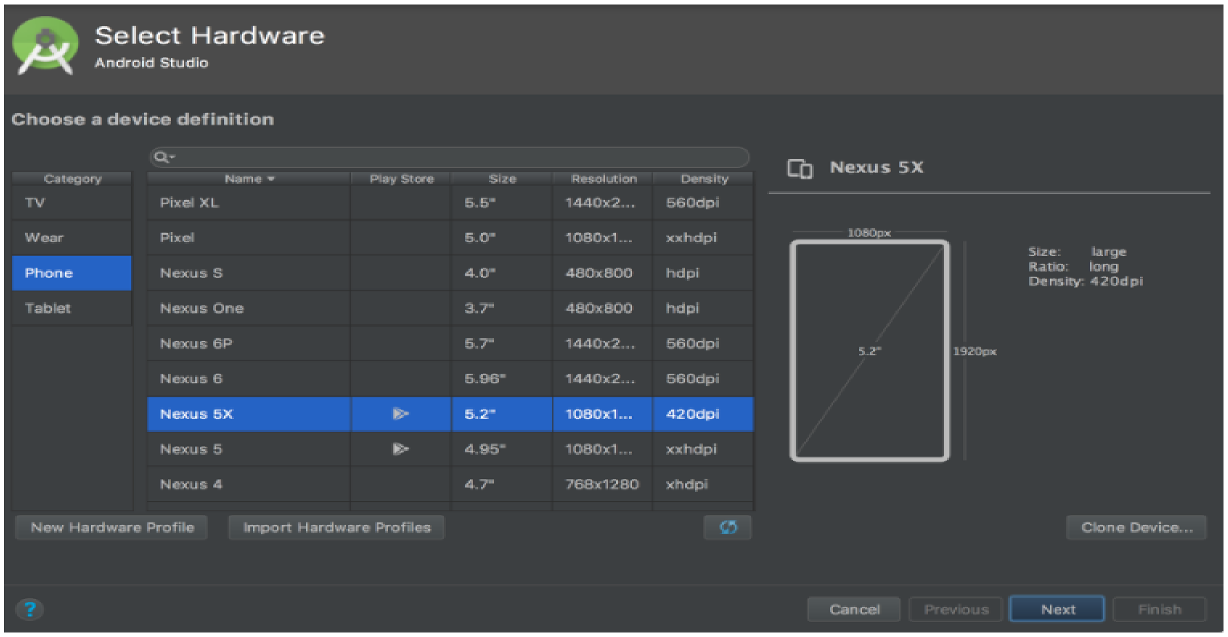
To open the AVD Manager, do one of the following:

- 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	



Select Hardware
Android Studio

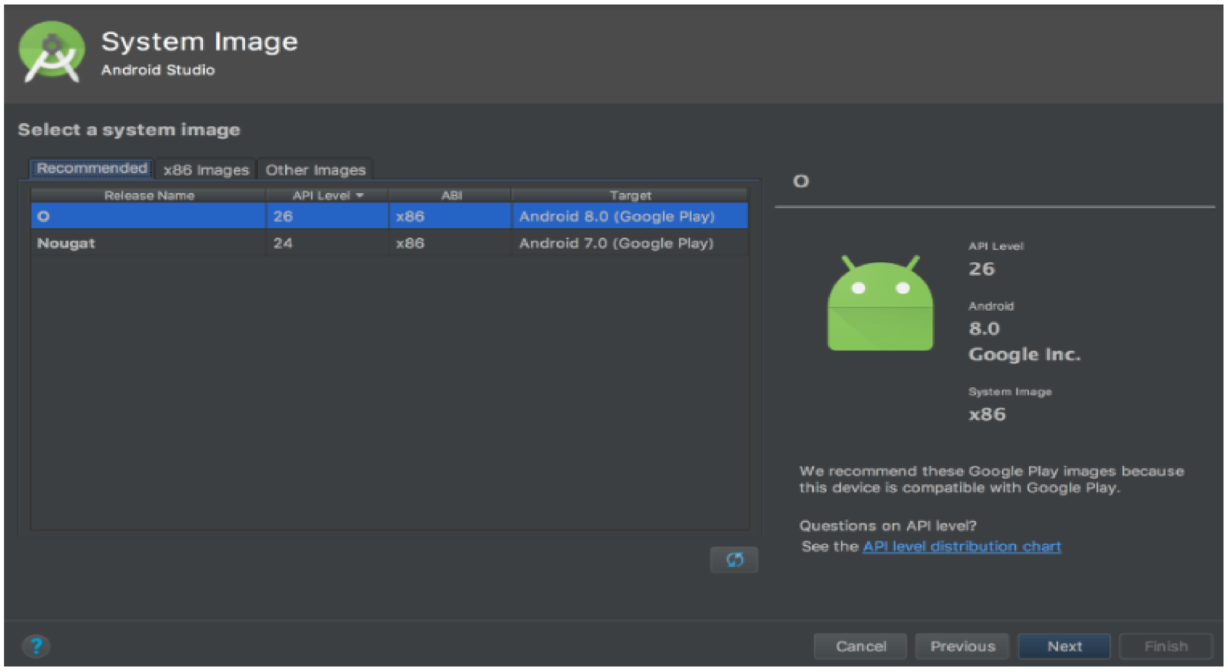
Choose a device definition

Category
TV
Wear
Phone
Tablet

Name	Play Store	Size	Resolution	Density
Pixel XL		5.5"	1440x2...	560dpi
Pixel		5.0"	1080x1...	xxhdpi
Nexus S		4.0"	480x800	hdpi
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

Size: large
Ratio: long
Density: 420dpi



System Image
Android Studio

Select a system image

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

O

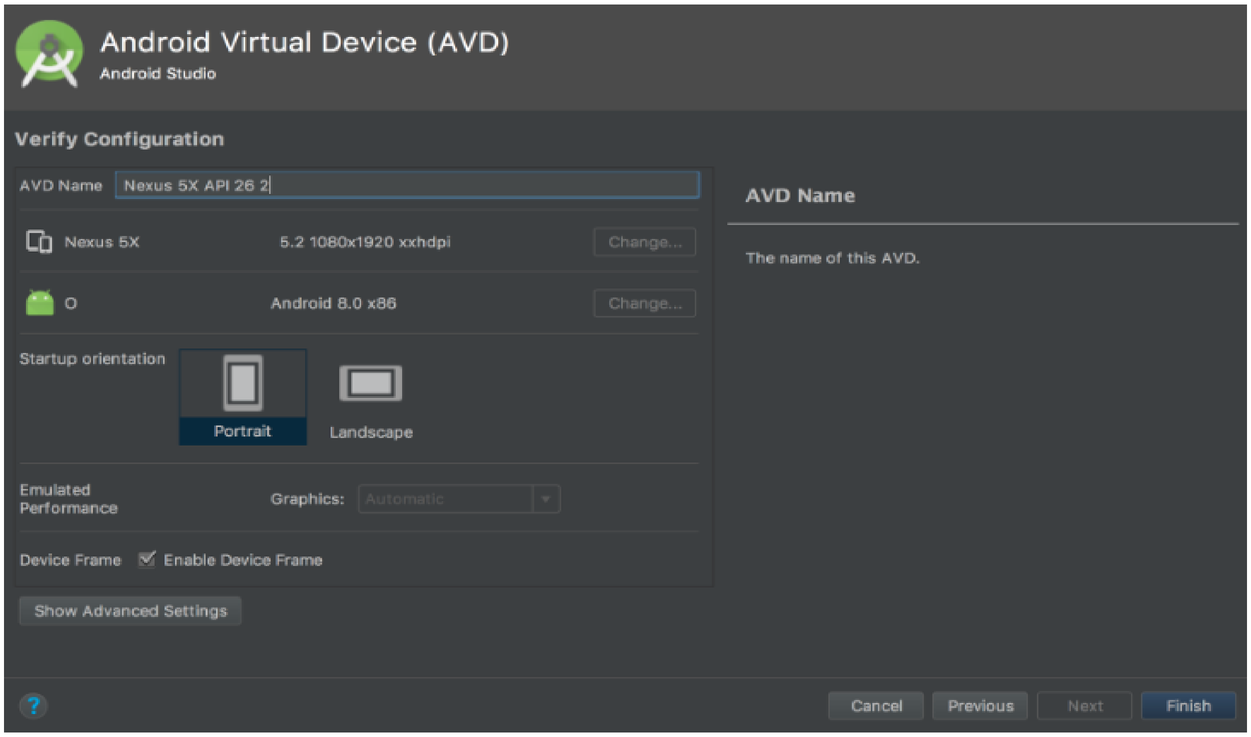
API Level
26

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



Android Virtual Device (AVD)
Android Studio

Verify Configuration

AVD Name: Nexus 5X API 26 2

Nexus 5X — 5.2 1080x1920 xxhdpi

O — Android 8.0 x86

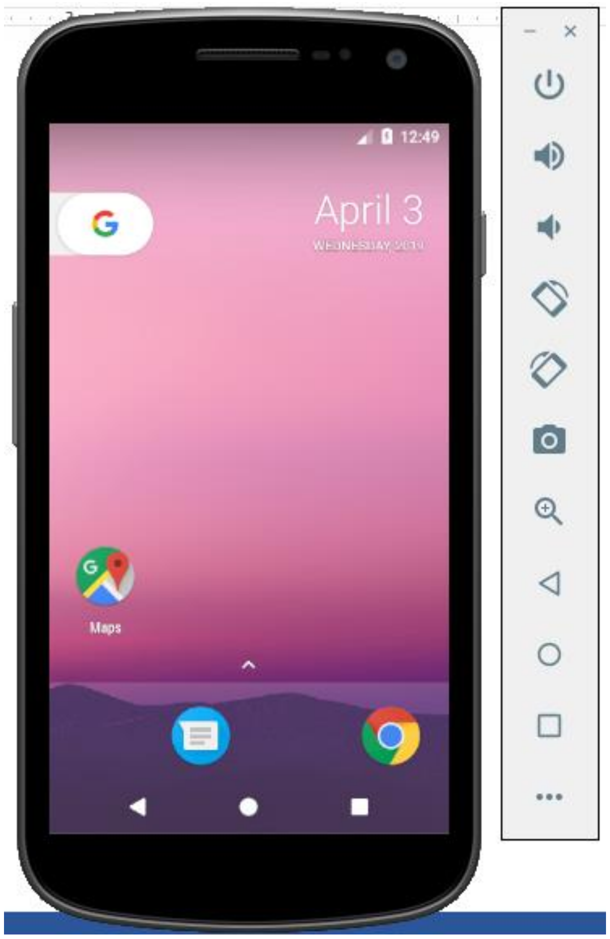
Startup orientation: Portrait / Landscape

Emulated Performance — Graphics: Automatic

Device Frame: ✓ Enable Device Frame

AVD Name

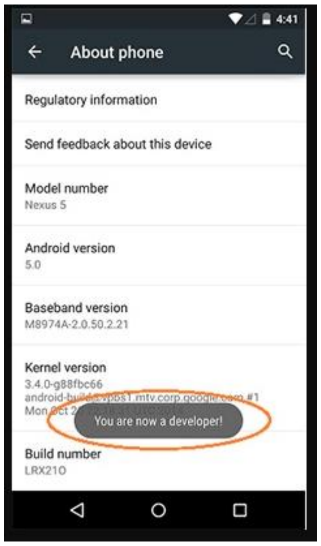
The name of this AVD.



Enable USB debugging

The very first step is to enable USB debugging on your Android device. To do this follow these steps On your phone (or tablet) go to **Settings=> About Phone**

Tap **Build Number** 7 times, after 7th time it will say **You are now a developer**.


4:41

About phone

Regulatory information

Send feedback about this device

Model number
Nexus 5

Android version
5.0

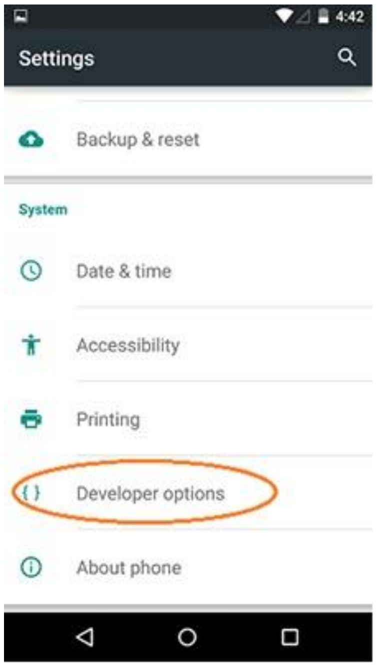
Baseband version
M8974A-2.0.50.2.21

Kernel version
3.4.0-g88fbc66
android-build@vpbs1.mtv.corp.google.com #1
Mon Oct 2

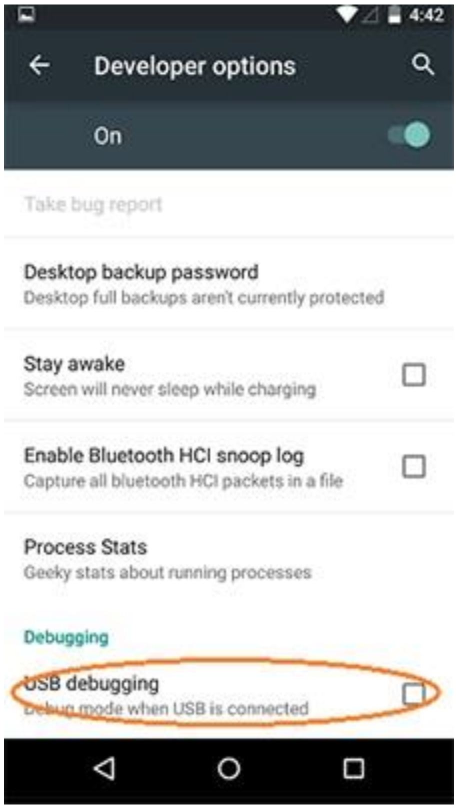
You are now a developer!

Build number
LRX210

You will notice Developer’s Options are now available.



Go to the **Developer option** and enable **USB debugging**


4:42

Developer options

On

Take bug report

Desktop backup password
Desktop full backups aren't currently protected

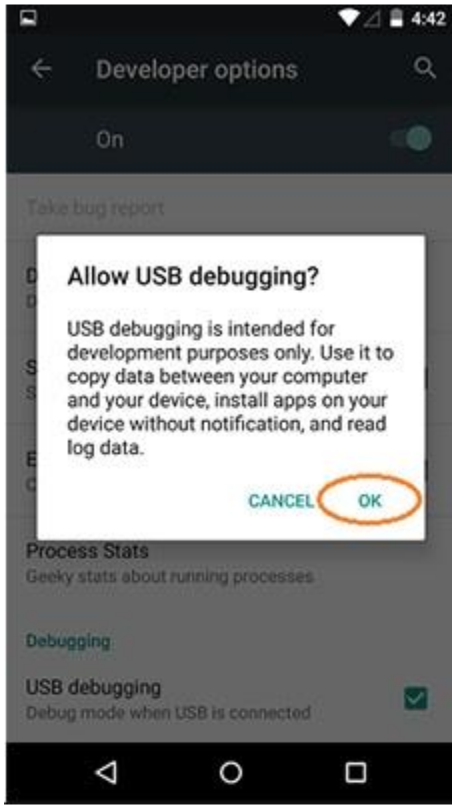
Stay awake ☐
Screen will never sleep while charging

Enable Bluetooth HCI snoop log ☐
Capture all bluetooth HCI packets in a file

Process Stats
Geeky stats about running processes

Debugging

USB debugging ☐
Debug mode when USB is connected


4:42

Developer options

On

Take bug report

Allow USB debugging?

USB debugging is intended for development purposes only. Use it to copy data between your computer and your device, install apps on your device without notification, and read log data.

CANCEL [✓ OK]

Process Stats
Geeky stats about running processes

Debugging

USB debugging ✓
Debug mode when USB is connected

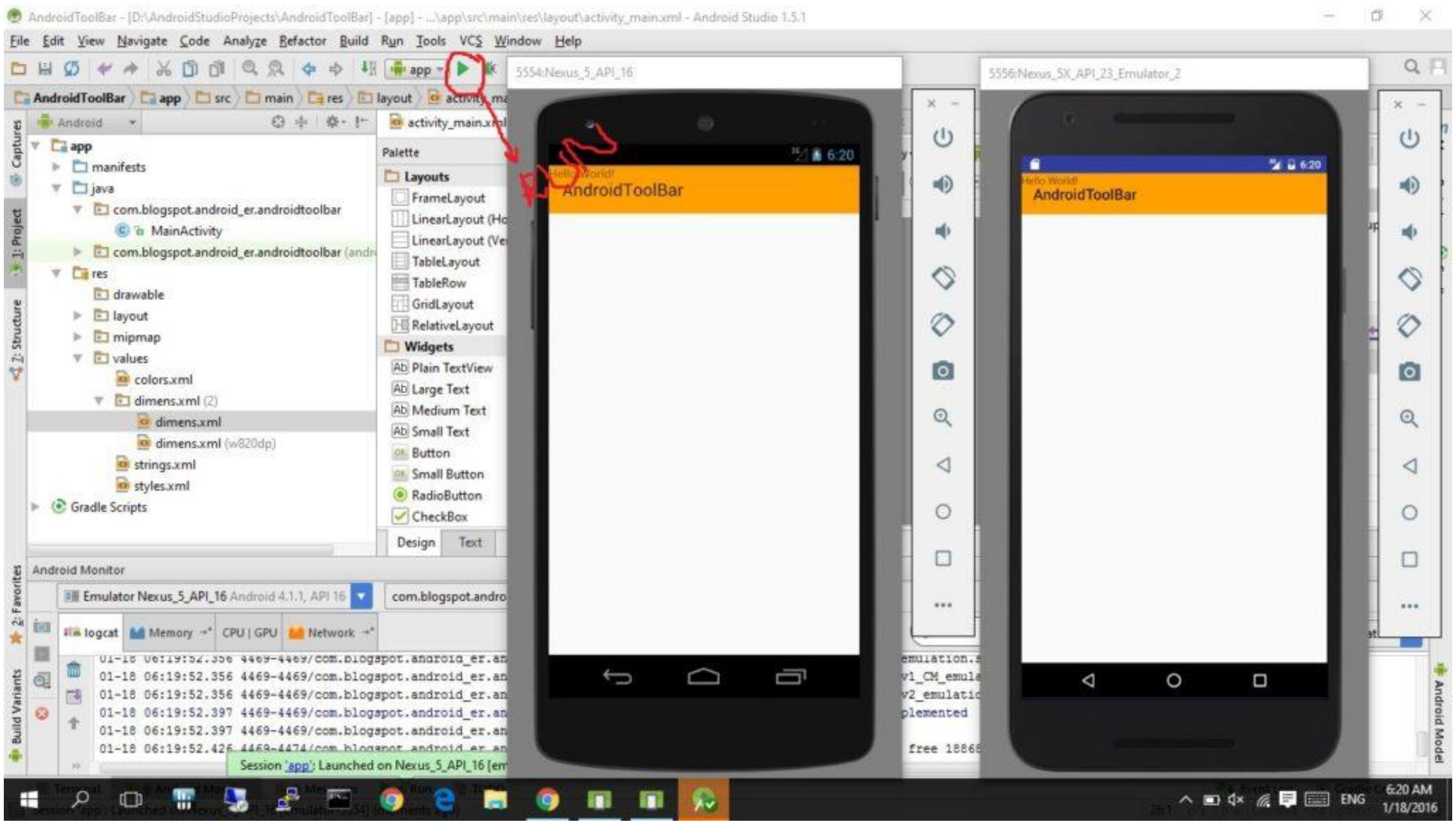
<u>Install USB driver</u>

Next step is to install USB driver for your Android device. For this follow instructions from your device manufacturer. For example, I am using Android smartphone of Huawei, so I just downloaded Huawei USB driver from their official website. If your device uses Google USB driver you can download from this link http://developer.android.com/sdk/win-usb.html. After installation you need to update it. Make sure your device is connected through a USB cable. Go to the **Control Panel => Device Manager** then locate and right click your Android device and click **Update driver software.**

Note: Make sure your Android device is not sleeping while connected through USB cable.

Step 3: Run your app

No you can run your Android app. Right click on the app and click **Run**. Or simply select run option from the tool bar menu shown below.

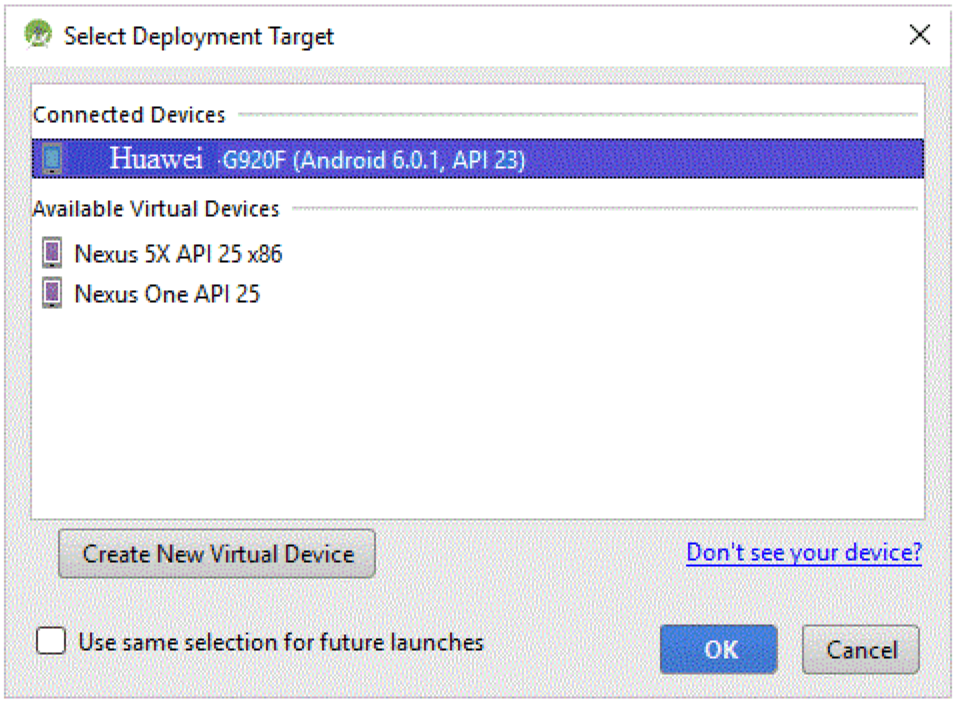

AndroidToolBar - [D:\AndroidStudioProjects\AndroidToolBar] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.5.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
5554:Nexus_5_API_16
5556:Nexus_5X_API_23_Emulator_2
AndroidToolBar
Hello World!
Android Monitor
Emulator Nexus_5_API_16 Android 4.1.1, API 16
logcat
Memory
CPU | GPU
Network

```
01-18 06:19:52.356 4469-4469/com.blogspot.android_er.an
01-18 06:19:52.356 4469-4469/com.blogspot.android_er.an
01-18 06:19:52.356 4469-4469/com.blogspot.android_er.an
01-18 06:19:52.397 4469-4469/com.blogspot.android_er.an
01-18 06:19:52.397 4469-4469/com.blogspot.android_er.an
01-18 06:19:52.426 4469-4474/com.blogspot.android_er.an
```

Session 'app': Launched on Nexus_5_API_16 (em

Android Studio screen

A window **Select Deployment Target** will appear, and a list of available devices will appear. Choose your device and click **OK**. Android Studio will run your application in your Android device.


