S.Y.B.Sc. C.S.

Android Application Development

Practicals

Practical 1- Install Android Studio and Run Hello World Program

Download Android Studio

Google provides Android Studio for the Windows, Mac OS X, and Linux platforms. You can download this software from the Android Studio homepage.

https://developer.android.com/studio/index.html

Click on Download Android Studio 3.0.1 for Windows

Hardware Requirements

- 2 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum, 4 GB Recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

Windows OS

- Microsoft Windows 7/8/10 (32-bit or 64-bit)
- JDK 8
- For accelerated emulator: 64-bit operating system and Intel processor with support for Intel VT-x, Intel EM64T (Intel 64), and Execute Disable (XD) Bit functionality

Mac OS

- Mac OS X 10.8.5 or higher, up to 10.11.4 (El Capitan)
- JDK 6

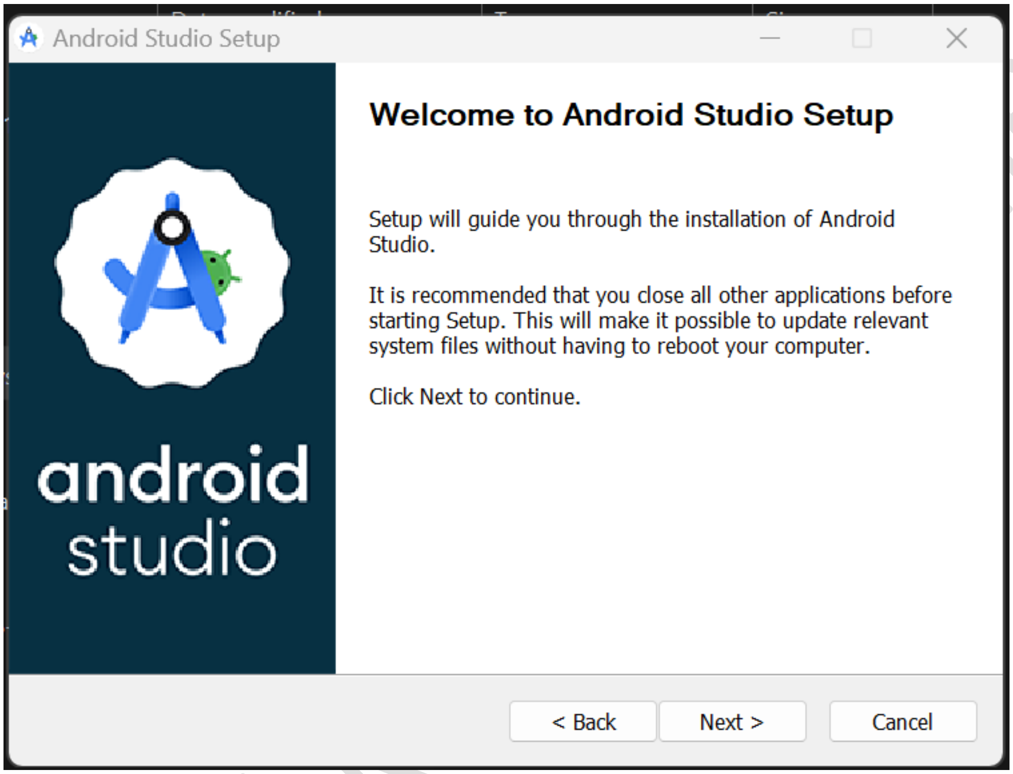
Linux OS

- **GNOME or KDE desktop**: Tested on Ubuntu 12.04, Precise Pangolin (64-bit distribution capable of running 32-bit applications)
- 64-bit distribution capable of running 32-bit applications
- GNU C Library (glibc) 2.11 or later
- JDK 8
- For accelerated emulator: Intel processor with support for Intel VT-x, Intel EM64T (Intel 64), and Execute Disable (XD) Bit functionality, or AMD processor with support for AMD Virtualization (AMD-V)

Download the appropriate file depending on whether your operating system is 32-bit or 64-bit OS. For 64-bit Windows 8.1 the file is android-studio-bundle-143.2821654-windows.exe

Installing Android Studio on 64-bit Windows 8.1

Launch `android-studio-bundle-143.2821654-windows.exe` to start the installation process. The installer responds by presenting the Android Studio Setup dialog box shown in Figure 1.



Android Studio Setup

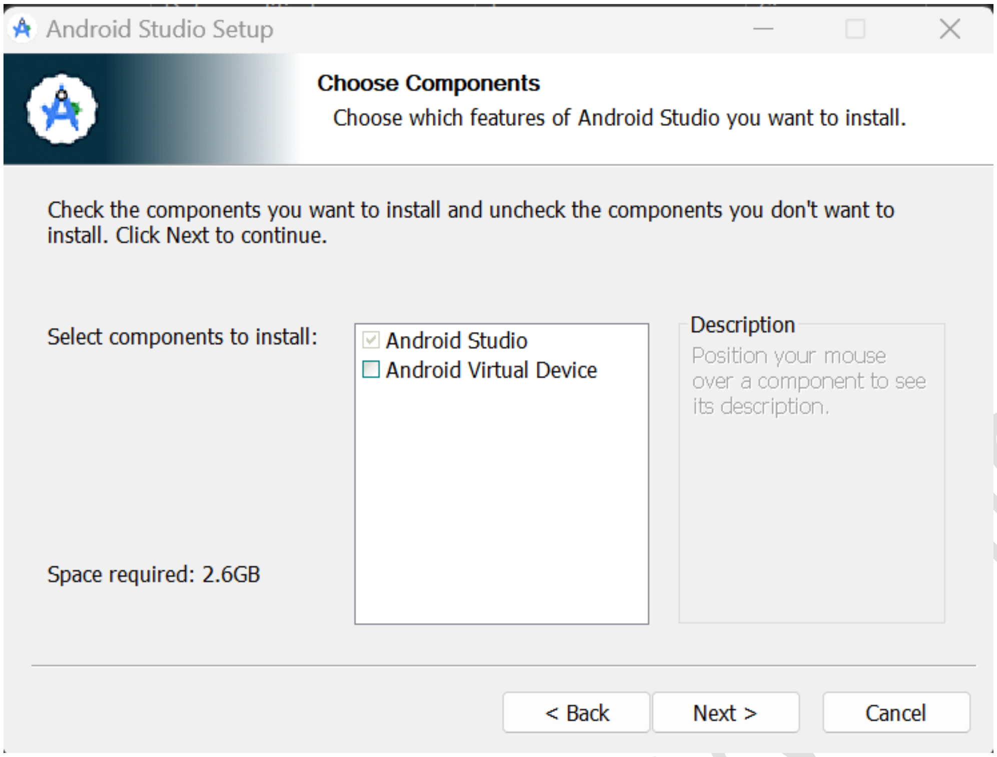
Welcome to Android Studio Setup

Setup will guide you through the installation of Android Studio.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Click Next



Android Studio Setup

Choose Components

Choose which features of Android Studio you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

☑ Android Studio

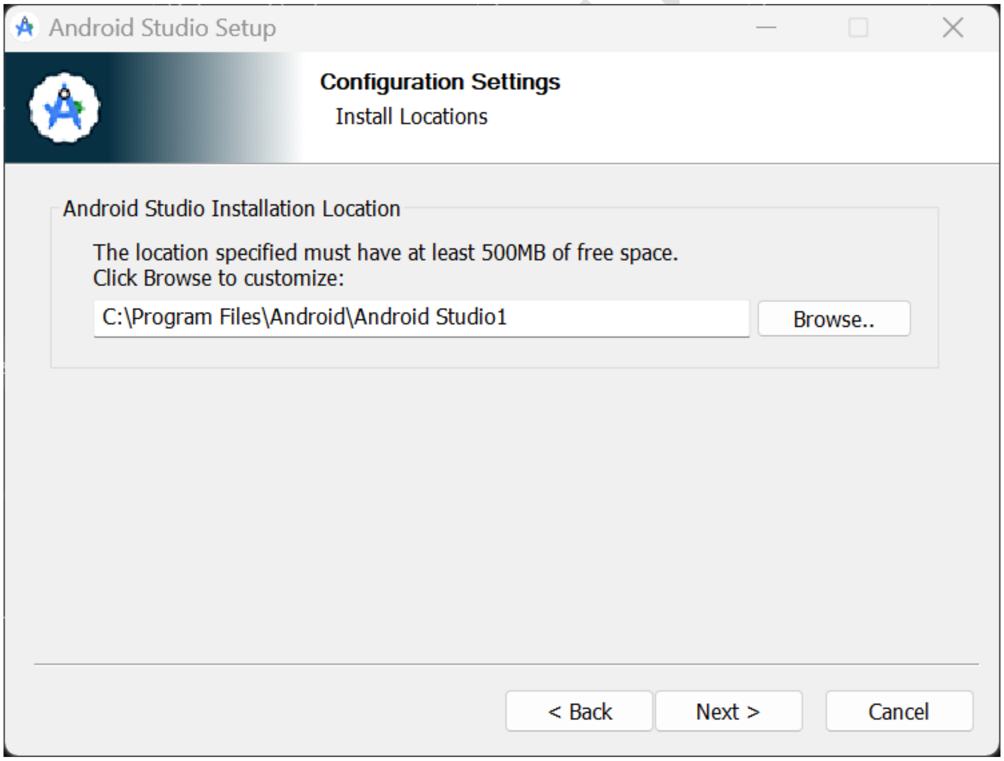
□ Android Virtual Device

Description

Position your mouse over a component to see its description.