Select Deployment Target

Connected Devices

Huawei G920F (Android 6.0.1, API 23)

Available Virtual Devices

Nexus 5X API 25 x86

Nexus One API 25

Create New Virtual Device
Don't see your device?
Use same selection for future launches
OK
Cancel

Deployment target

Click on **Use same selection for future launches** if you want to save this setting for later apps.

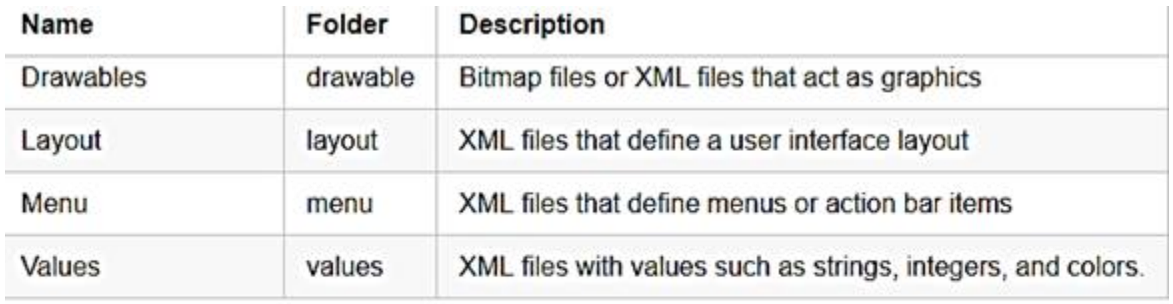
PRACTICAL 2

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

In this practical we will learn how to organize application resources, specify alternative resources and access them in your applications.

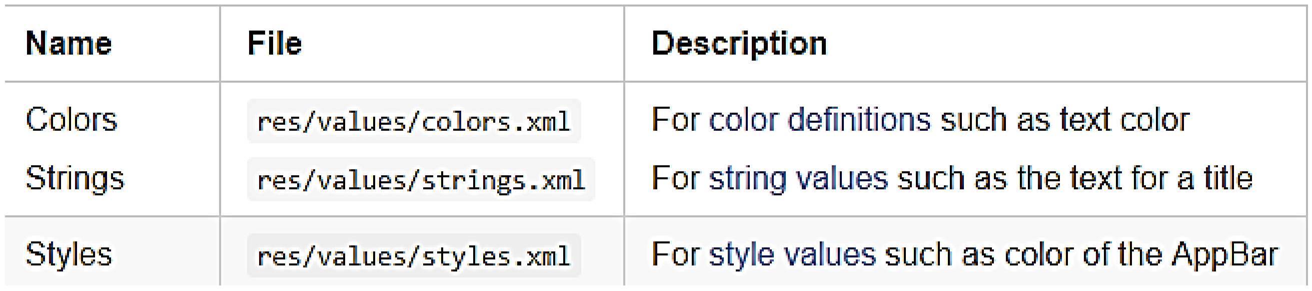
Types of Resources

The following are the most common types of resources within Android apps:



Name	Folder	Description
Drawables	drawable	Bitmap files or XML files that act as graphics
Layout	layout	XML files that define a user interface layout
Menu	menu	XML files that define menus or action bar items
Values	values	XML files with values such as strings, integers, and colors.

In addition, note the following key files stored within the `values` folder mentioned above:



Name	File	Description
Colors	`res/values/colors.xml`	For color definitions such as text color
Strings	`res/values/strings.xml`	For string values such as the text for a title
Styles	`res/values/styles.xml`	For style values such as color of the AppBar

<u>Example</u>

Demonstration the use of various resource files

activity_main.xml

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


    <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/sun"
            android:id="@+id/img"
            android:layout_centerHorizontal="true"
            android:layout_alignParentTop="true"
            android:layout_marginTop="117dp"/>
    <Button
            android:id="@+id/btn"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/button_name"
            android:layout_below="@+id/img"
            android:layout_centerHorizontal="true"
            android:background="@color/btn_color"
            android:layout_marginTop="123dp"
            android:onClick="show"/>
</RelativeLayout>
```

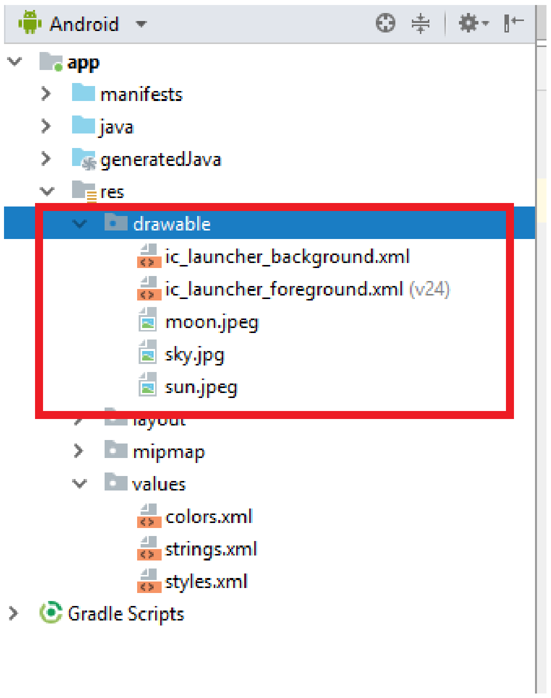
colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#008577</color>
    <color name="colorPrimaryDark">#00574B</color>
    <color name="colorAccent">#D81B60</color>
    <color name="btn_color">#FFC49662</color>
</resources>
```

string.xml

```
<resources>
    <string name="app_name">Resources</string>
    <string name="button_name">Click on me</string>
</resources>
```

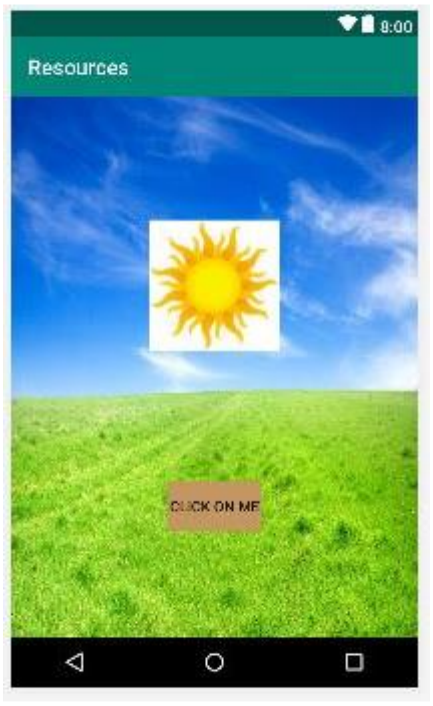
In Drawable add the image which you want to display



Mainactivity.xml

```
package com.example.admin.resources

import android.support.v7.app.AppCompatActivity

import android.os.Bundle

import android.view.View

import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

    }

    fun show(view: View){

        img.setImageResource(R.drawable.moon)

    }

}
```

Output:

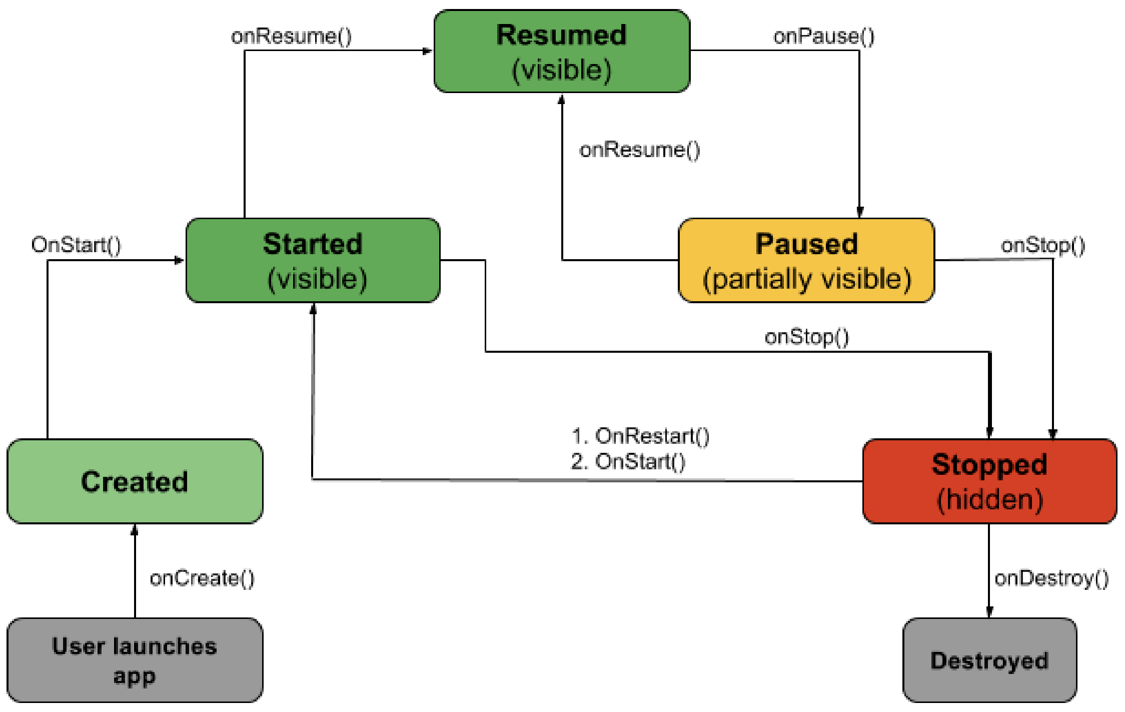


PRACTICAL 3

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.

```
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.

Coordinate Layout

Coordinator Layout is a super-powered Frame Layout. Coordinator Layout is a general-purpose container that allows for coordinating interactive behaviors between its children. Coordinator Layout manages interactions between its children, and as such needs to contain all the Views that interact with each other. The two general cases supported by Coordinator Layout are:

As a top-level content layout (meaning Coordinator Layout is at the root of all views within an activity or fragment).

As a container for a specific interaction with one or more child views.

By specifying Behaviors for child views of a Coordinator Layout you can provide many different interactions within a single parent and those views can also interact with one another.

XML Attributes

1. android:layout_gravity: Specifies the gravity of the child relative to the parent. If you specify an Anchor using app:layout_anchor, then this attribute would be ignored. And you have to use

2**. app:layout_anchorGravity** to position the child. Do not use both of these together in any view. It may cause of unexpected result.

3. app:layout_anchor: This attribute can be set on children of the Coordinator Layout to attach them to another view. The value would be the id of an anchor view that this view should position relative to.Note that, the anchor view can be any child View (a child of a child of a child of a CoordinatorLayout).

4. app:layout_anchorGravity: Specifies how an object should position relative to an anchor, on both the X and Y axes, within its parent's bounds.

5. **app:layout_insetEdge:** Specifies how this view insets the Coordinator Layout and make some other views dodge it. The child consumes the area of the screen it occupies and other children should not be placed in that area. Bottom is the default inset Edge of SnackBar

6. app:layout_dodgeInsetEdges: Specifies which edges of the child should be offset by insets. Bottom is the default dodgeInsetEdges of FAB button.

7.app:layout_behavior: The class name of a Behavior class defining.

Example:

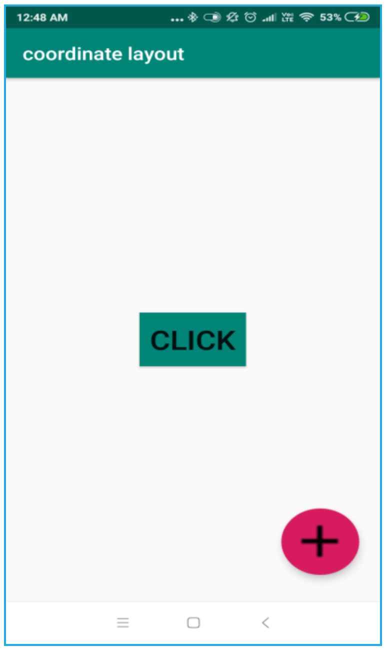
activity_main.xml

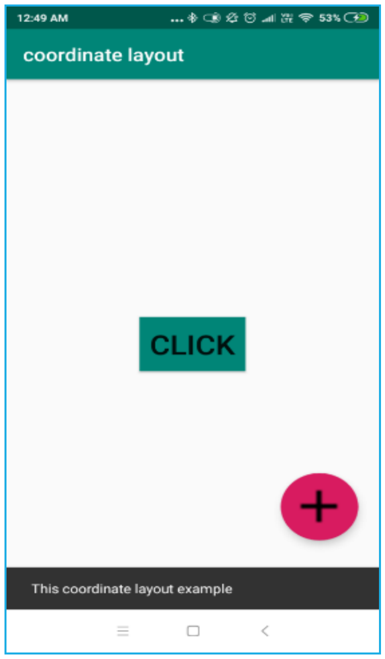
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/coordinate"
    tools:context=".MainActivity">
  <Button
        android:id="@+id/btn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Click"
        android:gravity="center"
        android:padding="10dp"
        android:layout_gravity="center"
        android:background="@color/colorPrimary"
        android:textSize="30sp"
        android:onClick="show"
  />
  <android.support.design.widget.FloatingActionButton
        android:id="@+id/fbtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|right"
        android:layout_marginRight="20dp"
        android:layout_marginBottom="30dp"
        android:src="@drawable/plus_icon"
        app:fabCustomSize="75dp"
        app:maxImageSize="60dp"
        android:scaleType="center"/>
</android.support.design.widget.CoordinatorLayout>
```

MainActivity.kt

```
    package com.example.user.coordinatelayout
    import android.support.v7.app.AppCompatActivity
    import android.os.Bundle
    import android.support.design.widget.Snackbar
    import android.view.View
    import kotlinx.android.synthetic.main.activity_main.*
    class MainActivity : AppCompatActivity() {
        override fun onCreate(savedInstanceState: Bundle?) {
            super.onCreate(savedInstanceState)
            setContentView(R.layout.activity_main)
        }

        fun show(view: View){

            var snackbar=Snackbar.make(coordinate,"This coordinate layout
    example",Snackbar.LENGTH_SHORT)

            snackbar.show()

        }

    }
```

Output:





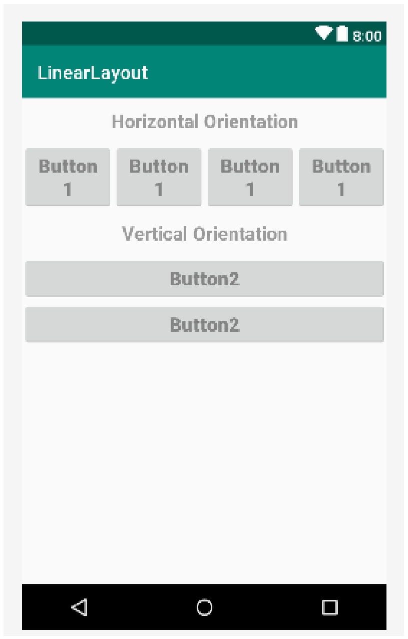
This coordinate layout example

LinearLayout

Linear layout is a simple layout used in android forlayout designing. In the Linear layout all the elements are displayed in linear fashion means all the childs/elements of a linear layout are displayed according to its orientation. The value for orientation property can be either horizontal or vertical.

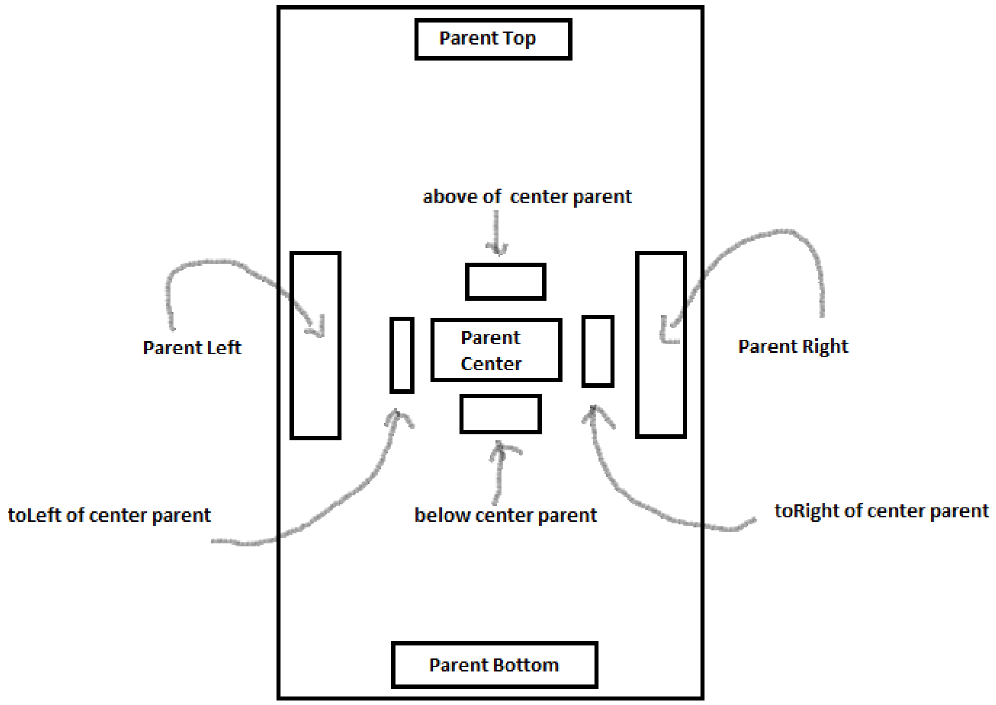
XML attributes:

1. **android:baselineAligned**:When set to false, prevents the layout from aligning its children's baselines.
2. **android:baselineAlignedChildIndex**:When a linear layout is part of another layout that is baseline aligned, it can specify which of its children to baseline align to (that is, which child TextView).
3. **android:divider:**Drawable to use as a vertical divider between buttons.
4. **android:gravity**:Specifies how an object should position its content, on both the X and Y axes, within its own bounds.
5. **android:measureWithLargestChild**:When set to true, all children with a weight will be considered having the minimum size of the largest child.
6. **android:orientation**:Should the layout be a column or a row? Use "horizontal" for a row, "vertical" for a column.
7. **android:weightSum**:Defines the maximum weight sum.



RelativeLayout:

The Relative Layout is very flexible layout used in android for custom layout designing. It gives us the flexibility to position our component/view based on the relative or sibling component's position. Just because it allows us to position the component anywhere we want so it is considered as most flexible layout. For the same reason Relative layout is the most used layout after the Linear Layout in Android. It allow its child view to position relative to each other or relative to the container or another container.In Relative Layout, you can use "above, below, left and right" to arrange the component's position in relation to other component.



1.android:layout:_above:Positions the bottom edge of this view above the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name"

2.android:layout_alignBottom:
Makes the bottom edge of this view match the bottom edge of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

3.android:layout_alignLeft:Makes the left edge of this view match the left edge of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

4.android:layout_alignParentBottom:If true, makes the bottom edge of this view match the bottom edge of the parent. Must be a boolean value, either "true" or "false".

5.android:layout_alignParentEnd:If true, makes the end edge of this view match the end edge of the parent. Must be a boolean value, either "true" or "false".

6.android:layout_alignParentLeft:If true, makes the left edge of this view match the left edge of the parent. Must be a boolean value, either "true" or "false".

7.android:layout_alignParentRight:If true, makes the right edge of this view match the right edge of the parent. Must be a boolean value, either "true" or "false".

8.android:layout_alignParentStart
If true, makes the start edge of this view match the start edge of the parent. Must be a boolean value, either "true" or "false".

9.android:layout_alignParentTop:If true, makes the top edge of this view match the top edge of the parent. Must be a boolean value, either "true" or "false".

10.android:layout_alignRight:Makes the right edge of this view match the right edge of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

11.android:layout_alignStart:Makes the start edge of this view match the start edge of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

12.android:layout_alignTop:Makes the top edge of this view match the top edge of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

13.android:layout_below

Positions the top edge of this view below the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

14.android:layout_centerHorizontal:If true, centers this child horizontally within its parent. Must be a boolean value, either "true" or "false".

15.android:layout_centerInParent:If true, centers this child horizontally and vertically within its parent. Must be a boolean value, either "true" or "false".
16.android:layout_centerVertical:If true, centers this child vertically within its parent. Must be a boolean value, either "true" or "false".

17.android:layout_toEndOf:Positions the start edge of this view to the end of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

18.android:layout_toLeftOf:Positions the right edge of this view to the left of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

19.android:layout_toRightOf:
Positions the left edge of this view to the right of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

20.android:layout_toStartOf.Positions the end edge of this view to the start of the given anchor view ID and must be a reference to another resource, in the form "@[+][package:]type:name".