Space required: 2.6GB

Uncheck Android Virtual Device and Click Next



Android Studio Setup

Configuration Settings

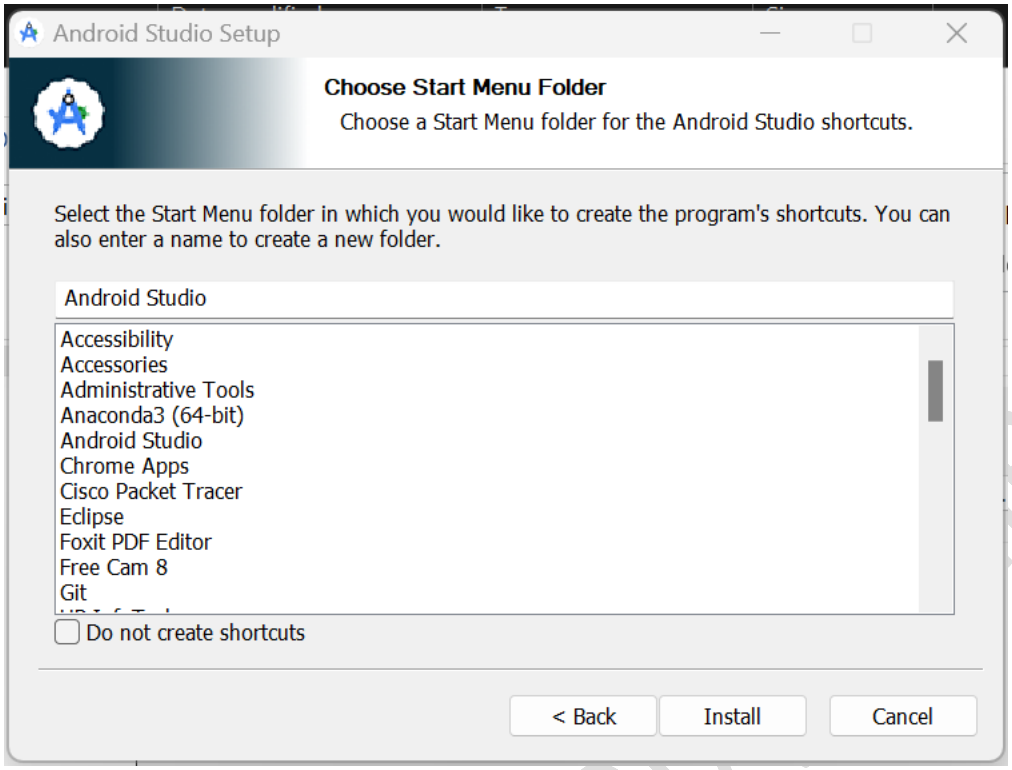
Install Locations

Android Studio Installation Location

The location specified must have at least 500MB of free space.
Click Browse to customize:

C:\Program Files\Android\Android Studio1

Keep the default path as it is and click Next



Choose Start Menu Folder

Choose a Start Menu folder for the Android Studio shortcuts.

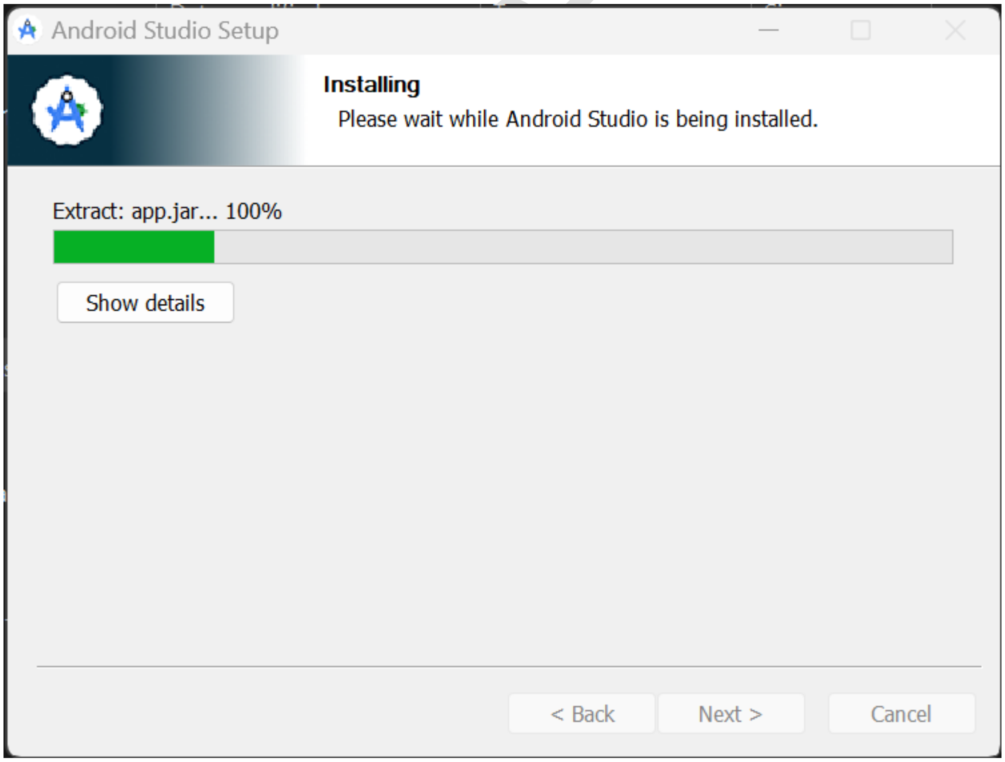
Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

Android Studio

Accessibility
Accessories
Administrative Tools
Anaconda3 (64-bit)
Android Studio
Chrome Apps
Cisco Packet Tracer
Eclipse
Foxit PDF Editor
Free Cam 8
Git

□ Do not create shortcuts

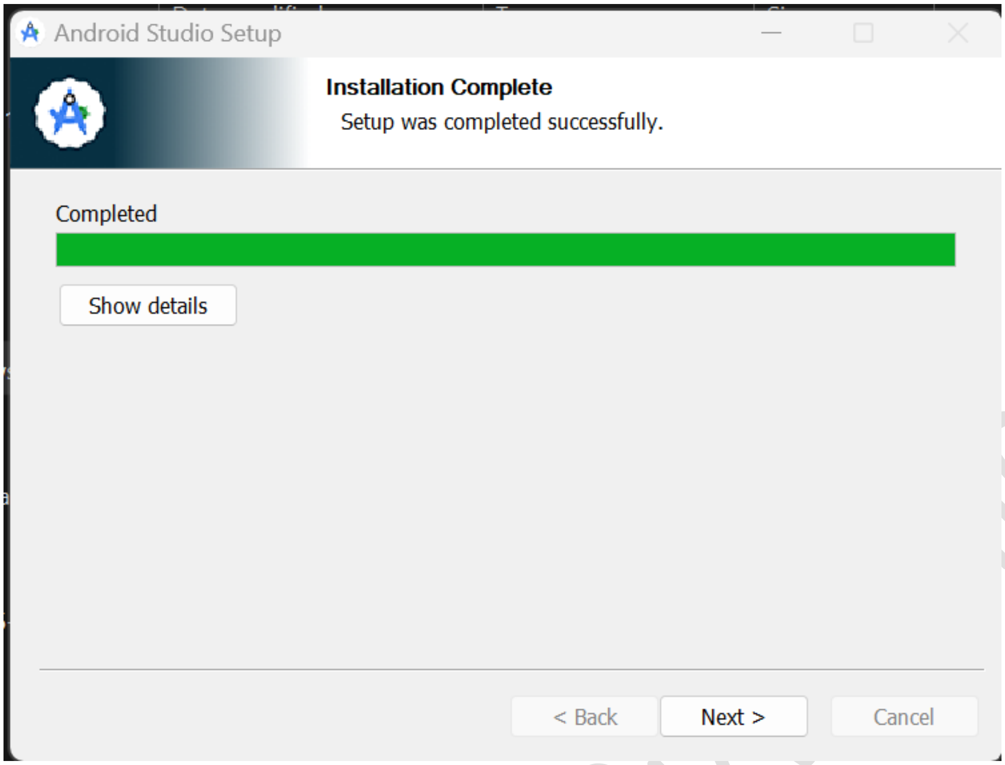
Click Install



Installing

Please wait while Android Studio is being installed.

Extract: app.jar... 100%

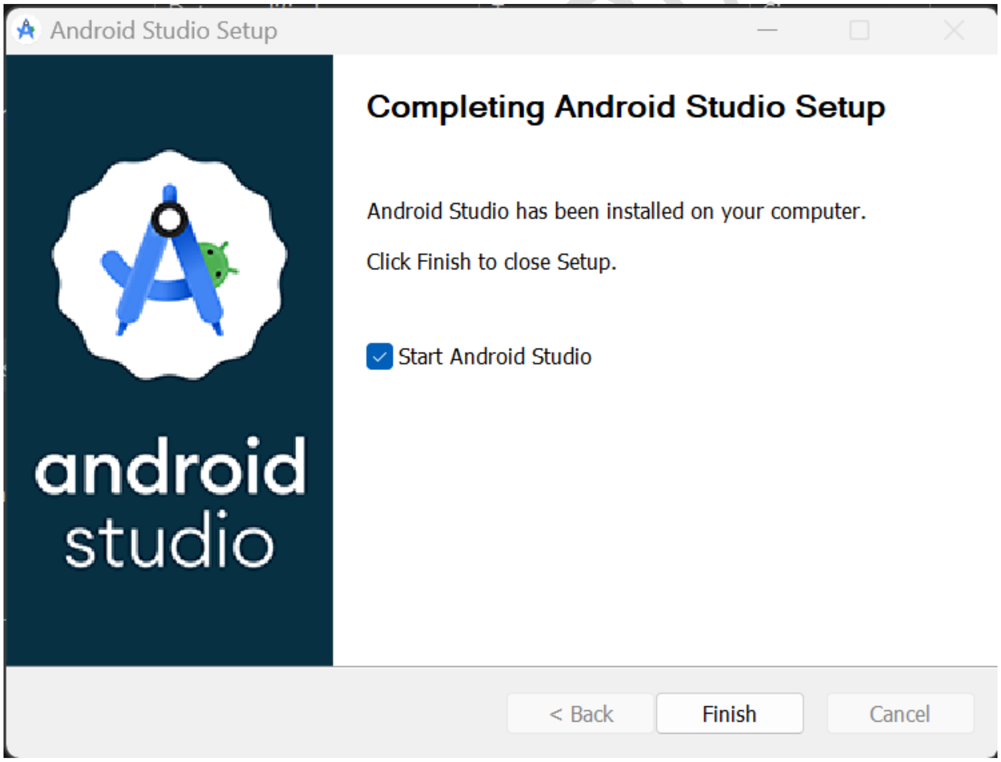


Android Studio Setup

Installation Complete

Setup was completed successfully.

Completed



Android Studio Setup

Completing Android Studio Setup

Android Studio has been installed on your computer.

Click Finish to close Setup.

☑ Start Android Studio

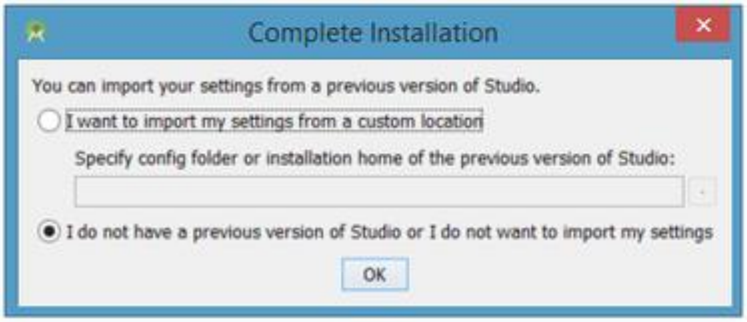
Click Finish

Running Android Studio

Android Studio presents a splash screen when it starts running:

Figure 7. Android Studio's start screen

On your first run, you'll be asked to respond to several configuration-oriented dialog boxes. The first dialog box focuses on importing settings from any previously installed version of Android Studio.



You can import your settings from a previous version of Studio.

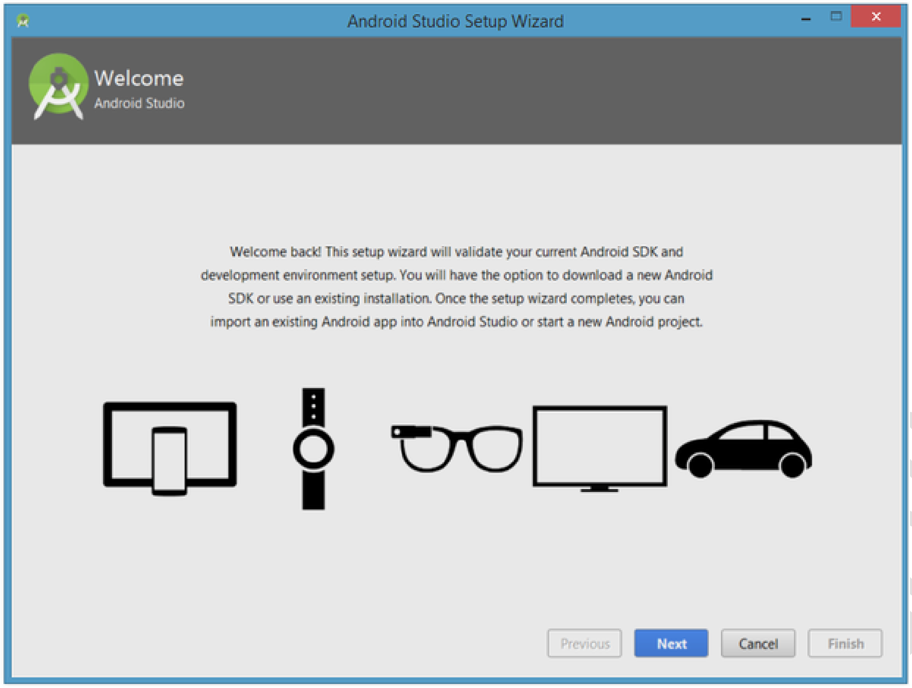
○ I want to import my settings from a custom location

Specify config folder or installation home of the previous version of Studio:

● I do not have a previous version of Studio or I do not want to import my settings