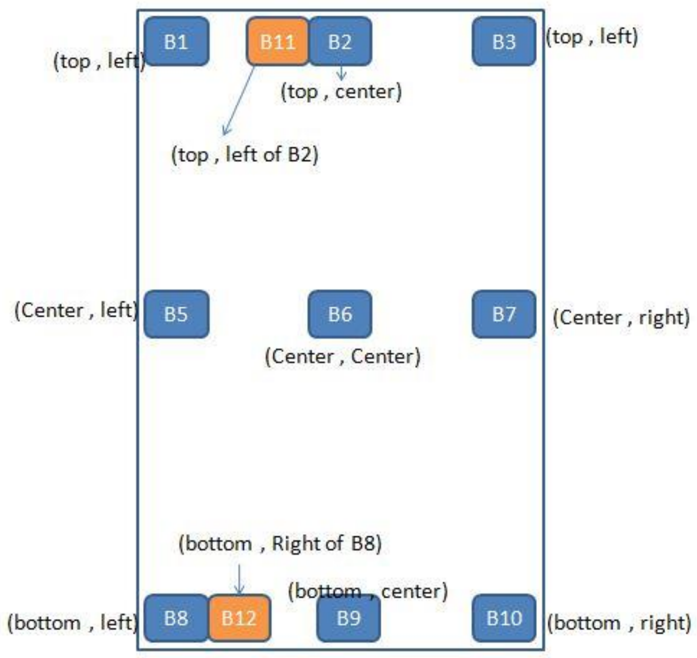
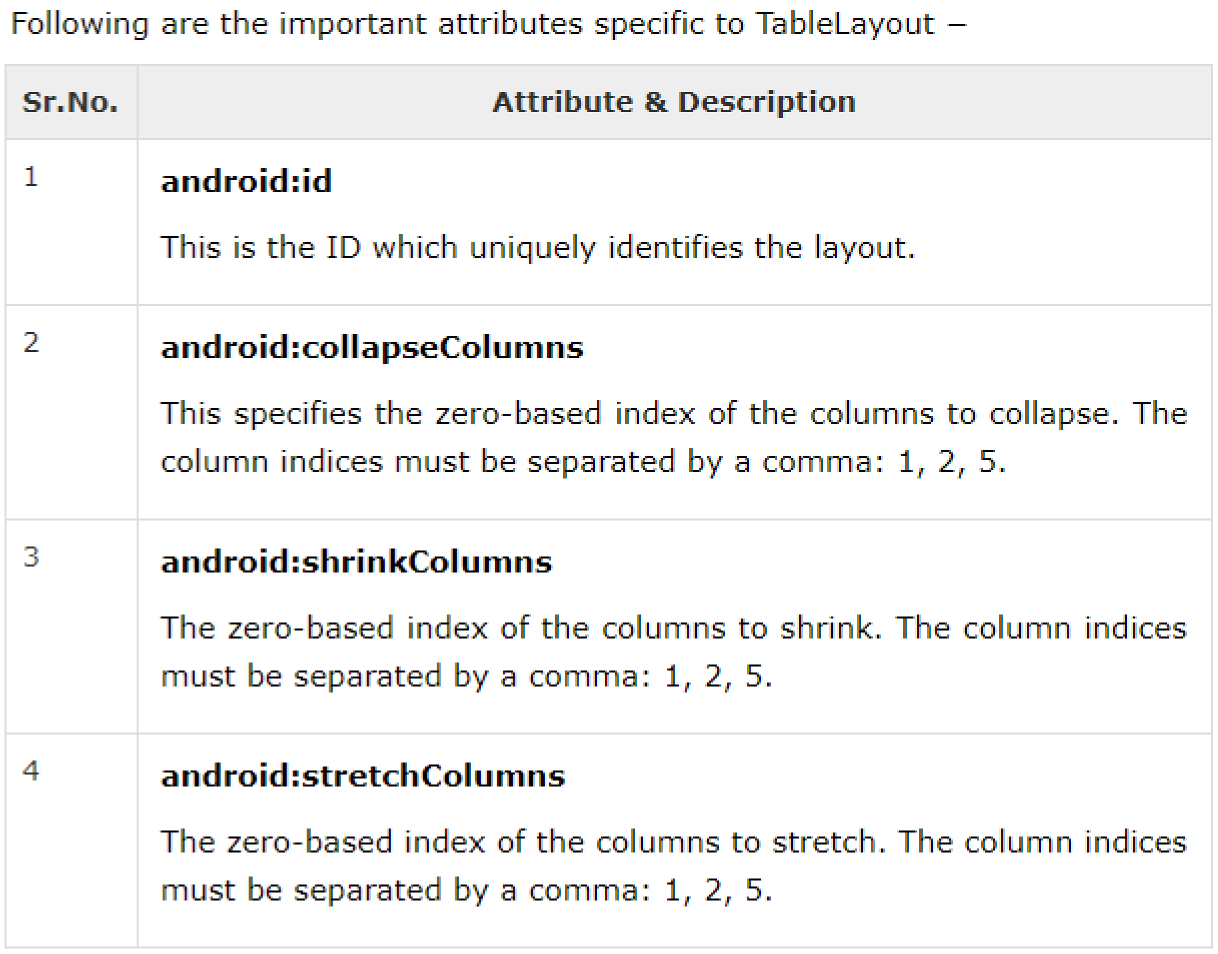


Table Layout

Android TableLayout going to be arranged groups of views into rows and columns. You will use the <TableRow> element to build a row in the table. Each row has zero or more cells; each cell can hold one View object.



Following are the important attributes specific to TableLayout –

Sr.No.	Attribute & Description
1	**android:id** This is the ID which uniquely identifies the layout.
2	**android:collapseColumns** This specifies the zero-based index of the columns to collapse. The column indices must be separated by a comma: 1, 2, 5.
3	**android:shrinkColumns** The zero-based index of the columns to shrink. The column indices must be separated by a comma: 1, 2, 5.
4	**android:stretchColumns** The zero-based index of the columns to stretch. The column indices must be separated by a comma: 1, 2, 5.

activity_main.xml

```
<TableLayout
        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:gravity="center">
    <TableRow
            android:layout_width="match_parent"
            android:layout_height="50dp"
            android:id="@+id/prerow">
        <TextView
                android:text=""
                android:layout_width="723dp"
                android:layout_height="50dp"
                android:id="@+id/pre_data"
                android:textSize="30sp"

        />
    </TableRow>
    <TableRow
            android:layout_width="match_parent"
            android:layout_height="70dp">
        <TextView
                android:text=""
                android:layout_width="723dp"
                android:layout_height="79dp"
                android:id="@+id/curr_data"
                android:textSize="30sp"
                tools:text="0"/>
    </TableRow>
    <TableLayout
            android:layout_width="match_parent"
            android:layout_height="90dp"
            android:layout_marginTop="2dp"
            android:layout_marginRight="3dp">
        <TableRow
                android:layout_width="match_parent"
                android:layout_height="90dp"
                android:orientation="vertical">

            <Button
                    android:text="CE"
                    android:layout_width="95dp"
                    android:layout_height="79dp"
                    android:layout_column="1"
                    android:layout_marginStart="5dp"
                    android:layout_marginLeft="5dp"
                    android:background="@color/colorPrimary"
                    android:textSize="20sp"
                    android:id="@+id/c"
                    android:onClick="ce"
                    android:layout_weight="50"/>
            <Button
                    android:text="C"
                android:layout_width="95dp"
                android:layout_height="79dp"
                android:layout_marginStart="5dp"
                android:layout_marginLeft="5dp"
                android:layout_column="2"
                android:textSize="20dp"
                android:background="@color/colorPrimary"
                android:id="@+id/ce"
                android:onClick="clear"
                android:layout_weight="50"/>
            <Button
                android:text="@string/rem"
                android:layout_width="95dp"
                android:layout_height="79dp"
                android:layout_marginStart="5dp"
                android:layout_marginLeft="5dp"
                android:layout_column="3"
                android:textSize="20dp"
                android:background="@color/colorPrimary"
                android:id="@+id/rem"
                android:onClick="remove"
                android:layout_weight="50"/>
            <Button
                android:text="/"
                android:layout_width="95dp"
                android:layout_height="79dp"
                android:layout_marginStart="5dp"
                android:layout_marginLeft="5dp"
                android:layout_column="4"
                android:textSize="20dp"
                android:background="@color/colorPrimary"
                android:id="@+id/divide"
                android:onClick="divide"
                android:layout_weight="50"/>
          </TableRow>
        </TableLayout>
        <TableLayout
            android:layout_width="match_parent"
            android:layout_height="90dp"
            android:layout_marginTop="2dp"
            android:layout_marginRight="3dp">
```

```
<TableRow
        android:layout_width="match_parent"
        android:layout_height="90dp">

    <Button
            android:text="7"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_column="1"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:background="@color/colorPrimary"
            android:textSize="20sp"
            android:id="@+id/seven"
            android:onClick="seven"
            android:layout_weight="50"/>
    <Button
            android:text="8"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="2"
            android:textSize="20dp"
            android:background="@color/colorPrimary"
            android:id="@+id/eight"
            android:onClick="eight"
            android:layout_weight="50"/>
    <Button
            android:text="9"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="3"
            android:textSize="20dp"
            android:background="@color/colorPrimary"
            android:id="@+id/nine"
            android:onClick="nine"
            android:layout_weight="50"/>
    <Button
```

```
            android:text="X"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="4"
            android:textSize="20dp"
            android:background="@color/colorPrimary"
            android:id="@+id/multi"
            android:onClick="mul"
            android:layout_weight="50"/>
    </TableRow>
</TableLayout>
<TableLayout
        android:layout_width="match_parent"
        android:layout_height="90dp"
        android:layout_marginTop="2dp"
        android:layout_marginRight="3dp">
    <TableRow
            android:layout_width="match_parent"
            android:layout_height="90dp">

        <Button
            android:text="4"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_column="1"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:background="@color/colorPrimary"
            android:textSize="20dp"
            android:id="@+id/four"
            android:onClick="four"
            android:layout_weight="50"/>
        <Button
            android:text="5"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="2"
            android:textSize="20sp"
            android:background="@color/colorPrimary"
            android:id="@+id/five"
            android:onClick="five"
            android:layout_weight="50"/>
        <Button
            android:text="6"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="3"
            android:textSize="20dp"
            android:background="@color/colorPrimary"
            android:id="@+id/six"
            android:onClick="six"
            android:layout_weight="50"/>
        <Button
            android:text="--"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="4"
            android:textSize="20dp"
            android:background="@color/colorPrimary"
            android:id="@+id/min"
            android:onClick="sub"
            android:layout_weight="50"/>
    </TableRow>
</TableLayout>
<TableLayout
        android:layout_width="match_parent"
        android:layout_height="90dp"
        android:layout_marginTop="2dp"
        android:layout_marginRight="3dp">
    <TableRow
          android:layout_width="match_parent"
          android:layout_height="90dp">
```

```
<Button
            android:text="1"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_column="1"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:background="@color/colorPrimary"
            android:textSize="20dp"
            android:id="@+id/one"
            android:onClick="one"
            android:layout_weight="50"/>
     <Button
            android:text="2"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="2"
            android:textSize="20dp"
            android:background="@color/colorPrimary"
            android:id="@+id/two"
            android:onClick="two"
            android:layout_weight="50"/>
     <Button
            android:text="3"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="3"
            android:textSize="20sp"
            android:background="@color/colorPrimary"
            android:id="@+id/three"
            android:onClick="three"
            android:layout_weight="50"/>
     <Button
            android:text="+"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="4"
            android:textSize="20sp"
            android:background="@color/colorPrimary"
            android:id="@+id/plus"
            android:onClick="add"
            android:layout_weight="50"/>
    </TableRow>
</TableLayout>
<TableLayout
        android:layout_width="match_parent"
        android:layout_height="90dp"
        android:layout_marginTop="2dp"
        android:layout_marginRight="3dp">
    <TableRow
            android:layout_width="match_parent"
            android:layout_height="90dp">
        <Button
            android:text="-"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_column="1"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:background="@color/colorPrimary"
            android:textSize="20sp"
            android:id="@+id/min_plus"
            android:onClick="min"
            android:layout_weight="50"/>
        <Button
            android:text="0"
            android:layout_width="95dp"
            android:layout_height="79dp"
            android:layout_marginStart="5dp"
            android:layout_marginLeft="5dp"
            android:layout_column="2"
            android:textSize="20sp"
            android:background="@color/colorPrimary"
            android:id="@+id/zero"
            android:onClick="zero"
            android:layout_weight="50"/>
            <Button
                android:text="."
                android:layout_width="95dp"
                android:layout_height="79dp"
                android:layout_marginStart="5dp"
                android:layout_marginLeft="5dp"
                android:layout_column="3"
                android:textSize="20sp"
                android:background="@color/colorPrimary"
                android:id="@+id/dot"
                android:onClick="pt"
                android:layout_weight="50"/>
            <Button
                android:text="="
                android:layout_width="95dp"
                android:layout_height="79dp"
                android:layout_marginStart="5dp"
                android:layout_marginLeft="5dp"
                android:layout_column="4"
                android:textSize="20sp"
                android:background="@color/colorPrimary"
                android:id="@+id/equal"
                android:onClick="equal"
                android:layout_weight="50"/>
        </TableRow>
    </TableLayout>
</TableLayout>
```

MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    var eq:String=""
    var pre_eq:String=""
    var curr_eq:String=""
    var op=0
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        pre_data.text=""
    }
    fun ce(view: View)
    {
```

```
    curr_data.text=""
    curr_eq=""
}
fun clear(view:View)
{
    curr_data.text=""
    pre_data.text=""
    curr_eq=""
    pre_eq=""
    op=0
}
fun remove(view:View)
{
    try {
        curr_eq = curr_eq.substring(0, curr_eq.length - 1)
        curr_data.text = curr_eq
    }
    catch (e:Exception)
    {
    }
}
fun add(view:View)
{
    Log.d("+",op.toString())
    if(op==0) {
        pre_eq = curr_eq
        pre_data.text=curr_eq+"+"
        curr_eq = ""
        curr_data.text = ""
    }
    else
    {
        if(op==1) {
            try {
                curr_eq = (pre_eq.toFloat()+curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"+"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
        {
        }
    }
    if(op==2) {
        try {
            curr_eq = ( pre_eq.toFloat()-curr_eq.toFloat()).toString()
            pre_data.text=curr_eq+"+"
            curr_data.text=""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
        }
    }
    if(op==3) {
        try {
            curr_eq = ( pre_eq.toFloat()*curr_eq.toFloat() ).toString()
            pre_data.text = curr_eq + "+"
            curr_data.text = ""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
        }
    }
    if(op==4) {
        try {
            curr_eq = ( pre_eq.toFloat()/curr_eq.toFloat()).toString()
            pre_data.text = curr_eq + "+"
            curr_data.text = ""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
        }
    }
}
op = 1
```

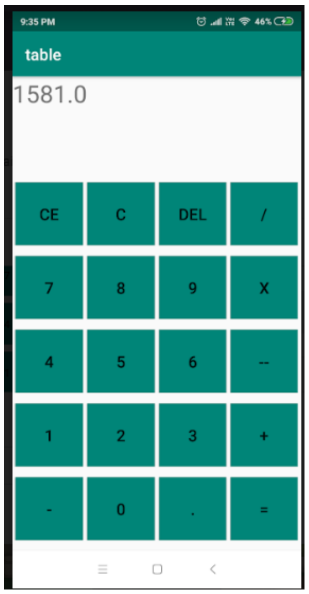
```
}
fun sub(view:View)
{
    if(op==0) {
        pre_eq = curr_eq
        pre_data.text=pre_eq+"-"
        curr_eq = ""
        curr_data.text = ""
    }
    else
    {
        if(op==1) {
            try {
                curr_eq = (pre_eq.toFloat()+curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"-"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==2) {
            try {
                curr_eq = ( pre_eq.toFloat()-curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"-"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==3) {
            try {
                curr_eq = ( pre_eq.toFloat()*curr_eq.toFloat() ).toString()
                pre_data.text = curr_eq + "-"
                curr_data.text = ""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==4) {
            try {
                curr_eq = ( pre_eq.toFloat()/curr_eq.toFloat()).toString()
                pre_data.text = curr_eq + "-"
                curr_data.text = ""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
    }
    op = 2
}
fun mul(view:View)
{
    if(op==0) {
        pre_eq = curr_eq
        pre_data.text=pre_eq+"x"
        curr_eq = ""
        curr_data.text = ""
    }
    else
    {
        if(op==1) {
            try {
                curr_eq = (pre_eq.toFloat()+curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"*"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==2) {
            try {
                curr_eq = ( pre_eq.toFloat()-curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"*"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==3) {
            try {
                curr_eq = ( pre_eq.toFloat()*curr_eq.toFloat() ).toString()
                pre_data.text = curr_eq + "*"
                curr_data.text = ""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==4) {
            try {
                curr_eq = ( pre_eq.toFloat()/curr_eq.toFloat()).toString()
                pre_data.text = curr_eq + "*"
                curr_data.text = ""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
    }
    op=3
}
```

```
fun divide(view:View)
{
    if(op==0) {
        pre_eq = curr_eq
        pre_data.text=pre_eq+"/"
        curr_eq = ""
        curr_data.text = ""
    }
    else
    {
        if(op==1) {
            try {
                curr_eq = (pre_eq.toFloat()+curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"/"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==2) {
            try {
                curr_eq = ( pre_eq.toFloat()-curr_eq.toFloat()).toString()
                pre_data.text=curr_eq+"/"
                curr_data.text=""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==3) {
            try {
                curr_eq = ( pre_eq.toFloat()*curr_eq.toFloat() ).toString()
                pre_data.text = curr_eq + "/"
                curr_data.text = ""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
        if(op==4) {
            try {
                curr_eq = ( pre_eq.toFloat()/curr_eq.toFloat()).toString()
                pre_data.text = curr_eq + "/"
                curr_data.text = ""
                pre_eq = curr_eq
                curr_eq = ""
            }
            catch (e:Exception)
            {
            }
        }
    }
    op=4
}
fun min(view:View)
{
    curr_eq=curr_eq+"-"
    curr_data.text=curr_eq
}
fun equal(view:View)
{
    if(op==1) {
        try {
            curr_eq = (pre_eq.toFloat()+curr_eq.toFloat()).toString()
            pre_data.text=curr_eq
            curr_data.text=""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
            e.printStackTrace()
        }
    }
    if(op==2) {
        try {
            curr_eq = ( pre_eq.toFloat()-curr_eq.toFloat()).toString()
            pre_data.text=curr_eq
            curr_data.text=""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
        }
    }
    if(op==3) {
        try {
            curr_eq = ( pre_eq.toFloat()*curr_eq.toFloat() ).toString()
            pre_data.text = curr_eq
            curr_data.text = ""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
        }
    }
    if(op==4) {
        try {
            curr_eq = ( pre_eq.toFloat()/curr_eq.toFloat()).toString()
            pre_data.text = curr_eq
            curr_data.text = ""
            pre_eq = curr_eq
            curr_eq = ""
        }
        catch (e:Exception)
        {
        }
    }
}
fun pt(view:View)
{
    curr_eq=curr_eq+"."
    curr_data.text=curr_eq
}
```

```
fun zero(view:View)
{
   curr_eq=curr_eq+"0"
   curr_data.text=curr_eq
}
fun one(view:View)
{
   curr_eq=curr_eq+"1"
   curr_data.text=curr_eq
}
fun two(view:View)
{
   curr_eq=curr_eq+"2"
   curr_data.text=curr_eq
}
fun three(view:View)
{
   curr_eq=curr_eq+"3"
   curr_data.text=curr_eq
}
fun  four(view:View)
{
   curr_eq=curr_eq+"4"
   curr_data.text=curr_eq
}
fun five(view:View)
{
   curr_eq=curr_eq+"5"
   curr_data.text=curr_eq
}
fun six(view:View)
{
   curr_eq=curr_eq+"6"
   curr_data.text=curr_eq
}
fun seven(view: View)
{
   curr_eq=curr_eq+"7"
   curr_data.text=curr_eq
}
fun eight(view:View)
```

```
    {
        curr_eq=curr_eq+"8"
        curr_data.text=curr_eq
    }
    fun nine(view:View)
    {
        curr_eq=curr_eq+"9"
        curr_data.text=curr_eq
    }
}
```

Output:



```
1581.0
```

GridLayout

Android **GridView** shows items in two-dimensional scrolling grid (rows & columns) and the grid items are not necessarily predetermined but they automatically inserted to the layout using a **ListAdapter.**

XML attributes :

1.android:rowCount: The maximum number of rows to create when

2.automatically positioning children or the no of child views going to fit ina row of the layout.

3.android:columnCount: The maximum number of columns to create when automatically positioning children.

4.android:alignmentMode: When set to alignMargins, causes alignment to take place between the outer boundary of a view, as defined by its margins

5.android:columnOrderPreserved – When set to true, forces column boundaries to appear in the same order as column indices.

6.android:rowOrderPreserved – When set to true, forces row boundaries to appear in the same order as row indices.

7.android:useDefaultMargins – When set to true, tells GridLayout to use default margins when none are specified in a view's layout parameters.We may use the weight property like in a linear layout to perfectly align the views

8.android:layout_weight: divide the layout into a given section orweights.

activity_main.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="10"
    android:background="@drawable/ic_launcher_foreground"
    tools:context=".MainActivity">

  <GridLayout
      android:id="@+id/grid"
      android:rowCount="3"
      android:columnCount="2"
      android:alignmentMode="alignMargins"
      android:columnOrderPreserved="false"
      android:layout_weight="8"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_marginTop="65dp"
    android:padding="14dp">

<android.support.v7.widget.CardView
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
        android:layout_marginBottom="16dp"
        android:layout_rowWeight="1"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        app:cardElevation="8dp"
        app:cardCornerRadius="8dp"
        android:id="@+id/fb">

    <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">
        <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textColor="#000"
                android:textSize="18sp"
                android:textStyle="bold"
                android:textAlignment="center"
                android:text="Facebook"/>

    </LinearLayout>

</android.support.v7.widget.CardView>

<android.support.v7.widget.CardView
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
        android:layout_marginBottom="16dp"
        android:layout_rowWeight="1"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        app:cardElevation="8dp"
        app:cardCornerRadius="8dp"
        android:id="@+id/google">

    <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">
        <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Google"
                android:textColor="#000"
                android:textSize="18sp"
                android:textStyle="bold"
                android:textAlignment="center"/>

    </LinearLayout>

</android.support.v7.widget.CardView>

<android.support.v7.widget.CardView
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
        android:layout_marginBottom="16dp"
        android:layout_rowWeight="1"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        app:cardElevation="8dp"
        app:cardCornerRadius="8dp"
        android:id="@+id/twitter">

    <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">
        <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Twitter"
                android:textColor="#000"
                android:textSize="18sp"
                android:textStyle="bold"
                android:textAlignment="center"/>

    </LinearLayout>

</android.support.v7.widget.CardView>

<android.support.v7.widget.CardView
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
        android:layout_marginBottom="16dp"
        android:layout_rowWeight="1"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        app:cardElevation="8dp"
        app:cardCornerRadius="8dp"
        android:id="@+id/linkd">

    <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">
        <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Linkedin"
                android:textColor="#000"
                android:textSize="18sp"
                android:textStyle="bold"
                android:textAlignment="center"/>

        </LinearLayout>

    </android.support.v7.widget.CardView>

    <android.support.v7.widget.CardView
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
        android:layout_marginBottom="16dp"
        android:layout_rowWeight="1"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        app:cardElevation="8dp"
        app:cardCornerRadius="8dp"
        android:id="@+id/insta">

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Instagram"
                android:textColor="#000"
                android:textSize="18sp"
                android:textStyle="bold"
                android:textAlignment="center"/>

        </LinearLayout>

    </android.support.v7.widget.CardView>

    <android.support.v7.widget.CardView
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
```