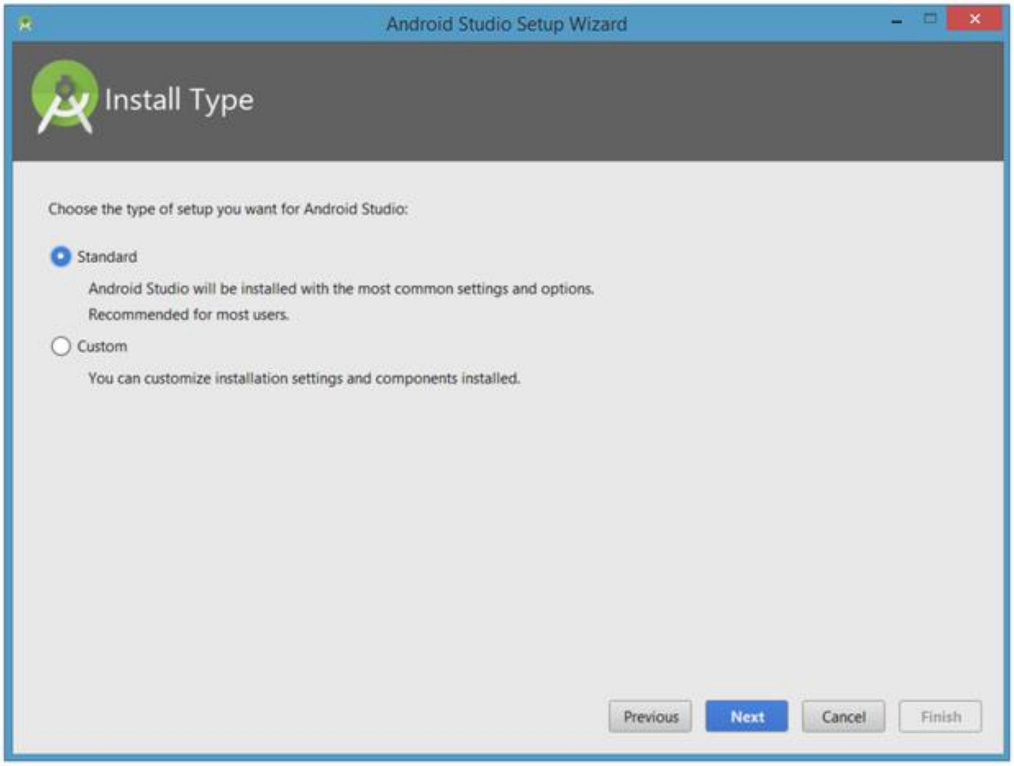
Figure 8. Import settings

If you don't have a previously installed version, you can just keep the default settings and click OK. Android Studio will respond with a slightly enhanced version of the splash screen, followed by the Android Studio Setup Wizard dialog box:



Welcome back! This setup wizard will validate your current Android SDK and development environment setup. You will have the option to download a new Android SDK or use an existing installation. Once the setup wizard completes, you can import an existing Android app into Android Studio or start a new Android project.

Figure 9. Validate your Android SDK and development environment setup
When you click Next, the setup wizard invites you to select an installation type for your SDK components. Keep the default standard setting.



Choose the type of setup you want for Android Studio:

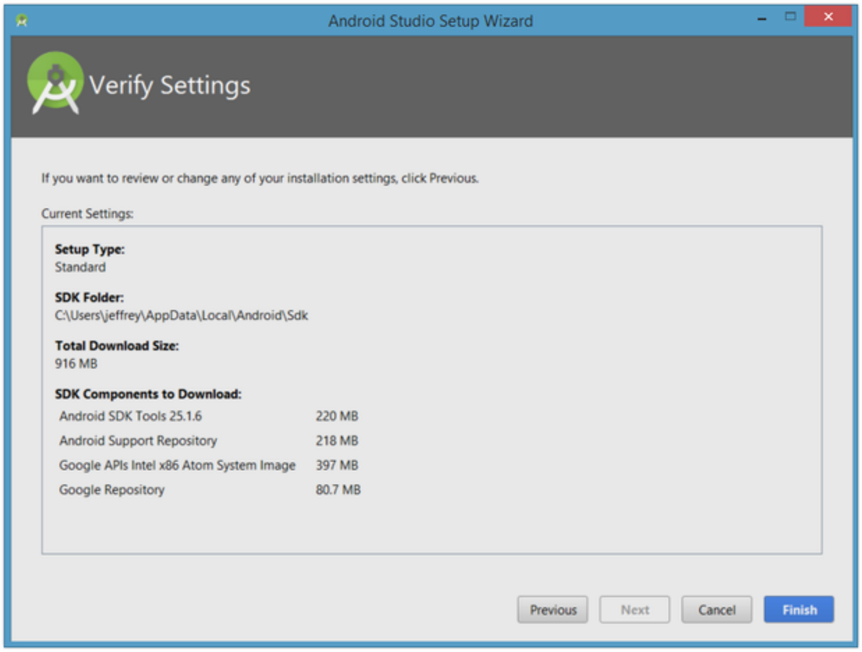
● Standard

Android Studio will be installed with the most common settings and options.
Recommended for most users.

○ Custom

You can customize installation settings and components installed.

Figure 10. Choose an installation type
Click Next and verify your settings, then click Finish to continue.



Verify Settings

If you want to review or change any of your installation settings, click Previous.

Current Settings:

Setup Type:
Standard

SDK Folder:
C:\Users\jeffrey\AppData\Local\Android\Sdk

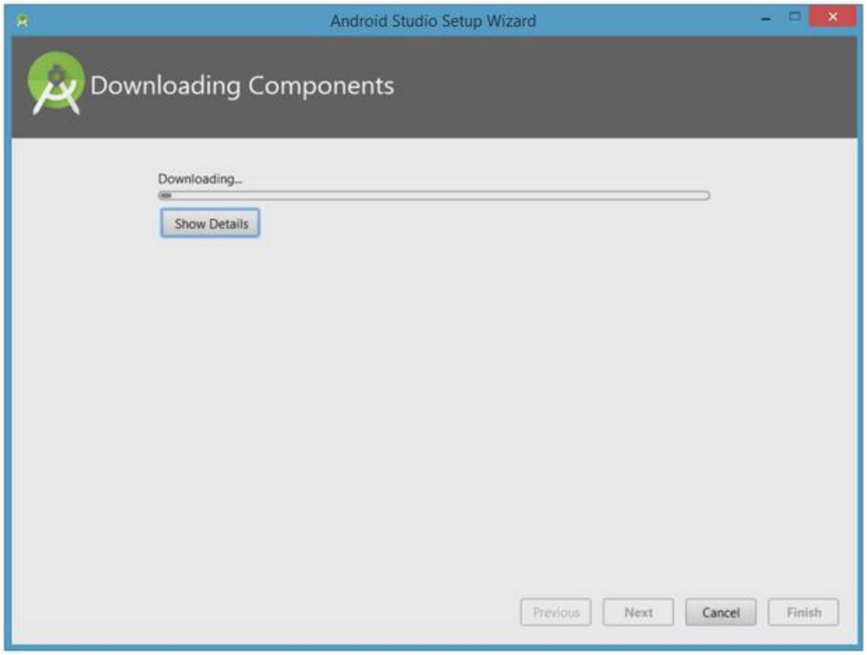
Total Download Size:
916 MB

SDK Components to Download:

Android SDK Tools 25.1.6	220 MB
Android Support Repository	218 MB
Google APIs Intel x86 Atom System Image	397 MB
Google Repository	80.7 MB

Figure 11. Review settings

The wizard will download and unzip various components. Click Show Details if you want to see more information about the archives being downloaded and their contents.

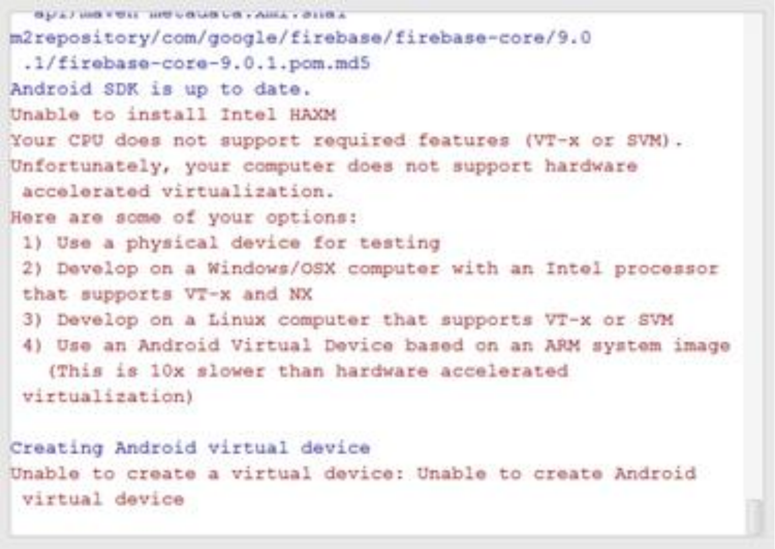


Downloading Components

Downloading...

Figure 12. The wizard downloads and unzips Android Studio components

If your computer isn't Intel based, you might get an unpleasant surprise after the components have completely downloaded and unzipped:



```
 api/maven-metadata.xml.sha1
m2repository/com/google/firebase/firebase-core/9.0
 .1/firebase-core-9.0.1.pom.md5
Android SDK is up to date.
Unable to install Intel HAXM
Your CPU does not support required features (VT-x or SVM).
Unfortunately, your computer does not support hardware
 accelerated virtualization.
Here are some of your options:
 1) Use a physical device for testing
 2) Develop on a Windows/OSX computer with an Intel processor
 that supports VT-x and NX
 3) Develop on a Linux computer that supports VT-x or SVM
 4) Use an Android Virtual Device based on an ARM system image
   (This is 10x slower than hardware accelerated
 virtualization)

Creating Android virtual device
Unable to create a virtual device: Unable to create Android
 virtual device
```

Figure 13. Intel-based hardware acceleration is unavailable
Your options are to either put up with the slow emulator or use an Android device to speed up development.