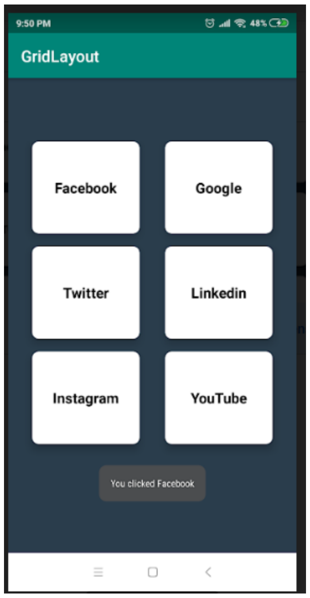
```
            android:layout_marginBottom="16dp"
            android:layout_rowWeight="1"
            android:layout_marginLeft="16dp"
            android:layout_marginRight="16dp"
            app:cardElevation="8dp"
            app:cardCornerRadius="8dp"
            android:id="@+id/youtube">

        <LinearLayout
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:layout_gravity="center_horizontal|center_vertical"
              android:layout_margin="16dp"
              android:orientation="vertical">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="YouTube"
                android:textColor="#000"
                android:textSize="18sp"
                android:textStyle="bold"
                android:textAlignment="center"/>

        </LinearLayout>

      </android.support.v7.widget.CardView>
   </GridLayout>
</LinearLayout>
```

MainActivity.kt

```
class MainActivity : AppCompatActivity() {
   override fun onCreate(savedInstanceState: Bundle?) {
      super.onCreate(savedInstanceState)
      setContentView(R.layout.activity_main)
      fb.setOnClickListener(click)
      google.setOnClickListener(click)
      twitter.setOnClickListener(click)
      insta.setOnClickListener(click)
      linkd.setOnClickListener(click)
      youtube.setOnClickListener(click)
   }
```

```
    val click=View.OnClickListener {v: View? ->
        var txt=""
        when(v!!.id)
        {
            R.id.fb->txt="Facebook"
            R.id.google->txt="Google"
            R.id.twitter->txt="Twitter"
            R.id.insta->txt="Instagram"
            R.id.linkd->txt="Linkedin"
            R.id.youtube->txt="YouTube"
        }
        Toast.makeText(this,"You clicked "+txt,Toast.LENGTH_SHORT).show()
    }
}
```

Output:



You clicked Facebook

PRACTICAL 5

Programming UI elements

Design App With UI:

MainActivity.kt:

```
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)
        }

    }
}
```

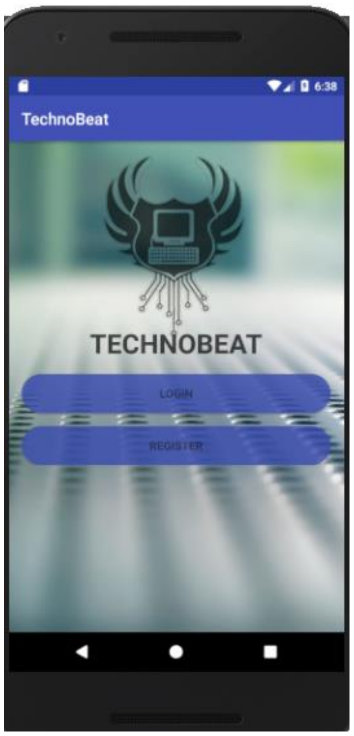
activity_main.xml

```
<?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:



AppBar

The app bar, also known as the action bar, is one of the most important design elements in your app's activities, because it provides a visual structure and interactive elements that are familiar to users. Using the app bar makes your app consistent with other Android apps, allowing users to quickly understand how to operate your app and have a great experience. The key functions of the app bar are as follows:

A dedicated space for giving your app an identity and indicating the user's location in the app.

Access to important actions in a predictable way, such as search.

Support for navigation and view switching (with tabs or drop-down lists).

Style.xml

In Apptheme change the parent theme to NoActionBar as given Below

```
<resources>

    <!--Base application theme. -->

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

       <item name="colorPrimary">@color/colorPrimary</item>

       <item name="colorPrimaryDark">@color/topbarcolor</item>
   </style>

   <style name="customtheme" parent="Theme.AppCompat.Light">

       <item name="android:background">@color/appbarcolor</item>

   </style>

</resources>
```

Colors.xml

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

<resources>

    <color name="colorPrimary">#008577</color>

    <color name="colorPrimaryDark">#00574B</color>

    <color name="colorAccent">#D81B60</color>

    <color name="appbarcolor">#6200ee</color>

    <color name="topbarcolor">#4c00d5</color>

</resources>
```

Here the search icon and folder icon are taken from internet you can download any icon and save the icon to drawable folder

Activity_main.xml

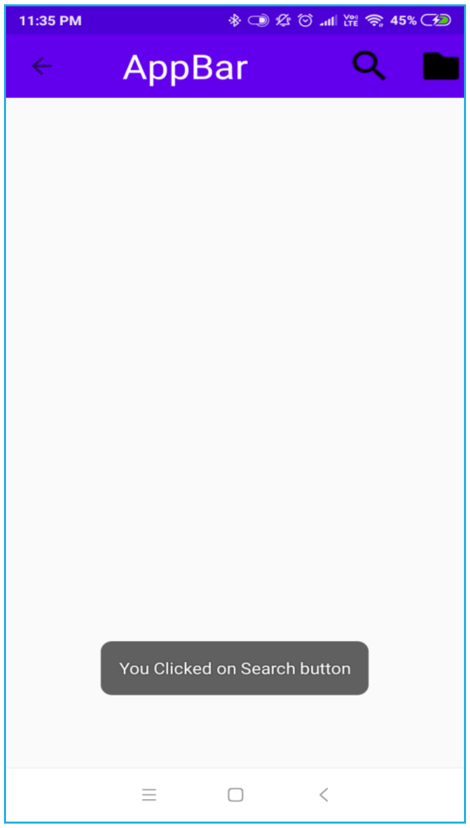
```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

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

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

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

        android:layout_width="match_parent"

        android:layout_height="match_parent"

        tools:context=".MainActivity">
```

```
<android.support.v7.widget.Toolbar
        android:id="@+id/custtoolbar"
        android:layout_width="match_parent"
        android:layout_height="?android:attr/actionBarSize"
        android:theme="@style/customtheme"
>
    <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="AppBar"
            android:textSize="30sp"
            android:textColor="#ffff"/>
    <ImageButton
            android:id="@+id/search"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:background="@android:color/transparent"
            android:src="@drawable/search"
            android:layout_marginLeft="90dp"
    />
    <ImageButton
            android:id="@+id/folder"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:background="@android:color/transparent"
            android:src="@drawable/folder"
            android:layout_marginLeft="20dp"
        />
    </android.support.v7.widget.Toolbar>

</RelativeLayout>
```

MainActivity.kt

```
package com.example.appbar

import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        setSupportActionBar(custtoolbar)
        supportActionBar!!.setDisplayHomeAsUpEnabled(true)
        supportActionBar!!.setHomeButtonEnabled(true)
        search.setOnClickListener {
            Toast.makeText(applicationContext,"You Clicked on Search
button",Toast.LENGTH_SHORT).show()

        }

        folder.setOnClickListener {

            Toast.makeText(applicationContext,"You Clicked on folder
button",Toast.LENGTH_SHORT).show()


        }

    }

}
```

Output:



You Clicked on Search button

PRACTICAL 6

Programming menus, dialog, dialog fragments

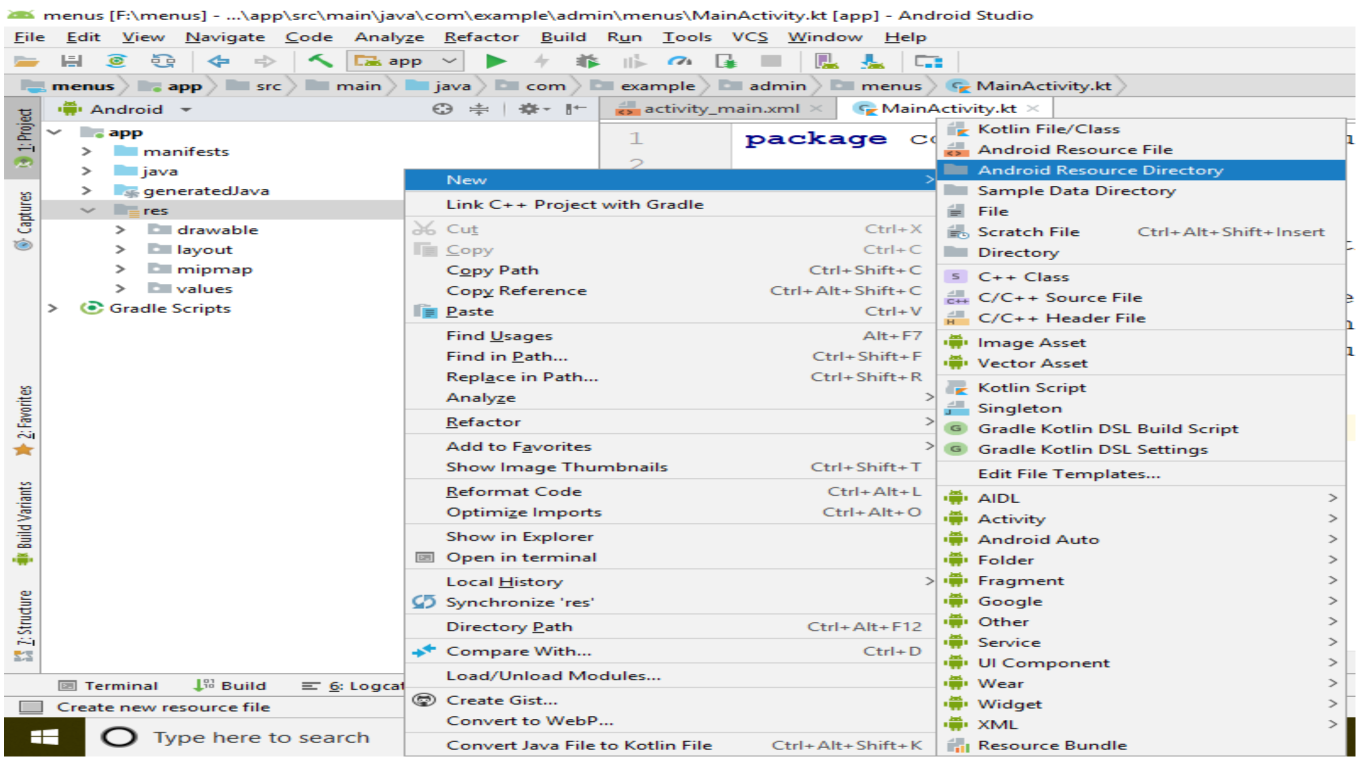
Menu

Example :

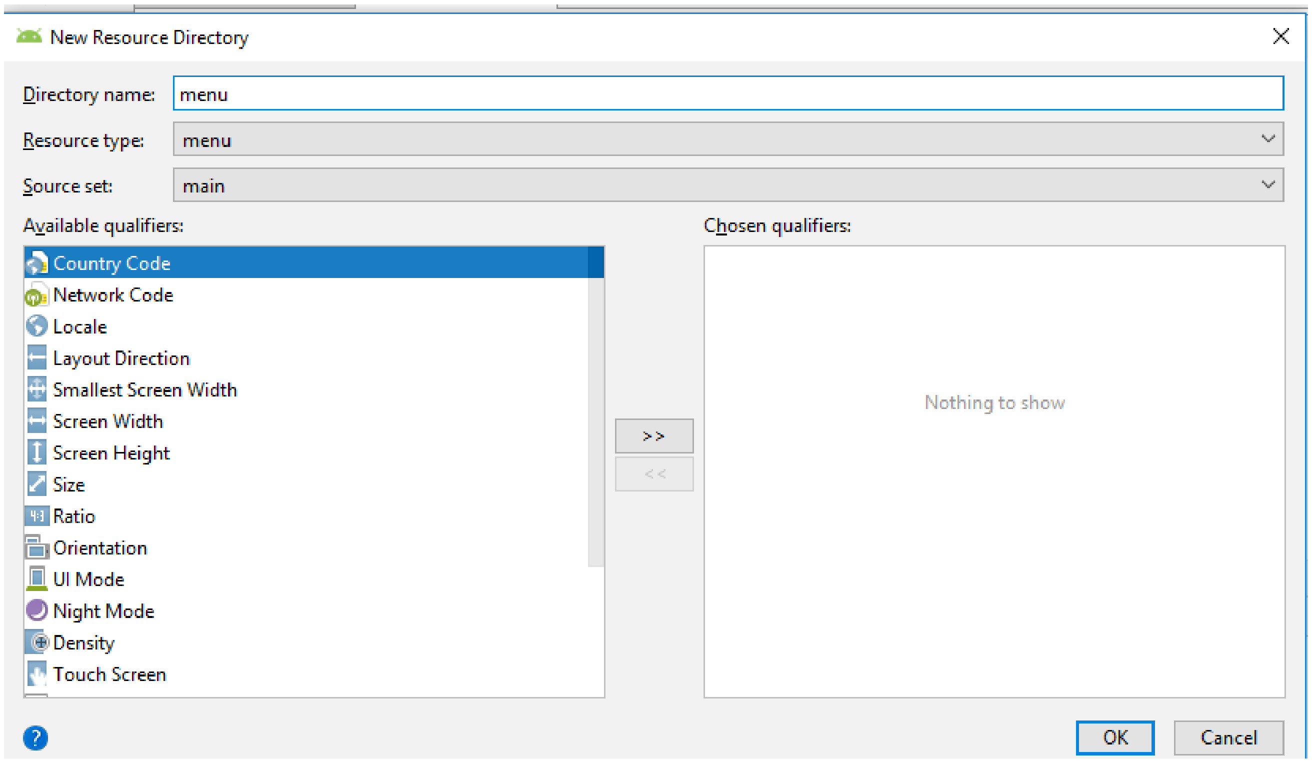
In Android, Options Menu is a primary collection of menu items for activity and it is useful to implement actions that have a global impact on the app, such as Settings, Search, etc.

For all menu types, Android provides a standard XML format to define menu items. Instead of building a menu in your activity's code, you should define a menu and all its items in an XML menu resource. You can then inflate the menu resource (load it as a Menu object) in your activity or fragment. To define the menu, create an XML file inside your project's res/menu/ directory

Right click on res directory then click new > Android Resource Directory



Change the Resource type form value to menu



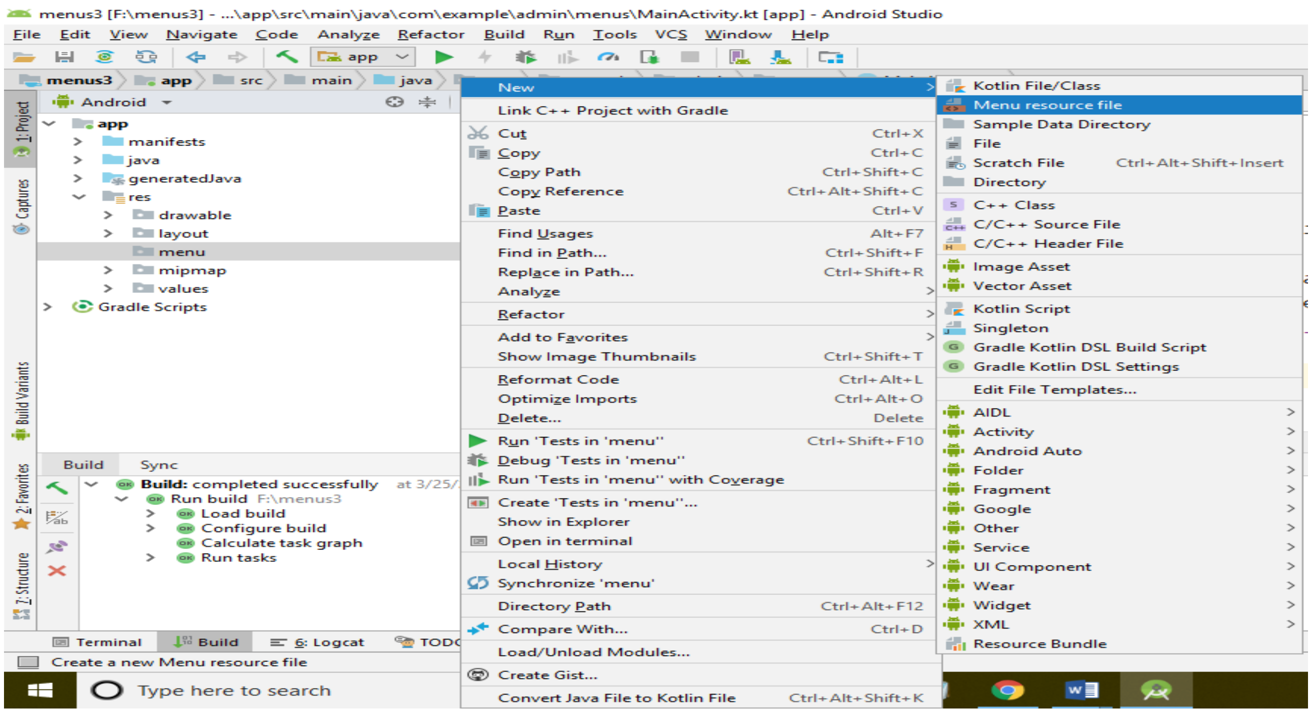
New Resource Directory

Directory name: menu

Resource type: menu

Source set: main

Then Right click on menu directory then new>Menu resource file

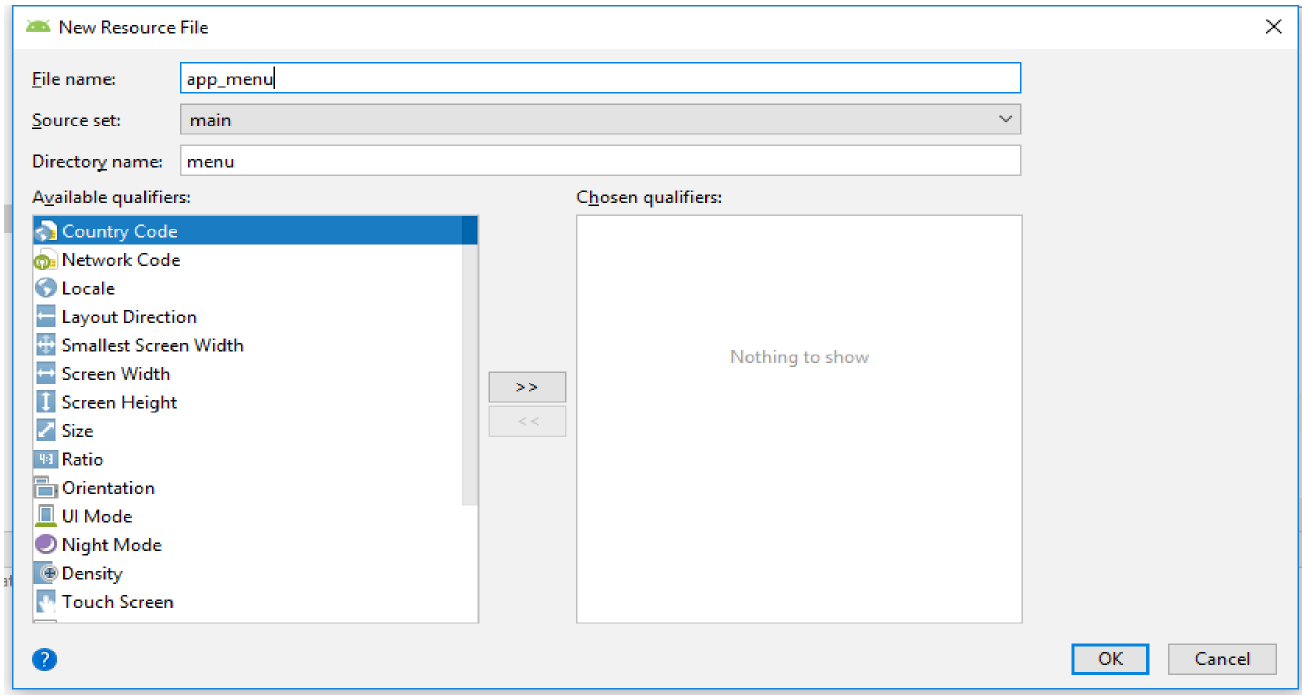


menus3 [F:\menus3] - ...\app\src\main\java\com\example\admin\menus\MainActivity.kt [app] - Android Studio

Build: completed successfully at 3/25/
Run build F:\menus3
Load build
Configure build
Calculate task graph
Run tasks

Create a new Menu resource file

Give the file name



New Resource File

File name: app_menu

Source set: main

Directory name: menu

Available qualifiers:

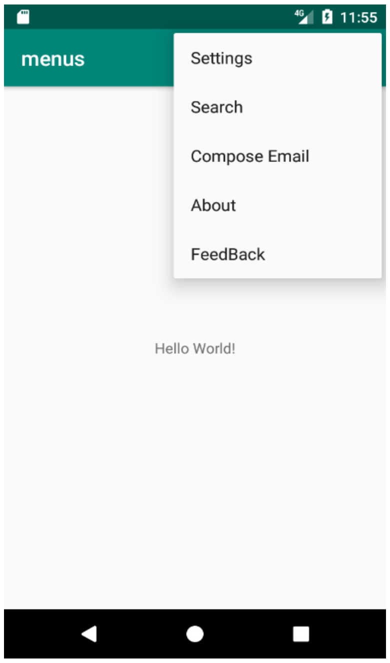
Chosen qualifiers:

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

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

    <item android:id="@+id/settings" android:title="Settings"/>

    <item android:id="@+id/search" android:title="Search"/>

    <item android:id="@+id/email" android:title="Compose Email"/>

    <item android:id="@+id/about" android:title="About"/>

    <item android:id="@+id/feedback" android:title="FeedBack"/>

</menu>
```

MainActivity.kt

```
package com.example.admin.menus

import android.support.v7.app.AppCompatActivity

import android.os.Bundle

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 {
        var inflater=menuInflater
        inflater.inflate(R.menu.app_menu,menu)
        return true
    }
    override fun onOptionsItemSelected(item: MenuItem?): Boolean {
        when (item!!.itemId) {
            R.id.settings -> {
                Toast.makeText(this, "you clicked on Settings", Toast.LENGTH_SHORT).show()
            }
            R.id.search -> {
                Toast.makeText(this, "you clicked on Search", Toast.LENGTH_SHORT).show()
            }
            R.id.email -> {
                Toast.makeText(this, "you clicked on Compose Email",
Toast.LENGTH_SHORT).show()
            }
            R.id.about -> {
                Toast.makeText(this, "you clicked on About", Toast.LENGTH_SHORT).show()
            }
            R.id.feedback -> {
                Toast.makeText(this, "you clicked on FeedBack", Toast.LENGTH_SHORT).show()
            }
        }
        return super.onOptionsItemSelected(item)
    }
}
```

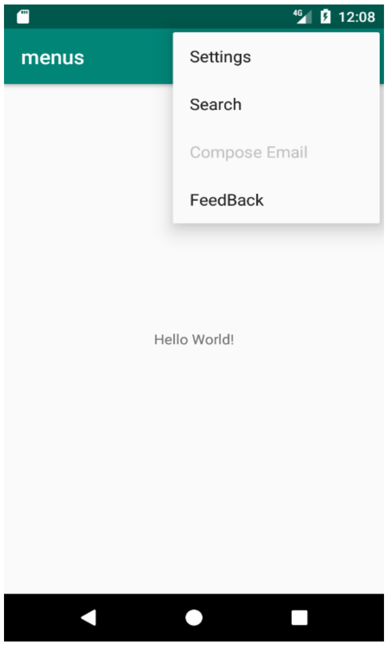
Output :



Example : To Remove and disable the item from menu item list, add the below code after onCreateOptionsMenu()

```
override fun onPrepareOptionsMenu(menu: Menu?): Boolean {
        //To  Disable the 3rd Menu Item ( Compose Email )
         menu?.getItem(2)?.setEnabled(false)


        //To remove  the 4th Menu Item ( About )
            menu?.getItem(3)?.isVisible = false
        return super.onPrepareOptionsMenu(menu)
    }
```

Output:



Dialog

A dialog is a small window that prompts the user to make a decision or enter additional information. A dialog does not fill the screen and is normally used for modal events that require users to take an action before they can proceed.

Example :

Acitivity_main.xml

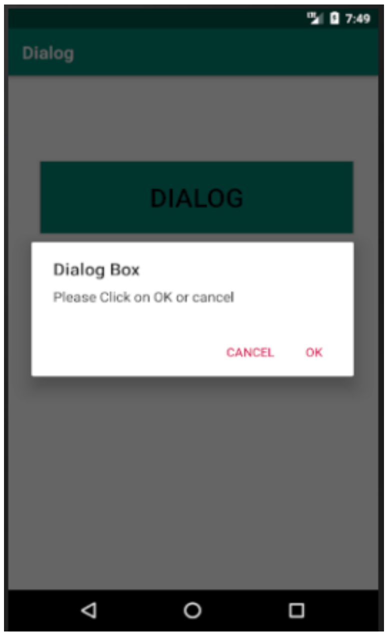
```
<Button
        android:text="Dialog"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/dialog"
        android:textSize="30sp"
        android:background="@color/colorPrimary"
        android:onClick="dialog"
```

```
            android:layout_alignParentStart="true"

            android:layout_marginTop="210dp"

            android:layout_marginStart="135dp"

            android:layout_alignParentTop="true"/>
```

mainActivity.kt

```
class MainActivity : AppCompatActivity() {


    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

    }

    fun dialog(view: View) {

        var alert= AlertDialog.Builder(this@MainActivity)

        alert.setTitle("Dialog Box")

        alert.setMessage("Please Click on OK or cancel")

        alert.setPositiveButton("OK"){

                dialog, which ->

            Toast.makeText(applicationContext,"You on Clicked Ok
button",Toast.LENGTH_SHORT).show()

        }

        alert.setNegativeButton("CANCEL"){

                dialog, which ->

            Toast.makeText(applicationContext,"You on Clicked Cancel
button",Toast.LENGTH_SHORT).show()

        }

        alert.show()
    }

}
```

Output:



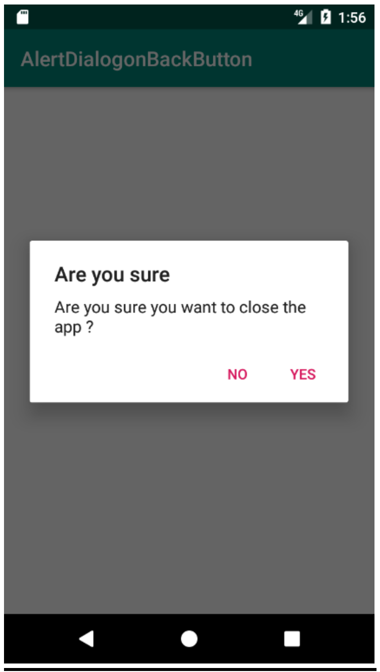
Dialog Box

Please Click on OK or cancel

Example : Alertdialog on Back Button Pressed event

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
    override fun onBackPressed()     {
        val builder = AlertDialog.Builder(this)
        builder.setTitle("Are you sure")
        builder.setMessage("Are you sure you want to close the app ?")
        builder.setPositiveButton("Yes",   {dialogInterface, i -> finish()})
        builder.setNegativeButton("No",{ dialogInterface, i ->  })
        builder.show()
    }
}
```

Output:



Are you sure

Are you sure you want to close the app ?

Practical No: 7

Programs on Intents,Event Listeners and Adapters

An intent is an abstract description of an operation to be performed. It is a way of ‘Message passing between two or more than two components of android’.

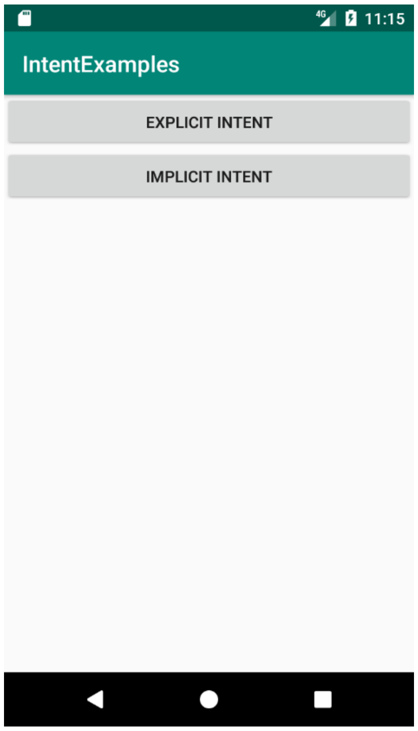
Example : Demonstration of explicit and implicit intents

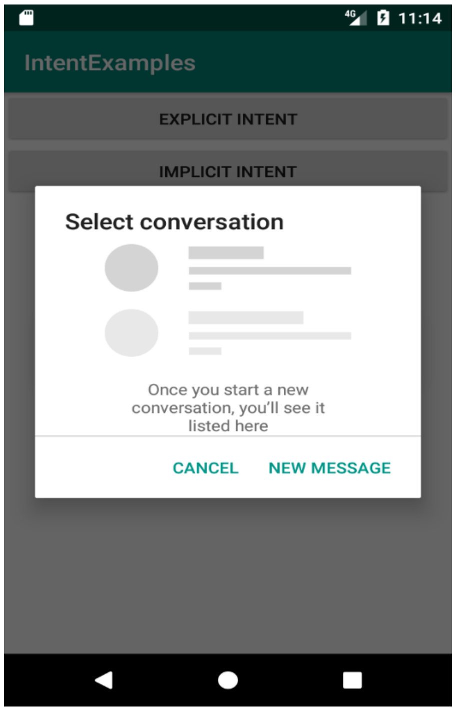
activity_main.xml

```
<Button
        android:id="@+id/btn_explicit"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Explicit Intent"/>
  <Button
        android:id="@+id/btn_implicit"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Implicit Intent"
        android:layout_below="@+id/btn_explicit"/>
```

Main_activity.kt

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val context = this
        btn_explicit.setOnClickListener {
            val intent = Intent(context,MainActivity::class.java)
            startActivity(intent)
            finish()
        }
        btn_implicit.setOnClickListener {
            var intent = Intent(Intent.ACTION_SEND)
            intent.putExtra(Intent.EXTRA_TEXT,"Hello...")
            intent.type = "text/plain"
            startActivity(intent)
        }
    }
}
```
Output





Events and Listeners

Events are a useful way to collect data about a user's interaction with interactive components of Applications.Like button presses or screen touch etc. The Android framework maintains an event queue as first-in, first-out (FIFO) basis. You can capture these events in your program and take appropriate action as per requirements.

Event Listeners − An event listener is an interface in the View class that contains a single callback method. These methods will be called by the Android framework when the View to which the listener has been registered is triggered by user interaction with the item in the UI.

onClickListener – Used to detect click style events whereby the user touches and then releases an area of the device display occupied by a view. Corresponds to the onClick() callback method which is passed a reference to the view that received the event as an argument.

onLongClickListener – Used to detect when the user maintains the touch over a view for an extended period. Corresponds to the onLongClick() callback method which is passed as an argument the view that received the event.

onTouchListener – Used to detect any form of contact with the touch screen including individual or multiple touches and gesture motions. Corresponding with the onTouch() callback, this topic will be covered in greater detail in the chapter entitled "Android Touch and

Multi-touch Event Handling”. The callback method is passed as arguments the view that received the event and a MotionEvent object.
onCreateContextMenuListener – Listens for the creation of a context menu as the result of a long click. Corresponds to the onCreateContextMenu() callback method. The callback is passed the menu, the view that received the event and a menu context object.
onFocusChangeListener – Detects when focus moves away from the current view as the result of interaction with a track-ball or navigation key. Corresponds to the onFocusChange() callback method which is passed the view that received the event and a Boolean value to indicate whether focus was gained or lost.
onKeyListener – Used to detect when a key on a device is pressed while a view has focus. Corresponds to the onKey() callback method. Passed as arguments are the view that received the event, the KeyCode of the physical key that was pressed and a KeyEvent object.
Example:
Activity_main.xml

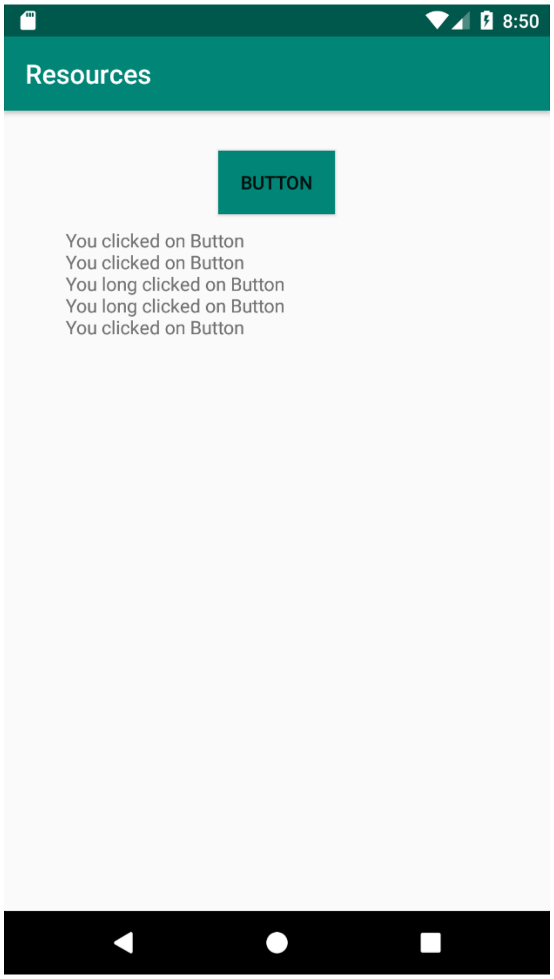
```
<RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity"
>
    <Button
            android:text="Button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/btn"
            android:layout_centerHorizontal="true"
            android:background="@color/colorPrimary"
            android:layout_marginTop="30dp"/>
    <TextView
            android:text=""
            android:layout_width="318dp"
            android:layout_height="294dp"
            android:layout_below="@+id/btn"
            android:layout_centerHorizontal="true"
            android:layout_marginTop="10dp"
            android:scrollbars="vertical"
            android:id="@+id/textv"/>
```

```
</RelativeLayout>
```

MainActivity.xml

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        btn.setOnClickListener {

            textv.text = "You clicked on Button\n" + textv.text.toString()
        }
        btn.setOnLongClickListener{
            textv.text = "You long clicked on Button\n" + textv.text.toString()
            true
        }
    }
}
```

Output:



```
You clicked on Button
You clicked on Button
You long clicked on Button
You long clicked on Button
You clicked on Button
```

Adapter

an **adapter** in **Android** carries the data from a source (e.g. ArrayList<>) and delivers it to a layout (.xml file). **Adapters** in **Android** are a bridge between the **adapter** view (e.g. ListView) and the underlying data for that view.

Example 1: Display the static data in list view using adapter

activity_main.xml

```
<ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/list"/>
```

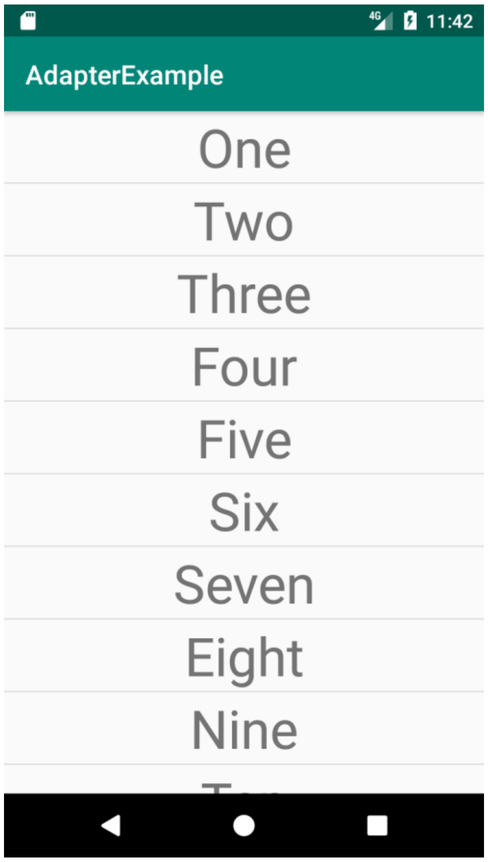
Mainactivity.kt

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

        var items=
arrayOf("One","Two","Three","Four","Five","Six","Seven","Eight","Nine","Ten")

        var arrayadapter= ArrayAdapter(this,R.layout.list_row,R.id.item_text,items);

        list.adapter=arrayadapter

    }

}
```

List_row.xml

```
<TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Hello"
        android:textSize="40sp"
        android:textAlignment="center"
        android:id="@+id/item_text"/>
```

Output:



One
Two
Three
Four
Five
Six
Seven
Eight
Nine

Practical No:8

Programs on Services,Notification and Broadcast Receivers

Services

A Service is an application component that can perform long-running operations in the background, and it doesn't provide a user interface.

Example : Creating a background service with notification

```
<RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">
    <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:hint="Enter the msg"
            android:textSize="30dp"
            android:ems="10"
            android:id="@+id/msg"/>
    <Button
            android:id="@+id/start"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Start Service"
            android:layout_below="@+id/msg"
            android:layout_marginTop="20dp"
            android:layout_centerHorizontal="true"
            android:background="@color/colorPrimary"
            android:onClick="startservice" />
    <Button
            android:id="@+id/stop"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Stop Service"
    android:layout_below="@+id/start"
    android:layout_marginTop="20dp"
    android:layout_centerHorizontal="true"
    android:background="@color/colorPrimary"
    android:onClick="stopservice"
    />

</RelativeLayout>
```

Create Myservice.kt(class) file

Myservice.kt

```
package com.example.admin.serviceexample

import android.app.PendingIntent
import android.app.Service
import android.content.Intent
import android.os.IBinder
import android.support.v4.app.NotificationCompat

class Myservice: Service() {
    public var CHANNEL_ID="myNotification"
    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        var msg:String?=intent?.getStringExtra("msg")
        val intent=Intent(this,MainActivity::class.java)
        val pendingIntent= PendingIntent.getActivity(this,0,intent,0)
        var notif = NotificationCompat.Builder(this, CHANNEL_ID)
            .setSmallIcon(R.drawable.notification_icon_background)
            .setContentTitle("The Background Service")
            .setContentText(msg)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)
            .setContentIntent(pendingIntent)
            .build()
        startForeground(1,notif)
        return START_NOT_STICKY
    }
    override fun onBind(intent: Intent?): IBinder? {
        return null
    }
    override fun onCreate() {
        super.onCreate()
    }

    override fun onDestroy() {
        super.onDestroy()
    }
}
```

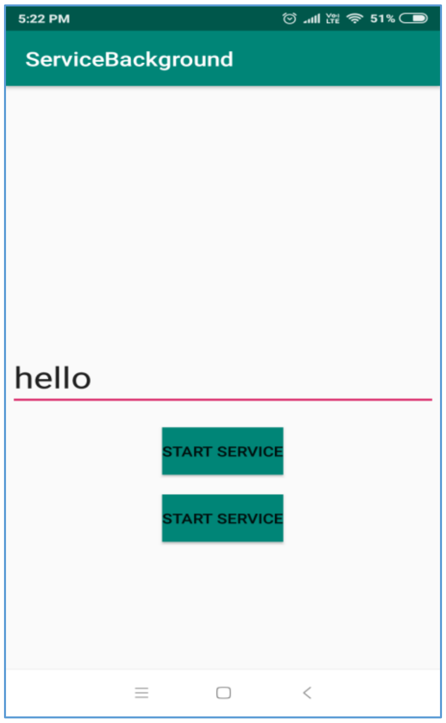
Add the service in androidmanifest.xml file in the application tag

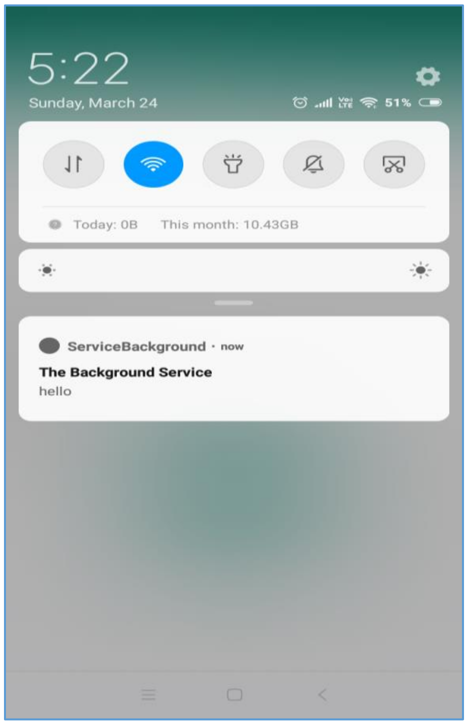
```
<service android:name=".Myservice" android:enabled="true"/>
```

MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    public var CHANNEL_ID = "myNotification"
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        registerNotificationChannel()
    }

    fun startservice(view: View) {
        var intent = Intent(this, Myservice::class.java)
        intent.putExtra("msg", msg.text.toString())
        startService(intent)
    }

    fun stopservice(view: View) {
        var intent = Intent(this, Myservice::class.java)
        stopService(intent)
    }

    fun registerNotificationChannel() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            var channel = NotificationChannel(CHANNEL_ID, "notification",
NotificationManager.IMPORTANCE_DEFAULT)
            var manage = getSystemService(Context.NOTIFICATION_SERVICE) as
NotificationManager
            manage.createNotificationChannel(channel)
        }
    }
}
```

Output:





ServiceBackground · now
The Background Service
hello

Notifications

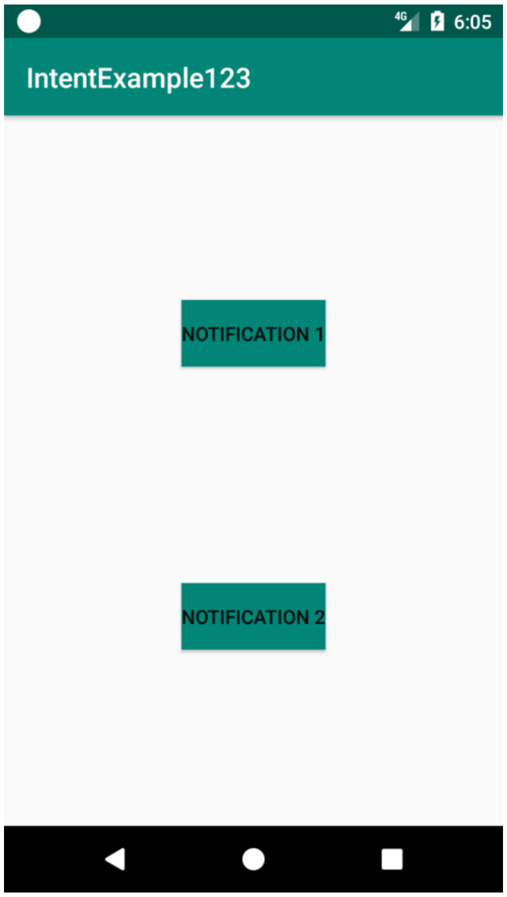
A notification is a message that Android displays outside your app's UI to provide the user with reminders, communication from other people, or other timely information from your app. Users can tap the notification to open your app or take an action directly from the notification.