Finally, click Finish to complete the wizard. You should see the Welcome to Android Studio dialog box:

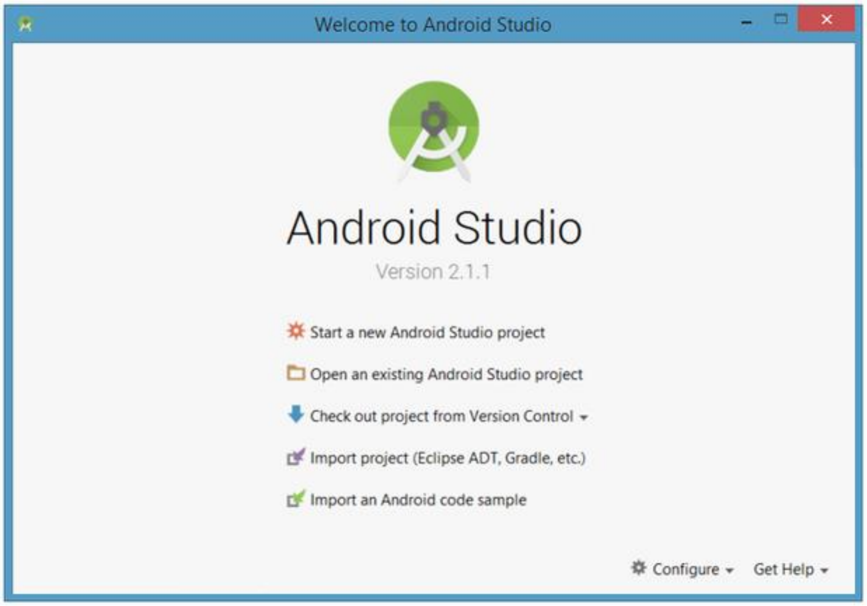

Welcome to Android Studio
Android Studio
Version 2.1.1
Start a new Android Studio project
Open an existing Android Studio project
Check out project from Version Control
Import project (Eclipse ADT, Gradle, etc.)
Import an Android code sample
Configure
Get Help

Figure 14. Welcome to Android Studio
You'll use this dialog to start up a new Android Studio project, work with an existing project, and more. You can access it anytime by double-clicking the Android Studio shortcut on your desktop.

Creating a Hello World mobile app using Android Studio

Run Android Studio. When the Welcome to Android Studio dialog box appears, click Start a new Android Studio project. Android Studio will respond with the New Project dialog box shown in Figure.

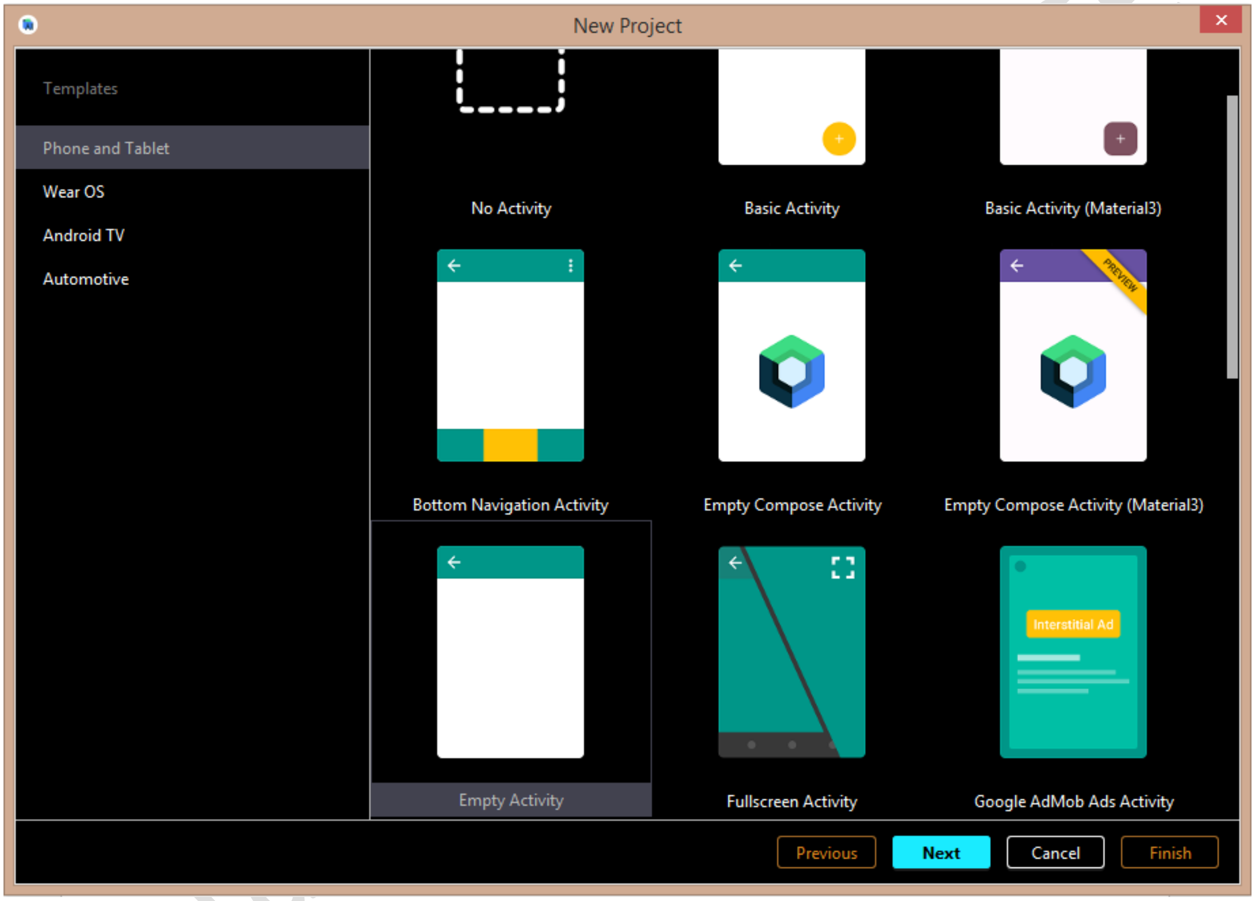
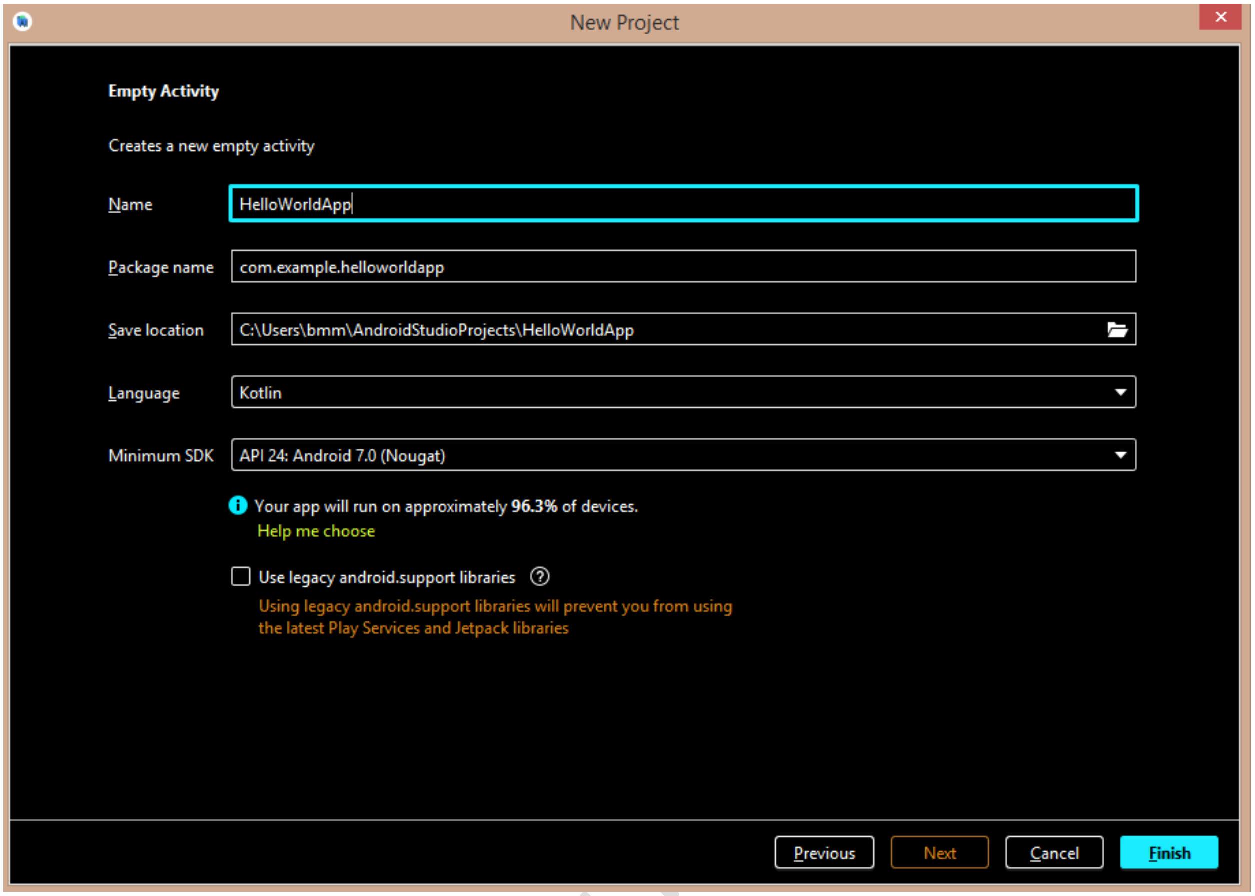