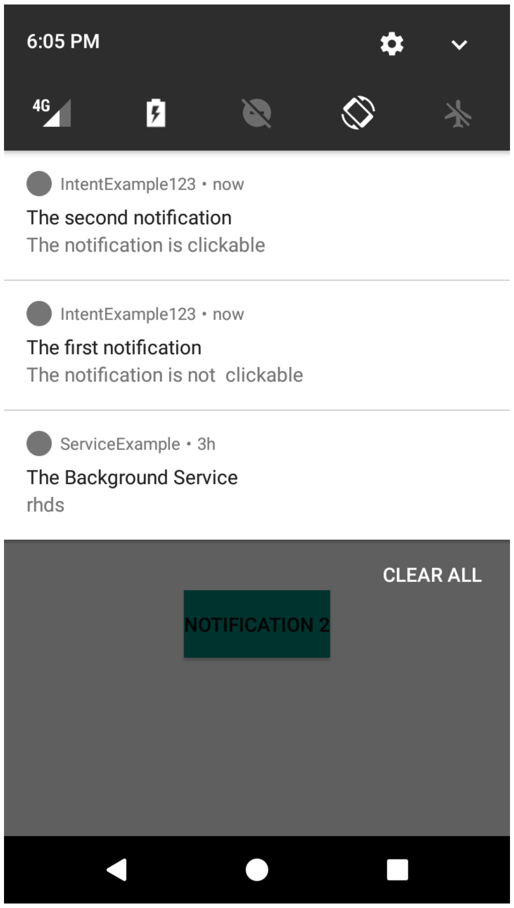
activity_main.xml

```
<RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">
    <Button
            android:id="@+id/btn1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Notification 1"
            android:background="@color/colorPrimary"
            android:layout_alignParentTop="true"
            android:layout_marginTop="133dp"
            android:onClick="show1"
            android:layout_centerInParent="true"/>
    <Button
            android:id="@+id/btn2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Notification 2"
            android:background="@color/colorPrimary"
            android:layout_marginTop="156dp"
            android:layout_below="@+id/btn1"
            android:onClick="show2"
            android:layout_centerInParent="true"/>
</RelativeLayout>
```

<u>MainActivity.kt</u>

```
class MainActivity : AppCompatActivity() {
    val CHANNEL_ID="123"
    val NOTIFICATION_ID_1=111
    val NOTIFICATION_ID_2=222
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        createchannel()
    }
    fun createchannel(){
        if(Build.VERSION.SDK_INT>=Build.VERSION_CODES.O){
            val channel= NotificationChannel(CHANNEL_ID,"notif",
NotificationManager.IMPORTANCE_DEFAULT)
            val
notificationManager:NotificationManager=getSystemService(Context.NOTIFICATION_
SERVICE) as NotificationManager
            notificationManager.createNotificationChannel(channel)
        }
    }
    fun show1(view: View){
        var notif = NotificationCompat.Builder(this, CHANNEL_ID)
            .setSmallIcon(R.drawable.notification_icon_background)
            .setContentTitle("The first notification")
            .setContentText("The notification is not  clickable")
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)
        var notificationManagerCompat = NotificationManagerCompat.from(this)
        notificationManagerCompat.notify(NOTIFICATION_ID_1,notif.build())
    }

    fun show2(view: View){
        val intent=Intent(this,MainActivity::class.java)
        val pendingIntent= PendingIntent.getActivity(this,0,intent,0)

        var notif = NotificationCompat.Builder(this, CHANNEL_ID)
            .setSmallIcon(R.drawable.notification_icon_background)
            .setContentTitle("The second notification")
            .setContentText("The notification is clickable")
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)
            .setContentIntent(pendingIntent)
            .setAutoCancel(true)
        var notificationManagerCompat = NotificationManagerCompat.from(this)
        notificationManagerCompat.notify(NOTIFICATION_ID_2,notif.build())
    }
}
```





IntentExample123 • now
The second notification
The notification is clickable

IntentExample123 • now
The first notification
The notification is not clickable

ServiceExample • 3h
The Background Service
rhds

Broadcast Receivers

Android apps can send or receive broadcast messages from the Android system and other Android apps, similar to the publish-subscribe design pattern. These broadcasts are sent when an event of interest occurs. For example, the Android system sends broadcasts when various system events occur, such as when the system boots up or the device starts charging.

Example:Creating a simple system broadcast receiver to receive the toast if the airplane mode is on or off.

Create the class file Myreceiver.kt

```
class Myreceiver: BroadcastReceiver()  {
    override fun onReceive(context: Context?, intent: Intent?) {
        Toast.makeText(context,"BroadCast is called",Toast.LENGTH_SHORT).show()
    }
}
```

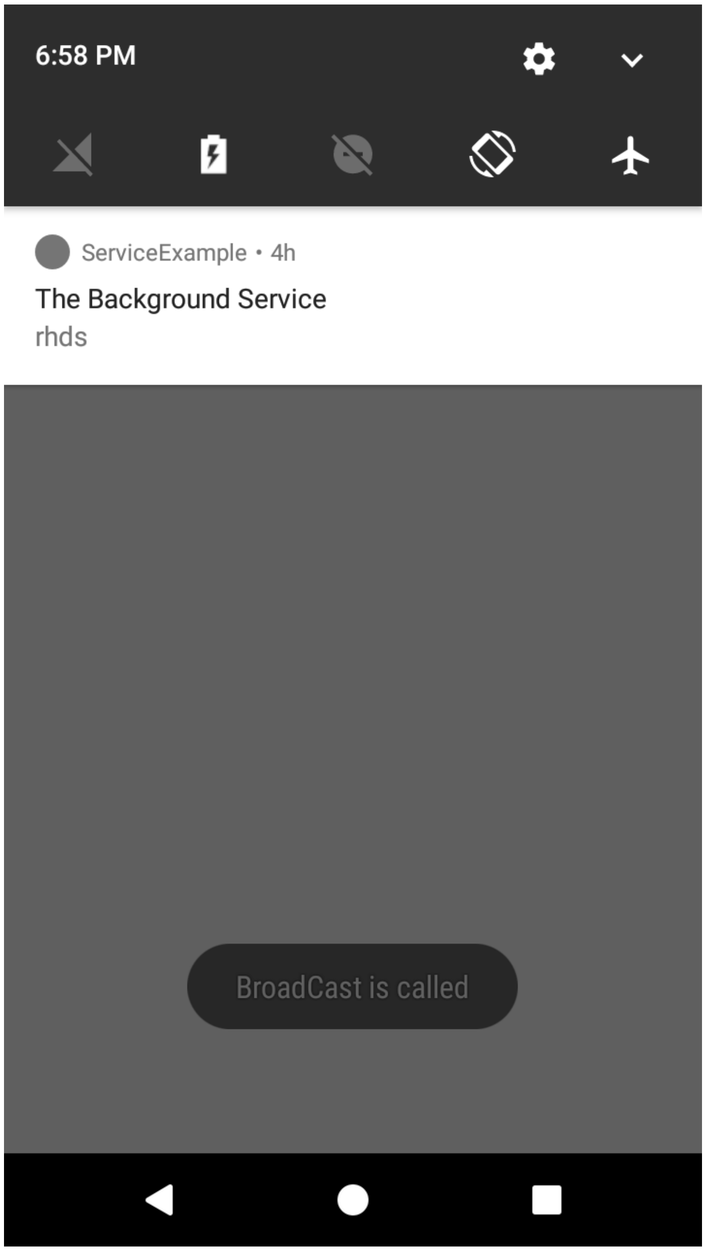
Add the receiver tag in application tag of androidmanifest.xml

```
<receiver android:name=".Myreceiver" android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.AIRPLANE_MODE">
                </action>
            </intent-filter>
```

MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    var myReceiver=Myreceiver()
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        register()
    }
    private fun register() {
        var filter= IntentFilter()
        filter.addAction(Intent.ACTION_AIRPLANE_MODE_CHANGED)
        registerReceiver(myReceiver,filter)
    }
    override fun onDestroy() {
        super.onDestroy()
        unregisterReceiver(myReceiver)
    }
}
```

Output:



The Background Service

rhds

BroadCast is called

PRACTICAL 9

Database Programming with SQLite

SQLite is a opensource SQL database that stores data to a text file on a device. Android comes in with built in SQLite database implementation.

SQLite supports all the relational database features. In order to access this database, you don't need to establish any kind of connections for it like JDBC,ODBC e.t.c.

Sqlite operations or queries are similar to mysql

Create a model class which used to store the data from user for temporarily

User.kt

```
package com.example.user.sqlite

class User {

   var id:Int=0

   lateinit var name:String

   var age:Int=0

   constructor()

   constructor(id:Int,name:String,age:Int){

      this.id=id

      this.name=name

      this.age=age

   }

override fun toString(): String {

      return " ID: $id name: $name age:$age"

   }

}
```

Activitymain.xml

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

<RelativeLayout

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

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

        android:layout_width="match_parent"

        android:layout_height="match_parent"

        tools:context=".MainActivity">

    <EditText

            android:layout_width="350dp"

            android:layout_height="wrap_content"

            android:inputType="textPersonName|number"

            android:ems="10"

            android:layout_alignParentTop="true"

            android:id="@+id/id"

            android:layout_marginTop="8dp"

            android:hint="@string/id"

            android:autofillHints="" tools:targetApi="o"

            android:layout_alignParentStart="true"

            android:layout_marginLeft="17dp"

            android:layout_marginStart="17dp"

            android:layout_alignParentLeft="true"

            android:layout_alignParentEnd="true"

            android:layout_alignParentRight="true"
            android:layout_marginRight="17dp"
            android:layout_marginEnd="17dp"/>
    <EditText
            android:layout_width="350dp"
            android:layout_height="wrap_content"
            android:ems="10"
            android:layout_alignParentStart="true"
            android:layout_marginStart="18dp"
            android:layout_alignParentLeft="true"
            android:id="@+id/name"
            android:layout_marginTop="18dp"
            android:layout_marginLeft="18dp"
            android:hint="@string/name"
            android:layout_below="@+id/id" android:inputType=""/>
    <EditText
            android:layout_width="350dp"
            android:layout_height="wrap_content"
           android:ems="10"
            android:layout_alignParentStart="true"
            android:layout_marginStart="16dp"
            android:layout_alignParentLeft="true"
            android:id="@+id/age"
            android:layout_marginTop="18dp"
            android:layout_marginLeft="16dp"
            android:hint="@string/age"

            android:layout_below="@+id/name"

            android:layout_alignParentEnd="true"

            android:layout_alignParentRight="true"

            android:layout_marginRight="18dp"

            android:layout_marginEnd="18dp" android:inputType="number"/>

        <Button

            android:text="@string/add"

            android:layout_width="wrap_content"

            android:layout_height="wrap_content"

            android:layout_alignParentTop="true"

            android:id="@+id/add"

            android:layout_marginTop="192dp"

            android:layout_alignParentStart="true"

            android:layout_marginLeft="21dp"

            android:layout_marginStart="21dp"

            android:layout_alignParentLeft="true"

            android:textColor="@android:color/white"

            android:background="@color/colorPrimary"

            android:onClick="add"/>

        <Button

            android:text="@string/update"

            android:layout_width="wrap_content"

            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"

            android:layout_marginStart="134dp"

            android:layout_alignParentLeft="true"

            android:layout_alignTop="@+id/add"

            android:id="@+id/update"

            android:layout_marginLeft="134dp"

            android:background="@color/colorPrimary"

            android:textColor="@android:color/white"

            android:onClick="update"/>

        <Button

            android:text="@string/delete"

            android:layout_width="wrap_content"

            android:layout_height="wrap_content"

            android:layout_alignParentRight="true"

            android:layout_alignParentEnd="true"

            android:layout_marginEnd="35dp"

            android:layout_alignTop="@+id/add"

            android:id="@+id/delete"

            android:textColor="@android:color/white"

            android:background="@color/colorPrimary"

           android:layout_marginRight="35dp"

            android:onClick="delete"/>

        <ListView

            android:layout_width="366dp"
            android:layout_height="245dp"

            android:layout_alignParentBottom="true"

            android:layout_marginBottom="1dp"

            android:id="@+id/userdata"

            android:layout_marginTop="14dp"

            android:layout_below="@+id/update"

            android:layout_centerHorizontal="true"/>

    </RelativeLayout>
```

Mainactivity.kt

```
package com.example.user.sqlite

import android.database.sqlite.SQLiteDatabase

import android.support.v7.app.AppCompatActivity

import android.os.Bundle

import android.view.View

import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    lateinit var db:DatabaseHelper

    lateinit var listuser:ArrayList<User>

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

        db= DatabaseHelper(this)

    }

    fun ListViewData()
    {
           listuser=db.allusers() as ArrayList<User>
           val adapter=ListAdapter(this,listuser,id,name,age)
            userdata.adapter=adapter
    }
    fun add(view: View)
    {
        val user=User()
        user.id =id.text.toString().trim().toInt()
        user.name=name.text.toString().trim()
        user.age=age.text.toString().trim().toInt()
        db.add(user)
        ListViewData()
    }
    fun update(view: View)
    {
        val user=User()
        user.id =id.text.toString().trim().toInt()
        user.name=name.text.toString().trim()
        user.age=age.text.toString().trim().toInt()
        db.update(user)
        ListViewData()
    }
fun delete(view: View)
```

```
    {
        val user=User()
        user.id =id.text.toString().trim().toInt()
        db.delete(user)
        ListViewData()
    }
}
```

Now create sqldatabasehelper class to perform the queries or CRUD operation

DatabaseHelper.kt

```
package com.example.user.sqlite


import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper
import java.util.*
class DatabaseHelper(context:Context):SQLiteOpenHelper(context,
DATABASE_NAME,null, DATABASE_VER){
    override fun onCreate(db: SQLiteDatabase?) {
        val create_tabel_query="CREATE TABLE IF NOT EXISTS "+ TABLE_NAME+
                "("+ ID+" INTEGER PRIMARY KEY,"+ NAME+" TEXT,"+ AGE+" INTEGER)"
        db!!.execSQL(create_tabel_query)
    }
```

```
override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {
    val drop_tabel_query="DROP TABLE IF EXISTS "+ TABLE_NAME
    db!!.execSQL(drop_tabel_query)
}
fun allusers():List<User>{
    val listuser=ArrayList<User>()
    val fetch_query="SELECT * FROM "+ TABLE_NAME
    val db:SQLiteDatabase=this.writableDatabase
    val cursor:Cursor=db.rawQuery(fetch_query,null)
    if(cursor.moveToFirst())
    {
        do{
            val user=User()
            user.id=cursor.getInt(cursor.getColumnIndex(ID))
            user.name=cursor.getString(cursor.getColumnIndex(NAME))
            user.age=cursor.getInt(cursor.getColumnIndex(AGE))
            listuser.add(user)
        }while (cursor.moveToNext())
    }
    return listuser
}
fun add(user:User){
    val db:SQLiteDatabase=this.writableDatabase
    var values=ContentValues()
```

```
        values.put(ID,user.id)

        values.put(NAME,user.name)

        values.put(AGE,user.age)

        db.insert(TABLE_NAME,null,values)

    }

    fun update(user:User){

        val db:SQLiteDatabase=this.writableDatabase

        var values=ContentValues()

        values.put(ID,user.id)

        values.put(NAME,user.name)

        values.put(AGE,user.age)

        db.update(TABLE_NAME,values,"$ID=?", arrayOf(user.id.toString()))

    }

    fun delete(user:User){

        val db:SQLiteDatabase=this.writableDatabase

        db.delete(TABLE_NAME,"$ID=?", arrayOf(user.id.toString()))

    }

    companion object {

            private val DATABASE_VER=1

            private val DATABASE_NAME="data.db"

            private val TABLE_NAME="USER"

            private val ID="id"

            private val NAME="name"

            private val AGE="age"
    }

}
```

Create a xml Resource file for row in list view which going to display the data from the database

list_row_layout.xml

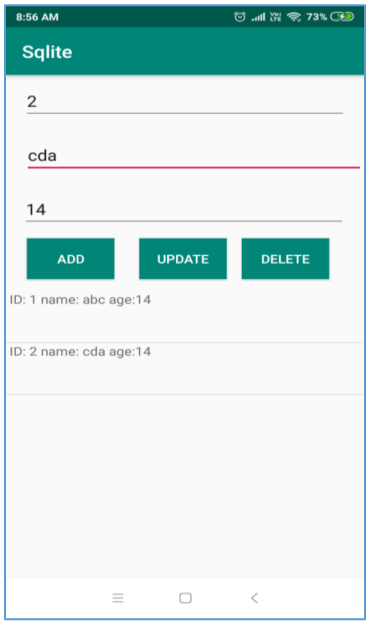
```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

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

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

        android:layout_width="match_parent"

        android:layout_height="match_parent">

    <TextView

            android:text="@string/user"

            android:layout_width="match_parent"

            android:layout_height="60dp"

            tools:layout_editor_absoluteY="5dp"

            tools:layout_editor_absoluteX="6dp"

            android:id="@+id/user_row_data"

            android:layout_alignParentTop="true"

            android:layout_marginTop="0dp" />

</RelativeLayout>
```

Now create a ListAdapter which lists the data from database in list view

ListAdapter.kt

```
package com.example.user.sqlite


import android.app.Activity

import android.content.Context

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import android.widget.BaseAdapter

import android.widget.EditText

import kotlinx.android.synthetic.main.list_row_layout.view.*

class ListAdapter(internal var activity: Activity,

                  internal var listuser:List<User>,

                  internal var userid:EditText,

                  internal var username:EditText,

                  internal var userage:EditText):BaseAdapter(){

internal var inflater:LayoutInflater

init {

         inflater=activity.getSystemService(Context.LAYOUT_INFLATER_SERVICE) as
LayoutInflater

}

    override fun getView(position: Int, convertView: View?, parent: ViewGroup?): View {

        val listrow:View
        listrow= inflater.inflate(R.layout.list_row_layout,null)

        listrow.user_row_data.text=listuser[position].toString()

        return listrow

    }

    override fun getItem(position: Int): Any {

        return listuser[position]

    }

    override fun getItemId(position: Int): Long {

        return  listuser[position].id.toLong()

    }

    override fun getCount(): Int {

        return listuser.size

    }

}
```

Output:



```
ID: 1 name: abc age:14
ID: 2 name: cda age:14
```

PRACTICAL NO.10

Programming threads, handles and asynchronized programs

Threads and Handers

A **thread** is a thread of execution in a program. The Java Virtual Machine allows an application to have multiple threads of execution running concurrently.

Every thread has a priority. Threads with higher priority are executed in preference to threads with lower priority.

A **Handler** allows you to send and process Message and Runnable objects associated with a thread's MessageQueue. Each Handler instance is associated with a single thread and that thread's message queue. When you create a new Handler, it is bound to the thread / message queue of the thread that is creating it -- from that point on, it will deliver messages and runnables to that message queue and execute them as they come out of the message queue.

An asynchronous task is defined by a computation that runs on a background thread and whose result is published on the UI thread. An asynchronous task is defined by 3 generic types, called Params, Progress and Result, and 4 steps, called onPreExecute, doInBackground, onProgressUpdate and onPostExecute.

Example :Thread and handler on progress bar

Activityman.xml

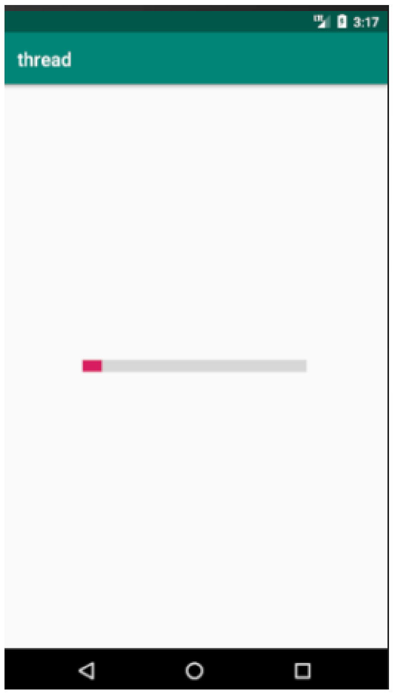
```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

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

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

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

        android:layout_width="match_parent"

        android:layout_height="match_parent"

        tools:context=".MainActivity">

    <ProgressBar

            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            style="?android:attr/progressBarStyleHorizontal"
            android:layout_centerInParent="true"
            android:max="50"
            android:maxWidth="100dp"
            android:scaleX="5"
            android:scaleY="3"
            android:id="@+id/progressBar"/>
</RelativeLayout>
```

Mainactivity.kt

```
package com.example.thread

import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.os.Handler
import kotlinx.android.synthetic.main.activity_main.*
import java.lang.Exception

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var currpos=0
        var handler=Handler()
    var thread=Thread(object :Runnable {
        override fun run() {
            for ( i in 1..50)
            {
                currpos=i
                try {
                    Thread.sleep(1000)


                }
                catch (e:Exception)
                {


                }
                handler.post(object :Runnable{
                    override fun run() {
                        progressBar.progress = currpos
                    }


                })
            }


        }
    })
    thread.start()
}
```

```
}
```

Output:



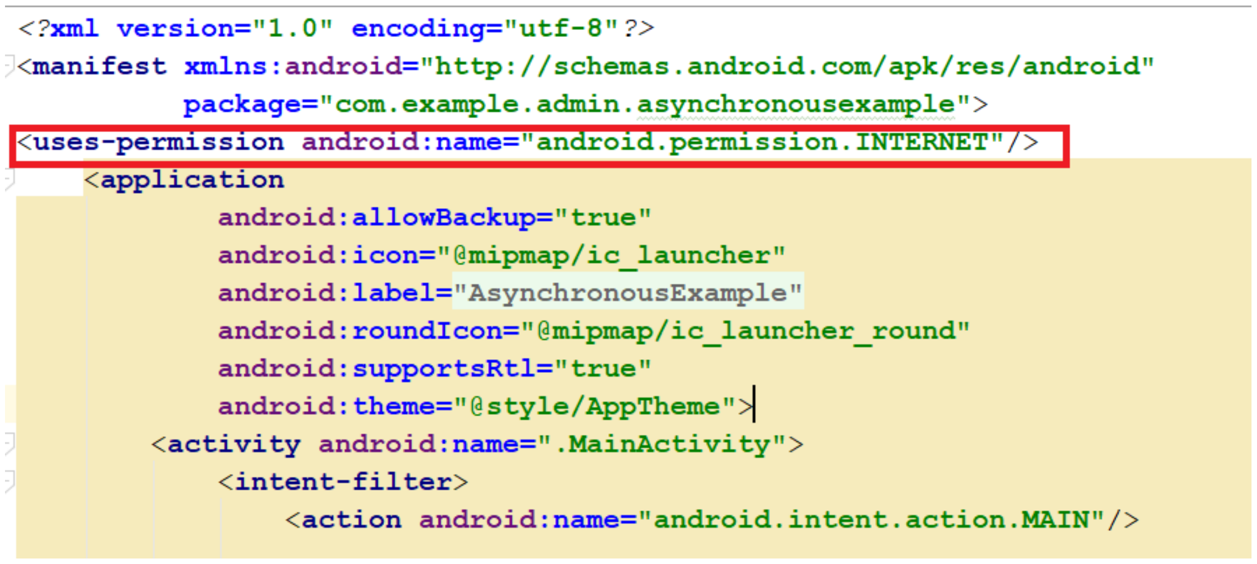
Asynchronized Programs

An asynchronous task is defined by a computation that runs on a background thread and whose result is published on the UI thread. An asynchronous task is defined by 3 generic types, called Params, Progress and Result, and 4 steps, called onPreExecute, doInBackground, onProgressUpdate and onPostExecute.