Figure 15. Create a new project
Select Empty Activity and Click Next



Empty Activity

Creates a new empty activity

Name: HelloWorldApp

Package name: com.example.helloworldapp

Save location: C:\Users\bmm\AndroidStudioProjects\HelloWorldApp

Language: Kotlin

Minimum SDK: API 24: Android 7.0 (Nougat)

Your app will run on approximately **96.3%** of devices.

Help me choose

□ Use legacy android.support libraries

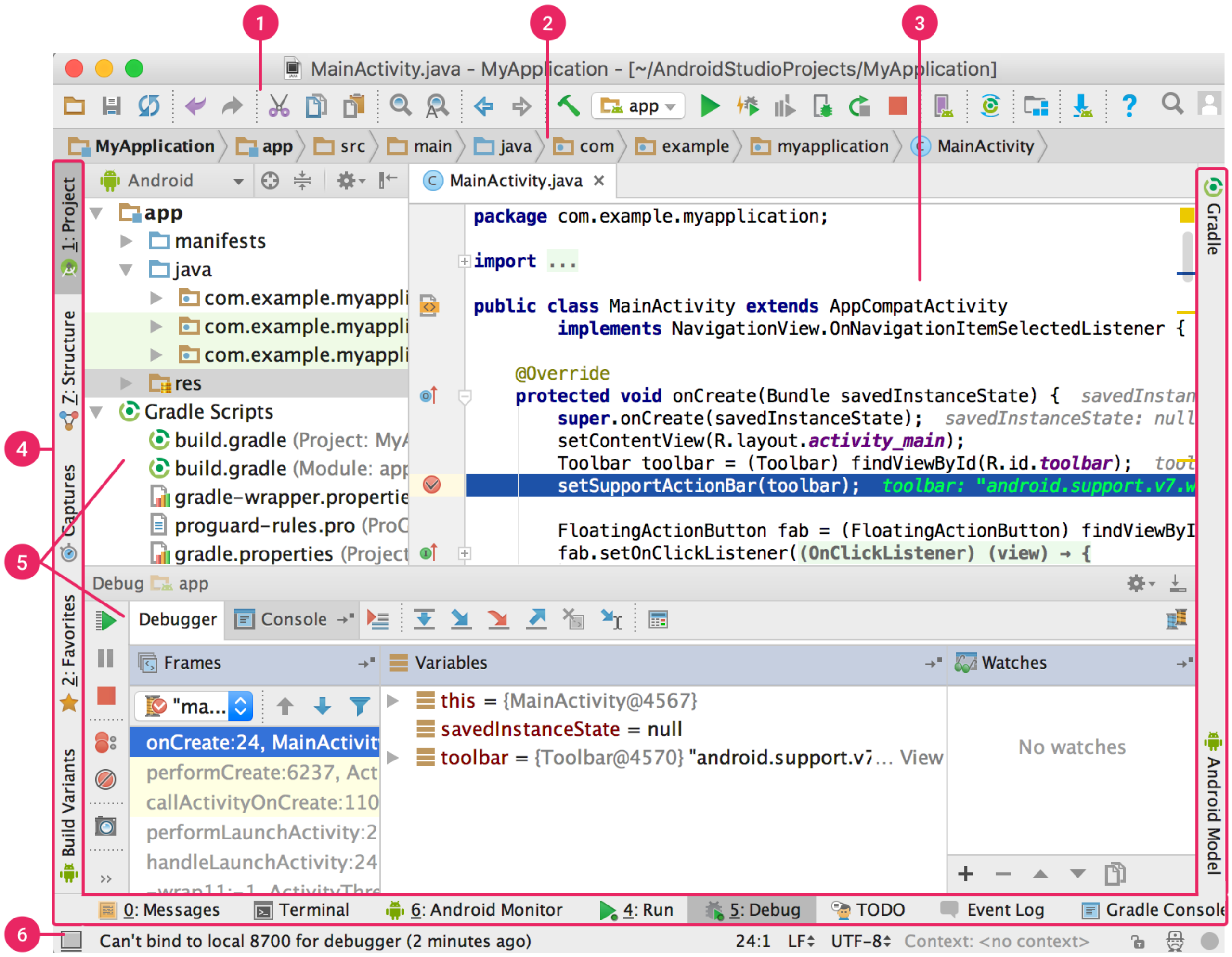
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Enter *the Name as "HelloWorldApp"* and company domain name (if required, or else keep it to default that appears in this box). Keep the project location as it is. Select Language as Kotlin. Select suitable Minimum SDK (Select the API level that is supported by your devices and your Android Studio version).

Now click Finish to complete this step. Android Studio will respond that it is creating the project, then take you to the project workspace.

The User Interface

The Android Studio main window is made up of several logical areas identified in the following figure .



```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

    @Override
    protected void onCreate(Bundle savedInstanceState) {  savedInstan
        super.onCreate(savedInstanceState);  savedInstanceState: null
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);  tool
        setSupportActionBar(toolbar);  toolbar: "android.support.v7.w

        FloatingActionButton fab = (FloatingActionButton) findViewByI
        fab.setOnClickListener((OnClickListener) (view) → {
```

```
onCreate:24, MainActivit
performCreate:6237, Act
callActivityOnCreate:110
performLaunchActivity:2
handleLaunchActivity:24
```

```
this = {MainActivity@4567}
savedInstanceState = null
toolbar = {Toolbar@4570} "android.support.v7... View
```

```
Can't bind to local 8700 for debugger (2 minutes ago)
```

Figure 3. The Android Studio main window.

1. The **toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the **Project** window.
3. The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
5. The **tool windows** give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.

6. The **status bar** displays the status of your project and the IDE itself, as well as any warnings or messages.

You can organize the main window to give yourself more screen space by hiding or moving toolbars and tool windows. You can also use keyboard shortcuts to access most IDE features.

At any time, you can search across your source code, databases, actions, elements of the user interface, and so on, by double-pressing the Shift key, or clicking the magnifying glass in the upper right-hand corner of the Android Studio window. This can be very useful if, for example, you are trying to locate a particular IDE action that you have forgotten how to trigger.

Accessing AVD Manager or SDK Manager from menu and tool bar

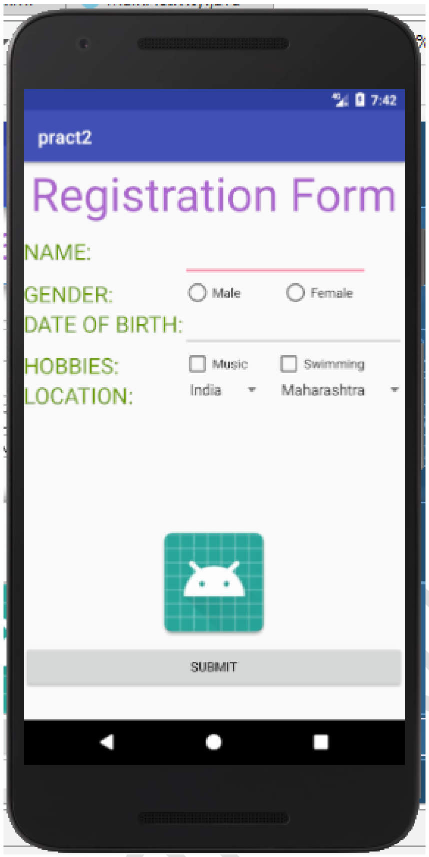
To access the traditional AVD Manager or SDK Manager, select Android from the Tools menu followed by AVD Manager or SDK Manager from the resulting pop-up menu (or click their tool bar icons).
The project window is organized into a tree whose main branches are App and Gradle Scripts. The App branch is further organized into manifests, java, and res subbranches:

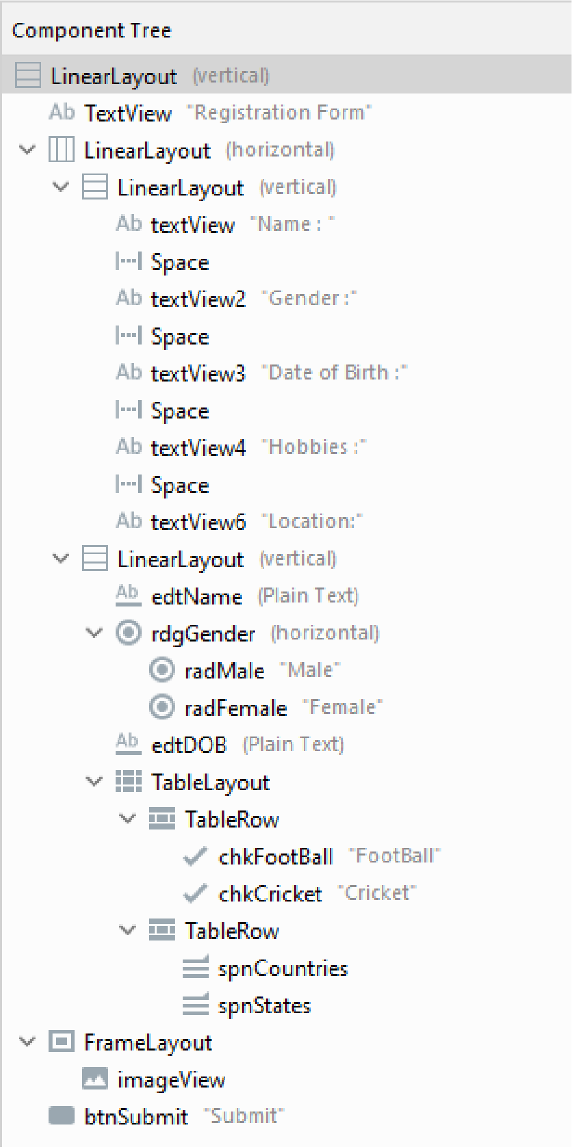
- **manifests** stores AndroidManifest.xml, which is an XML file that describes the structure of an Android app. This file also records permission settings (where applicable) and other details about the app.
- **java** stores an app's Java source files according to a package hierarchy, which is ca.javajeff.w2a in this example.
- **res** stores an app's resource files, which are organized into drawable, layout, mipmap, and values subbranches:
- o **drawable**: an initially empty location in which to store an app's artwork
- o **layout**: a location containing an app's layout files; initially, main.xml(the main activity's layout file) is stored here
- o **mipmap**: a location containing various ic_launcher.png files that store launcher screen icons of different resolutions
- o **values**: a location containing colors.xml, dimens.xml, strings.xml, and styles.xml

The Gradle Scripts branch identifies various `.gradle` (such as `build.gradle`) and `.properties` (such as `local.properties`) files that are used by the Gradle-based build system.

Attach Practical 1 as it is

Practical 2- Create an android app with Interactive User Interface using Layouts and views and their response to events/ user interaction





```
Component Tree
LinearLayout (vertical)
  Ab TextView "Registration Form"
  LinearLayout (horizontal)
    LinearLayout (vertical)
      Ab textView "Name : "
      Space
      Ab textView2 "Gender :"
      Space
      Ab textView3 "Date of Birth :"
      Space
      Ab textView4 "Hobbies :"
      Space
      Ab textView6 "Location:"
    LinearLayout (vertical)
      Ab edtName (Plain Text)
      rdgGender (horizontal)
        radMale "Male"
        radFemale "Female"
      Ab edtDOB (Plain Text)
      TableLayout
        TableRow
          chkFootBall "FootBall"
          chkCricket "Cricket"
        TableRow
          spnCountries
          spnStates
  FrameLayout
    imageView
  btnSubmit "Submit"
```

Steps to make layout:-

1. Right click on ConstraintLayout -> Convert View -> Select LinearLayout
2. Right click on LinearLayout -> LinearLayout -> Convert Orientation to vertical
3. Add a TextView set attributes in Attributes window => text-Registration Form, textStyle-bold true, textAlignment- center, textColor- "@android:color/holo_green_dark, textSize=24sp
4. Add LinearLayout (horizontal)

5. Add 2 LinearLayout(vertical)
6. In first LinearLayout(vertical) add 5 TextViews for Name,Gender,Date of Birth, Hobbies and Location. Give text property of each accordingly and change textSize and textColor. Put Space View in between the TextViews
7. In second LinearLayout(vertical)
    a. add PlainText (for name) id-edtName text-(blank)
    b. add Radiogroup id-rdgGender orientation-horizontal
    c. Add two RadioButtons within the radiogroup
    d. First radiobutton id-radMale text-Male
    e. Second radiobutton id-radFemale text-Female
8. Add a TextView id-edtDOB text-(blank)
9. Add TableLayout and drag TableRow within it.Keep 2 TableRows remove the rest
    a. Right click on first TableRow ->LinearLayout ->Convert Orientation to horizontal
    b. Repeat for second TableRow
    c. In the first TableRow add two CheckBoxes
    d. First Checkbox id-chkFootball text-Football
    e. Second Checkbox id-chkCricket text-Cricket
    f. In the second TableRow add two Spinners
    g. First Spinner id-spnCountries
    h. Second Spinner id-spnStates
10. In the main LinearLayout(Vertical) add FrameLayout
    a. Add ImageView in FrameLayout choose an image
11. In the main LinearLayout(Vertical) below FrameLayout add Button id-Submit onClick-onSubmitClick

activity_main.xml

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Registration Form"
        android:textAlignment="center"
        android:textColor="@android:color/holo_green_dark"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="222dp"
        android:orientation="horizontal">

        <LinearLayout
            android:layout_width="198dp"
            android:layout_height="match_parent"
            android:orientation="vertical">

            <TextView
                android:id="@+id/textView"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Name : "
                android:textColor="@color/teal_200"
                android:textSize="18sp" />

            <Space
                android:layout_width="match_parent"
                android:layout_height="16dp" />

            <TextView
                android:id="@+id/textView2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Gender :"
                android:textColor="@color/teal_200"
                android:textSize="18sp" />

            <Space
                android:layout_width="match_parent"
                android:layout_height="12dp" />

            <TextView
                android:id="@+id/textView3"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Date of Birth :"
                android:textColor="@color/teal_200"
                android:textSize="18sp" />

            <Space
                android:layout_width="match_parent"
                android:layout_height="20dp" />

            <TextView
                android:id="@+id/textView4"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Hobbies :"
                android:textColor="@color/teal_200"
                android:textSize="18sp" />

            <Space
                android:layout_width="match_parent"
                android:layout_height="15dp" />

            <TextView
                android:id="@+id/textView6"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Location:"
                android:textColor="@color/teal_200"
                android:textSize="18sp" />
        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical">


            <EditText
                android:id="@+id/edtName"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:ems="10"
                android:inputType="textPersonName" />

            <RadioGroup
                android:id="@+id/rdgGender"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal">

                <RadioButton
                    android:id="@+id/radMale"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Male" />

                <RadioButton
                    android:id="@+id/radFemale"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Female" />

            </RadioGroup>

            <EditText
                android:id="@+id/edtDOB"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:ems="10"
                android:inputType="textPersonName" />

            <TableLayout
                android:layout_width="match_parent"
                android:layout_height="71dp"
                android:orientation="vertical">

                <TableRow
                    android:layout_width="match_parent"
                    android:layout_height="31dp">