Example: Downloading the image from any url

Note: In AndroidManifest.xml add internet permission by following tag in manifest tag

<uses-permission android:name="android.permission.INTERNET"/>



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="com.example.admin.asynchronousexample">
<uses-permission android:name="android.permission.INTERNET"/>
    <application
            android:allowBackup="true"
            android:icon="@mipmap/ic_launcher"
            android:label="AsynchronousExample"
            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"/>
```

Activitymain.xml

Activitymain.xml

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

<RelativeLayout

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

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

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

    android:layout_width="match_parent"

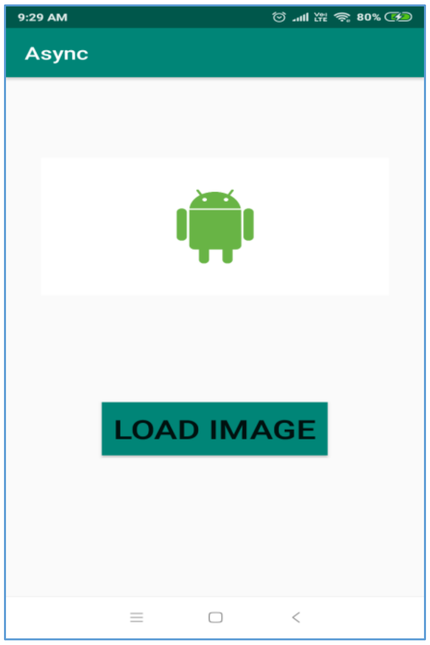
```
        android:layout_height="match_parent"
        tools:context=".MainActivity">
    <ImageView
            android:layout_width="300dp"
            android:layout_height="300dp"
            android:id="@+id/img"
            android:layout_marginTop="20dp"
            android:layout_alignParentTop="true"
            android:layout_centerHorizontal="true"
    />
    <Button
            android:id="@+id/btn"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/img"
            android:layout_marginTop="50dp"
            android:layout_centerInParent="true"
            android:background="@color/colorPrimary"
            android:text="Load Image"
            android:padding="10dp"
            android:textSize="30dp"
            android:onClick="download"
    />
</RelativeLayout>
```

Mainactivty.kt

```
package com.example.user.async


import android.graphics.Bitmap

import android.graphics.BitmapFactory

import android.os.AsyncTask

import android.support.v7.app.AppCompatActivity

import android.os.Bundle

import android.view.View

import kotlinx.android.synthetic.main.activity_main.*

import java.lang.Exception

import java.net.URL

class MainActivity : AppCompatActivity() {

    lateinit var url:String

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)


    }

    fun download(view: View){

        url="https://www.android.com/static/2016/img/share/andy-lg.png"

        var a=downloadimage().execute(url)


    }
```

```
   inner class downloadimage:AsyncTask<String,Int,Bitmap>(){

      override fun doInBackground(vararg params: String?): Bitmap? {

          var img:Bitmap?=null

         try {

            var url = URL(params[0])

            var inputStream = url.openStream()

            img= BitmapFactory.decodeStream(inputStream)


    }

         catch (e:Exception){

            e.printStackTrace()

         }

 return img

      }


      override fun onPostExecute(result: Bitmap?) {

         img.setImageBitmap(result)

         super.onPostExecute(result)

      }

   }

}
```

Output:



Practical No.11

Programming Media API and Telephone API

Media API

Provides classes that manage various media interfaces in audio and video.

The Media APIs are used to play and, in some cases, record media files. This includes audio (e.g., play MP3s or other music files, ringtones, game sound effects, or DTMF tones) and video (e.g., play a video streamed over the web or from local storage).

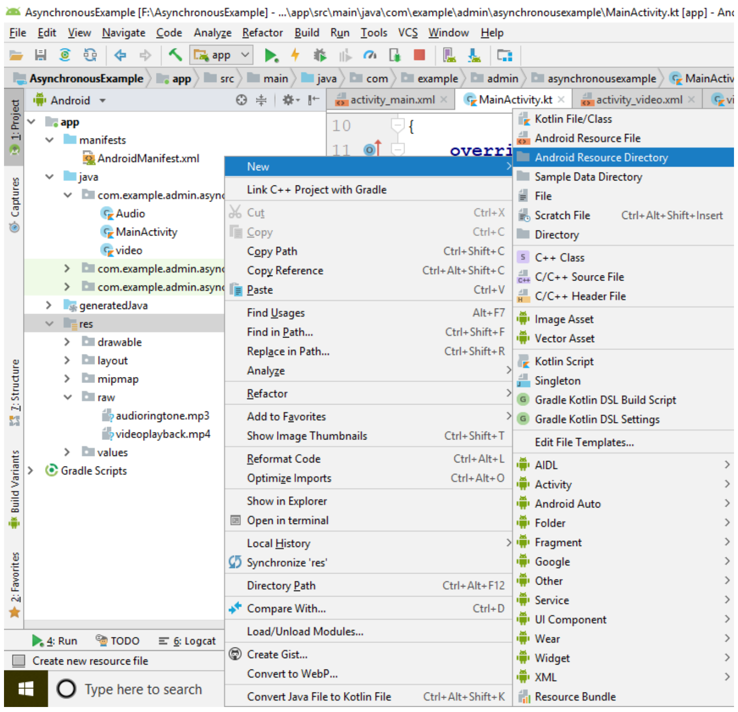
Remember android support only some video or audio codec so see the list of codec supported on developer.android site

Activity_main.xml

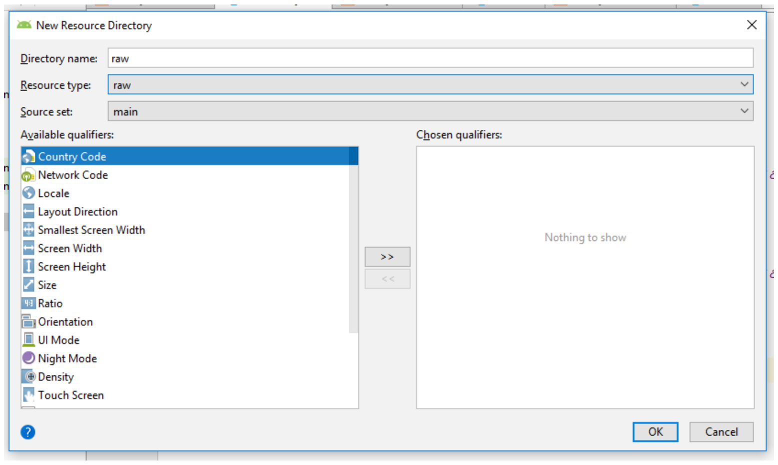
```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">


  <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/audio"
      android:layout_alignParentTop="true"
      android:text="Audio"
      android:background="@color/colorPrimary"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="150dp"
      android:onClick="audio"
  />
```

```
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/video"
        android:text="video"
        android:background="@color/colorPrimary"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/audio"
        android:layout_marginTop="60dp"
        android:onClick="video"
    />
</RelativeLayout>
```

Create android resource directory under res folder to store audio or video file



Change the resource type to raw



New Resource Directory

Directory name: raw

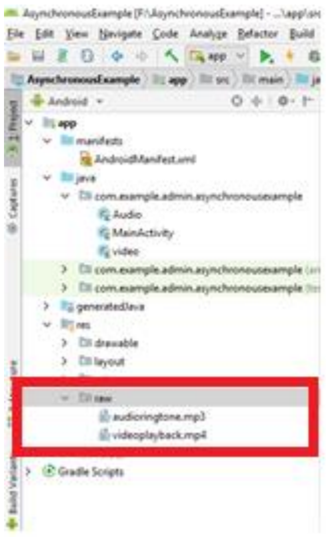
Resource type: raw

Source set: main

Available qualifiers: Country Code, Network Code, Locale, Layout Direction, Smallest Screen Width, Screen Width, Screen Height, Size, Ratio, Orientation, UI Mode, Night Mode, Density, Touch Screen

Chosen qualifiers: Nothing to show

Paste the audio (.mp3) / video(.mp4) file in the raw folder


AsynchronousExample [F:\AsynchronousExample] - ...\app\src
File Edit View Navigate Code Analyze Refactor Build
app
AsynchronousExample
app
src
main
Android
app
manifests
AndroidManifest.xml
java
com.example.admin.asynchronousexample
Audio
MainActivity
video
com.example.admin.asynchronousexample
com.example.admin.asynchronousexample
generatedJava
res
drawable
layout
raw
audioringtone.mp3
videoplayback.mp4
Gradle Scripts

Create the activities for audio and video files and rename them

activity_audio.xml

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Audio">
```

```
<SeekBar
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/audiobar"
        android:layout_alignParentTop="true"
        android:layout_marginTop="130dp"/>
<Button
        android:id="@+id/play"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Play"
        android:background="@color/colorPrimary"
        android:layout_below="@id/audiobar"
        android:layout_alignParentLeft="true"
        android:layout_marginLeft="40dp"
        android:layout_marginTop="150dp"
        android:layout_marginRight="70dp"
        android:onClick="play"
/>
<Button
        android:id="@+id/stop"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Stop"
        android:background="@color/colorPrimary"
        android:layout_below="@id/audiobar"
        android:layout_alignParentRight="true"
        android:layout_toRightOf="@+id/play"
        android:layout_marginRight="40dp"
        android:layout_marginLeft="70dp"
```

```
            android:layout_marginTop="150dp"
            android:onClick="stop"
    />
    <Button
            android:id="@+id/back"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="BACK"
            android:background="@color/colorPrimary"
            android:layout_below="@+id/play"
            android:layout_marginTop="50dp"
            android:layout_centerHorizontal="true"
            android:onClick="back"
    />
</RelativeLayout>
```

Audio.kt

```
class Audio : AppCompatActivity() {
    lateinit var mediaPlayer: MediaPlayer
    lateinit var handler: Handler
    lateinit var runnable: Runnable
    @RequiresApi(Build.VERSION_CODES.LOLLIPOP)
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_audio)
        mediaPlayer = MediaPlayer.create(applicationContext, R.raw.audioringtone)
        var attributes = AudioAttributes.Builder()
            .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC)
            .build()
        mediaPlayer.setAudioAttributes(attributes)
        handler = Handler()
        mediaPlayer.setOnPreparedListener(object : MediaPlayer.OnPreparedListener {
            override fun onPrepared(mp: MediaPlayer?) {
                audiobar.max = mediaPlayer.duration
                changeprogress()
                mediaPlayer.start()

            } })
        audiobar.setOnSeekBarChangeListener(object :
SeekBar.OnSeekBarChangeListener {
            override fun onProgressChanged(seekBar: SeekBar?, progress: Int, input:
Boolean) {
//
Toast.makeText(applicationContext,progress.toString(),Toast.LENGTH_SHORT).show
()
                if (input) {
                    mediaPlayer.seekTo(progress)
                }
            }
            override fun onStartTrackingTouch(seekBar: SeekBar?) {
            }
            override fun onStopTrackingTouch(seekBar: SeekBar?) {
            }
        }) }
    fun changeprogress() {
        audiobar.progress = mediaPlayer.currentPosition
        if (mediaPlayer.isPlaying) {
            runnable = Runnable {
                changeprogress()
            }
            handler.postDelayed(runnable, 1000)
        }
    }
    fun play(view: View) {
        audiobar.progress = mediaPlayer.currentPosition
        mediaPlayer.start()
    }
    fun stop(view: View) {
        mediaPlayer.pause()
    }
    fun back(view:View){
        startActivity(Intent(this,MainActivity::class.java))
    }

    override fun onResume() {
        super.onResume()
        mediaPlayer.start()
        audiobar.progress = mediaPlayer.currentPosition
    }
    override fun onPause() {
        super.onPause()
        mediaPlayer.pause()
    }
    override fun onDestroy() {
        super.onDestroy()
        mediaPlayer.release()
        handler.removeCallbacks(runnable)
    }
}
```

activity_video.xml

```
<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=".video">

    <VideoView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/videoView"/>
</RelativeLayout>
```

video.kt

```
class video : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_video)
        var mediaController=MediaController(this)
        var
path="android.resource://com.example.admin.asynchronousexample/${R.raw.videopla
yback}"
        var uri= Uri.parse(path)
        videoView.setVideoURI(uri)
        mediaController.setAnchorView(videoView)
        videoView.setMediaController(mediaController)
        videoView.start()
    }
}
```

<u>Note: In above kt file (Video.kt)</u> :

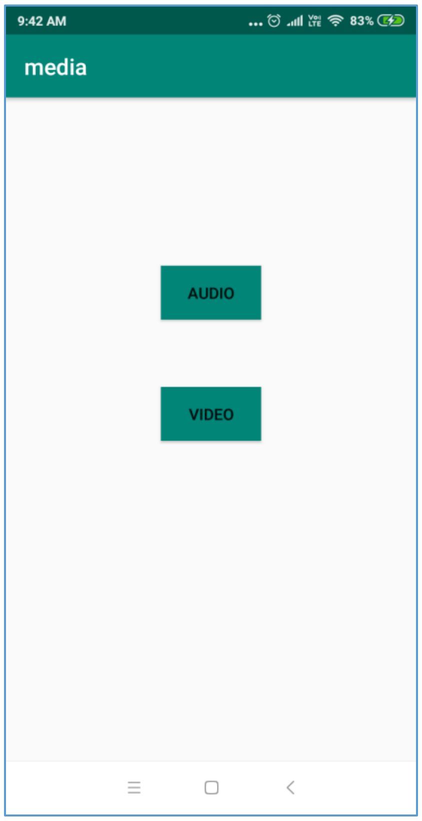
var path="android.resource://com.example.admin.asynchronousexample/${R.raw.videoplayback}"

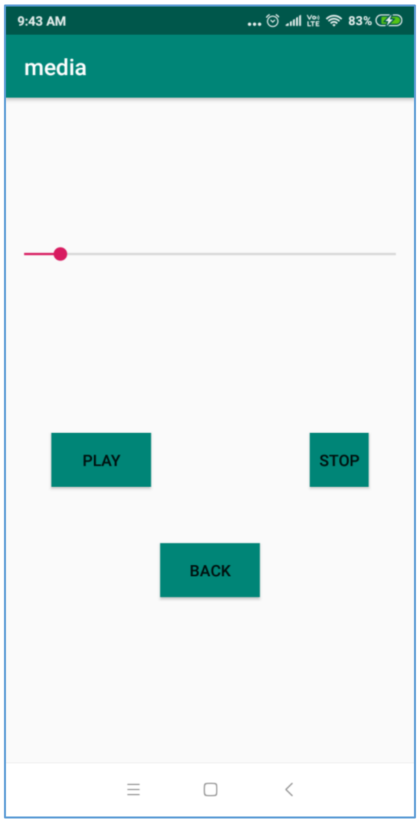
this path (com.example.admin.asynchronousexample)should be as per your package name

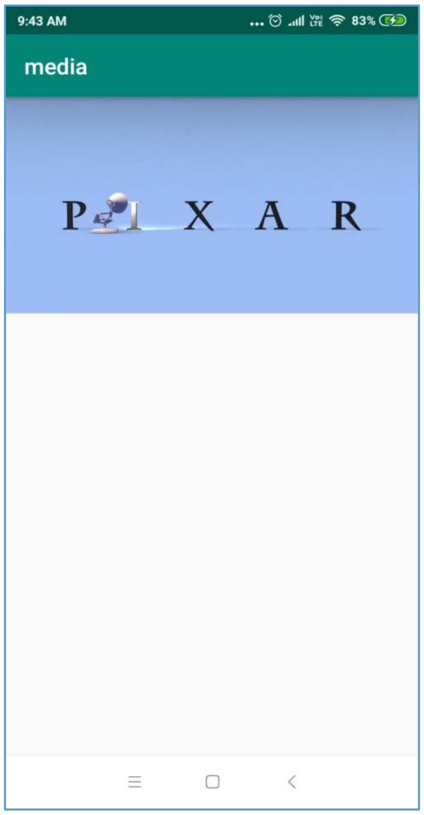
<u>MainActivity.kt</u>

```
class MainActivity : AppCompatActivity()
{
    override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
}
fun audio(view: View){
    var intent= Intent(this,Audio::class.java)
    startActivity(intent)
}
fun video(view:View){
    var intent= Intent(this,video::class.java)
    startActivity(intent)
}
}
```

Output:


9:42 AM
83%
media
AUDIO
VIDEO


9:43 AM
83%
media
PLAY
STOP
BACK


9:43 AM
83%
media
PIXAR

Telephone API

The Telephony provider contains data related to phone operation, specifically SMS and MMS messages, access to the APN list, including the MMSC to use, and the service state.

You can access contacts using contact class you can make a calland many more functions are available in telephone api

Add the follwing permissions in androidmanifest.xml file in the manifest tag

```
<uses-permission android:name="android.permission.READ_CONTACTS" >
   </uses-permission>
   <uses-permission android:name="android.permission.CALL_PHONE">
   </uses-permission>
```

Activity_main.xml

```
<RelativeLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      tools:context=".MainActivity">
   <TextView
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:id="@+id/title"
         android:text="Click on number to make a call"
         android:textSize="25sp"
         android:textAlignment="center"
         android:textStyle="bold"
   />
   <ListView
```

```
          android:id="@+id/contacts"
          android:layout_width="match_parent"
          android:layout_height="match_parent"
          android:layout_below="@+id/title"
    >
    </ListView>
</RelativeLayout>
```

Create a new xml resource file for inflating in listadapter and rename it (contact_data.xml)

```
<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:id="@+id/cdata"
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textSize="25sp"/>
</LinearLayout>
```

<u>Create a model class from storing the contact details like name,phone number etc</u>

<u>contact.kt</u>

```
class contact {
    var pno:String?=null
    var name:String?=null
    fun getContact():String
    {
        return "$name \n$pno"
    }
}
```

Create a listadapter class for storing the contact details in list

listadapter.kt

```
class listadapter(var activity:Activity, var contactlist:ArrayList<contact>):BaseAdapter(){
    override fun getView(position: Int, convertView: View?, parent: ViewGroup?): View {
        var inflater=LayoutInflater.from(activity)
        var view=inflater.inflate(R.layout.contact_data,null)
        view.cdata.text=contactlist[position].getContact()

        view.setOnClickListener {
            var alertDialog=AlertDialog.Builder(activity)
            alertDialog.setTitle("Make a call")

alertDialog.setMessage("to:${contactlist[position].name}\n${contactlist[position].pno}")
            alertDialog.setPositiveButton("Yes",DialogInterface.OnClickListener {
                    dialog, which ->
                try {
                    var intent = Intent(Intent.ACTION_CALL)
                    intent.data = Uri.parse("tel:${contactlist[position].pno}")
                    activity.startActivity(intent)
                }
                catch(e:Exception){
```

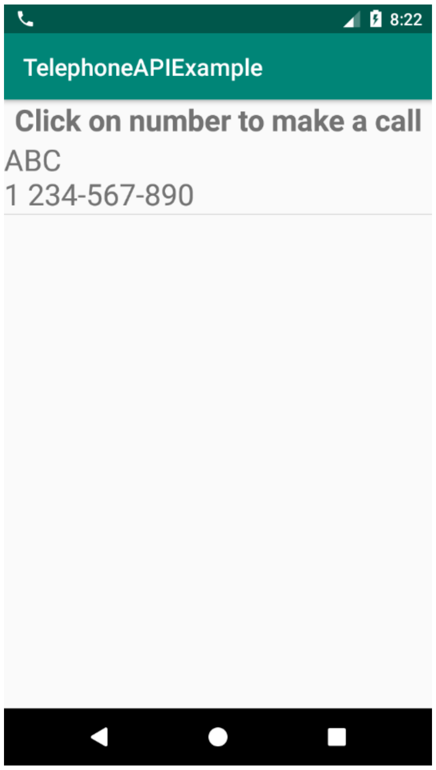
```
                Toast.makeText(activity,"Please allow the permission" +

                        "",Toast.LENGTH_SHORT).show()

            }

        })

        alertDialog.setNegativeButton("No",DialogInterface.OnClickListener { dialog,
which ->

            dialog.cancel()

        })

        alertDialog.create()

        alertDialog.show()

    }

    return view

}


override fun getItem(position: Int): Any {

    return contactlist[position]

}


override fun getItemId(position: Int): Long {

    return position.toLong()

}


override fun getCount(): Int {

    return contactlist.size
    }


}
```

<u>MainActivity.kt</u>

```
class MainActivity : AppCompatActivity() {

    lateinit var adapter:listadapter

    lateinit var contactList:ArrayList<contact>

    var cursor: Cursor?=null

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

        var permissionCode1= ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_CONTACTS)

        var permissionCode2= ContextCompat.checkSelfPermission(this,
Manifest.permission.CALL_PHONE)

        if(permissionCode1==PackageManager.PERMISSION_GRANTED &&
permissionCode2==PackageManager.PERMISSION_GRANTED)

        {

            showContacts()

        }

        else{

            ActivityCompat.requestPermissions(this,
arrayOf(Manifest.permission.READ_CONTACTS,Manifest.permission.CALL_PHONE),
100)

        }

        adapter= listadapter(this,contactList)
        contacts.adapter=adapter

    }



    override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out
String>, grantResults: IntArray) {

        super.onRequestPermissionsResult(requestCode, permissions, grantResults)

        if(requestCode==100){

            showContacts()

        }

        else{

            Toast.makeText(this,"Please Provide the Contact
permission",Toast.LENGTH_SHORT).show()

        }

    }

    private fun showContacts() {


cursor=contentResolver.query(ContactsContract.CommonDataKinds.Phone.CONTEN
T_URI,

            null,null,null,ContactsContract.Contacts.SORT_KEY_PRIMARY )

        contactList=ArrayList<contact>()

        var number: String?

        var lastnumber:String?=null

        while(cursor!!.moveToNext())

        {


number=(cursor!!.getString(cursor!!.getColumnIndex(ContactsContract.CommonDataKi
nds.Phone.NUMBER)))
```

```
            if(number!=null ) {
                number=number.replace("\\s".toRegex(),"")
                if (!number!!.equals(lastnumber)) {
                    lastnumber = number


                    var contact = contact()
                    contact.name =

cursor!!.getString(cursor!!.getColumnIndex(ContactsContract.CommonDataKinds.Phon
e.DISPLAY_NAME))




                    contact.pno =
cursor!!.getString(cursor!!.getColumnIndex(ContactsContract.CommonDataKinds.Phon
e.NUMBER))
                    contactList.add(contact)
                }
            }
        }
        cursor!!.close()


    }
}
```

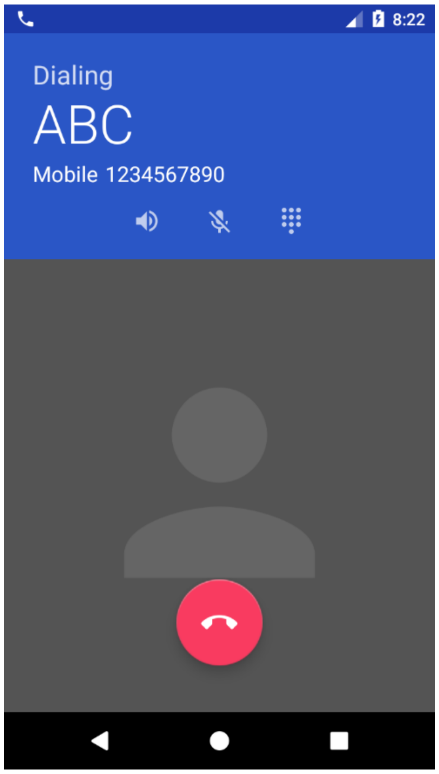
<u>Output:</u>


8:22

TelephoneAPIExample

Click on number to make a call

ABC
1 234-567-890


8:22

Dialing

ABC

Mobile 1234567890

PRACTICAL 12

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
```

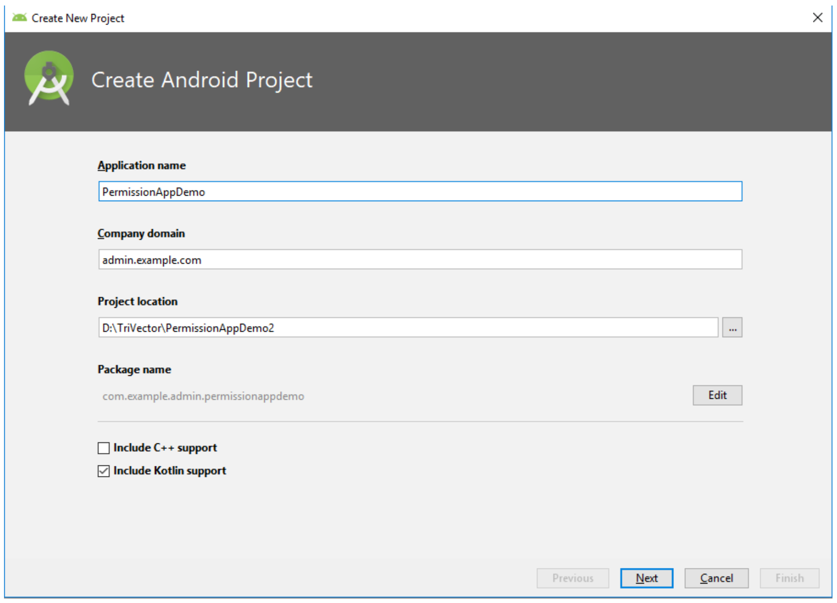
Extra Packages requied in MainActivity.kt

```
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**



Application name: PermissionAppDemo

Company domain: admin.example.com

Project location: D:\TriVector\PermissionAppDemo2

Package name: com.example.admin.permissionappdemo

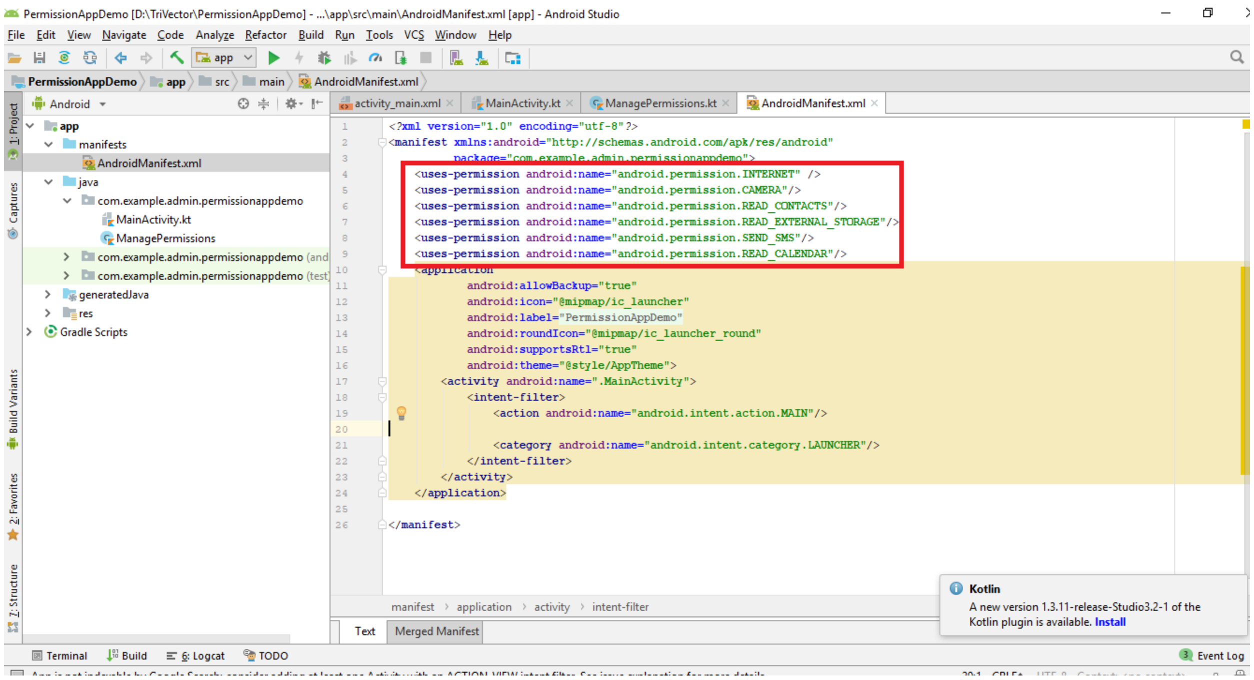
☐ Include C++ support

☑ Include Kotlin support

2. **An app must publicize the permissions it requires by including <uses-permission> tags in the app manifest.**

```
<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"/>
```



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="com.example.admin.permissionappdemo">
    <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"/>
    <application
            android:allowBackup="true"
            android:icon="@mipmap/ic_launcher"
            android:label="PermissionAppDemo"
            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>
    </application>

</manifest>
```

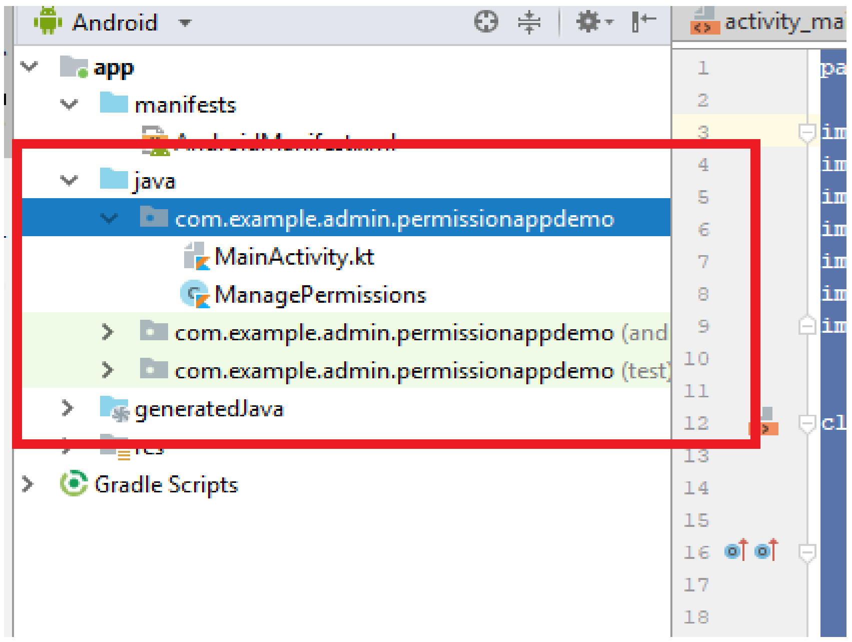
Kotlin
A new version 1.3.11-release-Studio3.2-1 of the Kotlin plugin is available. Install

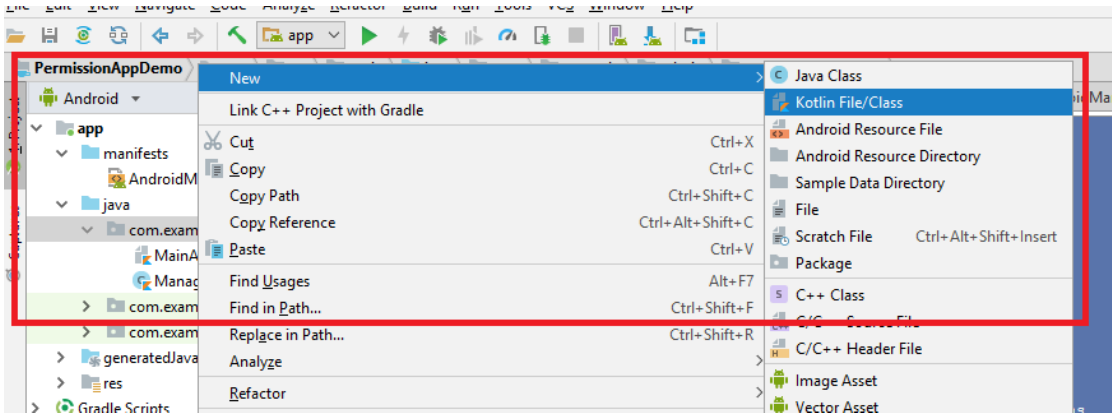
3. MainActivity.kt

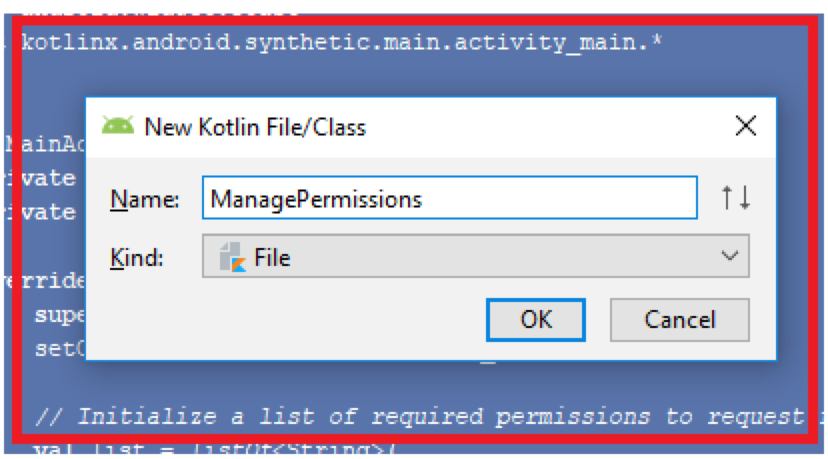
```
package com.example.admin.permissionappdemo

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()
}
```

4. Create a New Kotlin Class

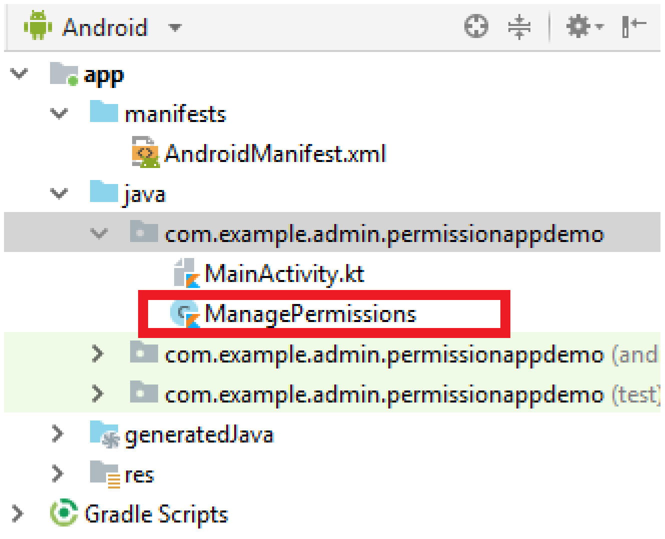
app->src->main->java->com.example.admin.permissionappdemo


Android
activity_ma
app
manifests
AndroidManifest.xml
java
com.example.admin.permissionappdemo
MainActivity.kt
ManagePermissions
com.example.admin.permissionappdemo (and
com.example.admin.permissionappdemo (test)
generatedJava
res
Gradle Scripts





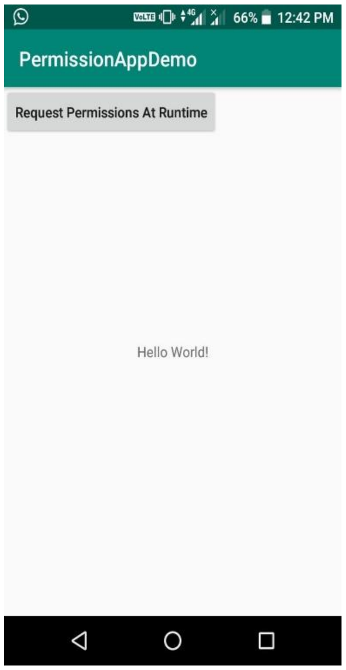
Class file is generated

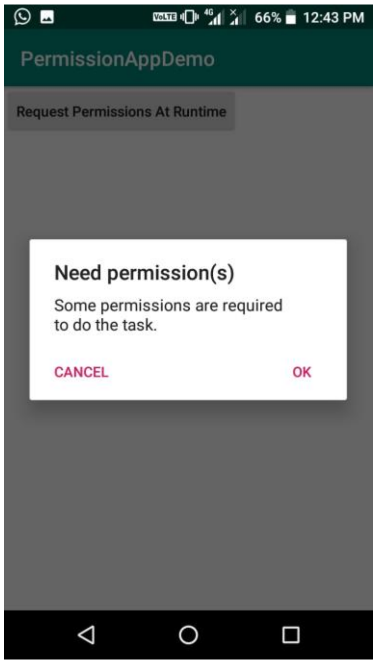


Android
app
manifests
AndroidManifest.xml
java
com.example.admin.permissionappdemo
MainActivity.kt
ManagePermissions
com.example.admin.permissionappdemo (and
com.example.admin.permissionappdemo (test)
generatedJava
res
Gradle Scripts

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

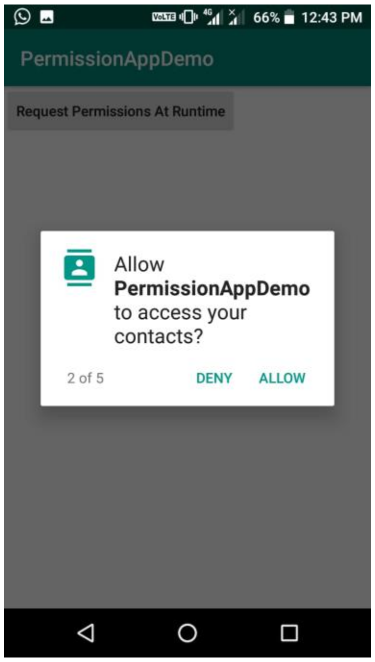
```kotlin
import android.app.Activity
import android.content.pm.PackageManager
import android.support.v4.app.ActivityCompat
import android.support.v4.content.ContextCompat
import android.support.v7.app.AlertDialog


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
        }
        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
    }
}
```



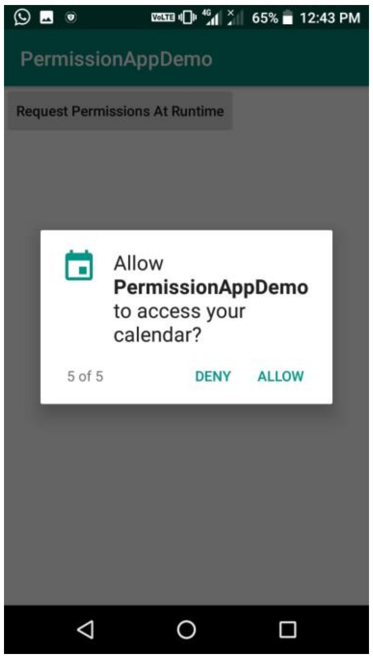


Need permission(s)

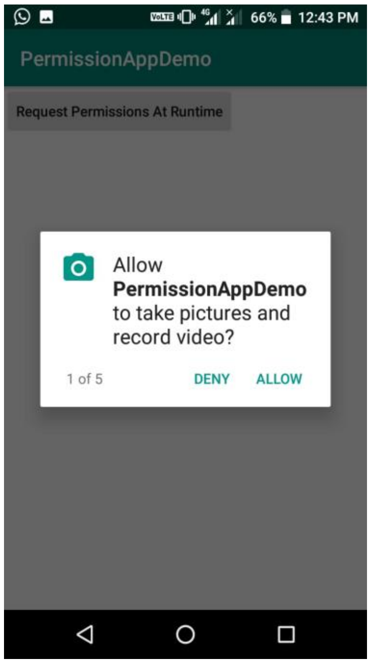
Some permissions are required to do the task.



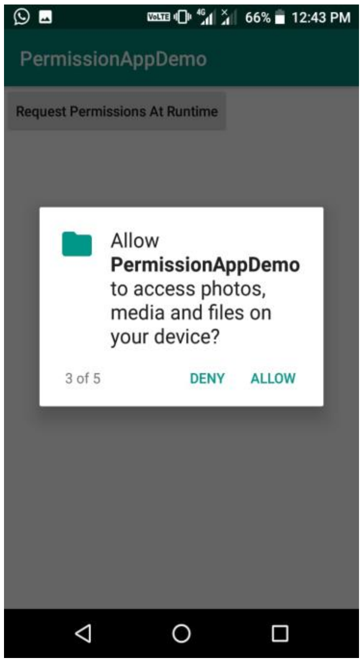
Allow **PermissionAppDemo** to access your contacts?



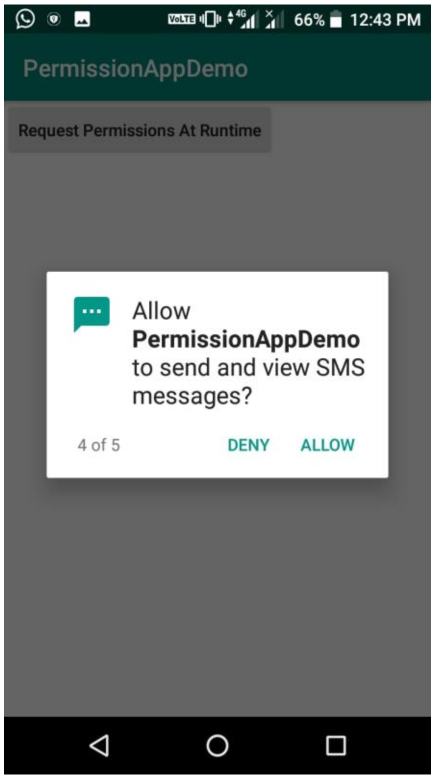
Allow **PermissionAppDemo** to access your calendar?



Allow **PermissionAppDemo** to take pictures and record video?

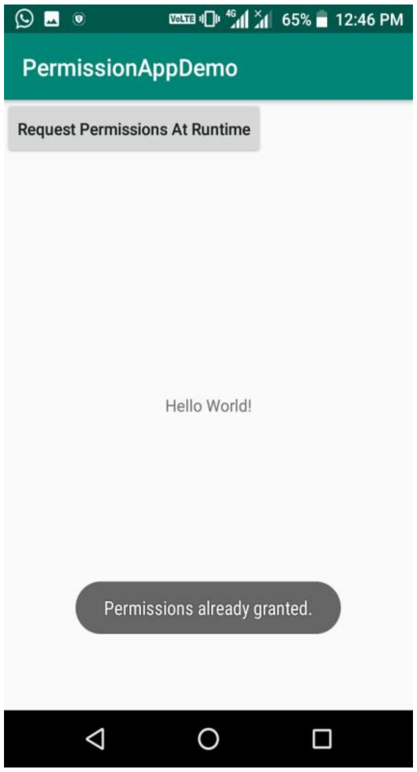


Allow **PermissionAppDemo** to access photos, media and files on your device?



Allow **PermissionAppDemo** to send and view SMS messages?

4 of 5 DENY ALLOW



Hello World!

Permissions already granted.