                    <CheckBox
                        android:id="@+id/chkFootBall"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="FootBall" />

                    <CheckBox
                        android:id="@+id/chkCricket"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="Cricket" />
                </TableRow>

                <TableRow
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content">

                    <Spinner
                        android:id="@+id/spnCountries"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content" />

                    <Spinner
                        android:id="@+id/spnStates"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content" />
                </TableRow>

            </TableLayout>
        </LinearLayout>
    </LinearLayout>

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="218dp">

        <ImageView
            android:id="@+id/imageView"
            android:layout_width="match_parent"
            android:layout_height="222dp"
            tools:srcCompat="@drawable/flowers" />

    </FrameLayout>

    <Button
        android:id="@+id/btnSubmit"
        android:layout_width="match_parent"
        android:layout_height="79dp"
        android:onClick="onSubmitClick"
        android:text="Submit"
        android:textAlignment="center"
        android:textSize="24sp" />

</LinearLayout>
```

Strings.xml

```
<resources>
    <string name="app_name">pract2</string>
    <string-array name="states">
        <item>Maharashtra</item>
        <item>Telangana</item>
        <item>Gujarat</item>
        <item>Rajasthan</item>
    </string-array>
</resources>
```

MainActivity.kt

```
package com.example.sycskotlinpract2

import android.app.Activity
import android.os.Bundle
import android.view.View
import android.widget.*

class MainActivity : Activity(), AdapterView.OnItemSelectedListener {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val spnCountries = findViewById<Spinner>(R.id.spnCountries)
        spnCountries.onItemSelectedListener = this
        val arrCountries = arrayOf("India", "Bangladesh", "Iran", "Malaysia")
        val adapter = ArrayAdapter(
            this,
            android.R.layout.simple_spinner_item, arrCountries
        )
        spnCountries.adapter = adapter
    }

    override fun onItemSelected(parent: AdapterView<*>, view: View, position:
Int, id: Long) {
        val spnStates = findViewById<Spinner>(R.id.spnStates)
        val str = parent.getItemAtPosition(position).toString()
        Toast.makeText(this, "Selected :$str", Toast.LENGTH_SHORT).show()
        if (str === "India") {
            val adap = ArrayAdapter(
                this,
                android.R.layout.simple_spinner_item,
resources.getStringArray(R.array.states)
            )

adap.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
            spnStates.adapter = adap
        } else {
            spnStates.adapter = null
        }
    }

    override fun onNothingSelected(parent: AdapterView<*>?) {}
    fun onSubmitClick(view: View?) {
        val chkCricket = findViewById<CheckBox>(R.id.chkCricket)
        val chkFootball = findViewById<CheckBox>(R.id.chkFootBall)
        val radMale = findViewById<RadioButton>(R.id.radMale)
        val radFemale = findViewById<RadioButton>(R.id.radFemale)
        val edtName = findViewById<EditText>(R.id.edtName)
        val edtDOB = findViewById<EditText>(R.id.edtDOB)
        var strHobbies = ""
        var strSalutation = ""
        if (chkCricket.isChecked) strHobbies = "Cricket\n"
        if (chkFootball.isChecked) strHobbies = "Football\n"
        if (radMale.isChecked) strSalutation = "Mr."
        if (radFemale.isChecked) strSalutation = "Ms."
        Toast.makeText(
            this, "Thank you for registration"+
strSalutation+edtName.text+"\nYour Date of Birth is"+edtDOB.text+
                    "\nYour Hobbies are"+strHobbies,
Toast.LENGTH_LONG).show()
    }
}
```

To compile the project
Build-> Make Project

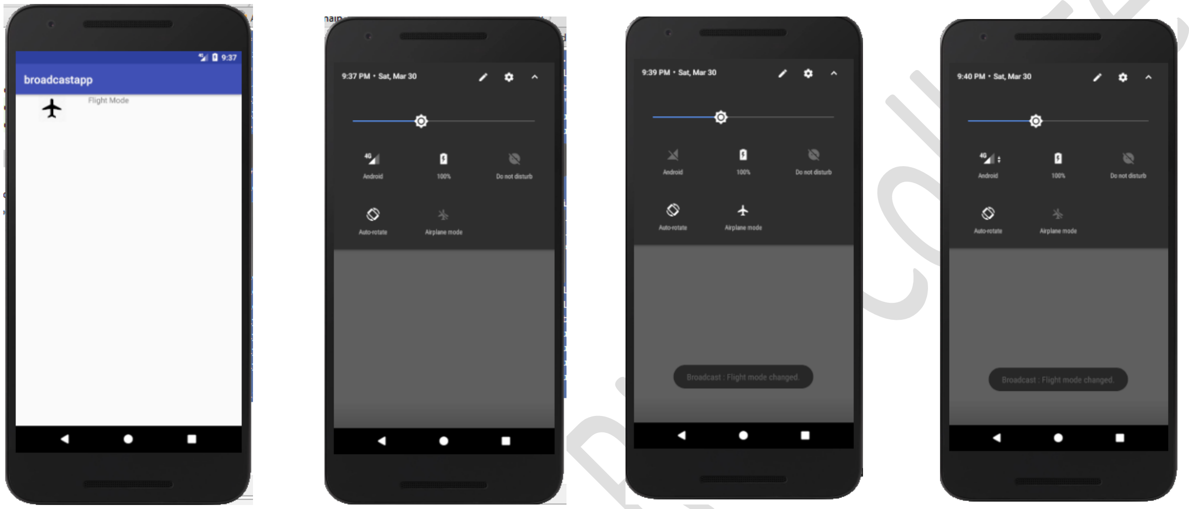
To make APK
Build->Build Bundles/APKs-> Build APKs
Click on Locate

In the journal attach activity_main.xml file code
Write Strings.xml and MainActivity.kt

Practical 3
Create an Android application to demonstrate the use of Broadcast listeners.

Broadcast Receivers

Apps can receive and android Broadcast in two ways: through manifest-declared receivers and context-registered receivers. In this example, we are applying both the ways



activity_main.xml

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

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="133dp"
        android:layout_height="50dp"
        app:srcCompat="@drawable/airplanemode" />

    <TextView
        android:layout_width="210dp"
        android:layout_height="57dp"
        android:text="Flight Mode"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</LinearLayout>
```

Right click on app-> New -> Other -> Broadcast Receiver

MyReceiver.kt

```
package com.example.admin.broadcastapp

import android.content.BroadcastReceiver
import android.content.Context
import android.content.Intent
import android.widget.Toast

class MyReceiver : BroadcastReceiver() {

    override fun onReceive(context: Context, intent: Intent) {
      Toast.makeText(context,”Broadcast : Flight mode changed",
Toast.LENGTH_LONG).show()
    }
}
```

<receiver> tag is added in the AndroidManifest.xml file. Add the <intent-filter> and <action> tags under the <receiver> tag

AndroidManifest.xml

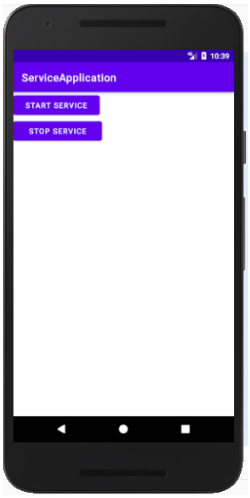
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.admin.broadcastapp">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.AIRPLANE_MODE"/>
            </intent-filter>
        </receiver>
    </application>

</manifest>
```

MainActivity.kt

```
package com.example.admin.broadcastapp
import android.content.Context
import android.content.Intent
import android.content.IntentFilter
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
       val receiver:BroadcastReceiver=MyReceiver()
       val filter:IntentFilter=IntentFilter()
       filter.addAction("android.intent.action.AIRPLANE_MODE")
       registerReceiver(receiver,filter)

    }
}
```

(Write MyReceiver.kt file and the highlighted part of AndroidManifest.xml file. Stick rest of the code and screenshots)

Practical 4:-Create an Android application to create and use services.



In res/layout/activity_main.xml file add two buttons and make changes to their properties as shown below

activity_main.xml

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


    <Button
        android:id="@+id/btnStartService"
        android:layout_width="162dp"
        android:layout_height="wrap_content"
        android:onClick="StartService"
        android:text="START SERVICE" />

    <Button
        android:id="@+id/btnStopService"
        android:layout_width="166dp"
        android:layout_height="wrap_content"
        android:onClick="StopService"
        android:text="STOP SERVICE" />
</LinearLayout>
```

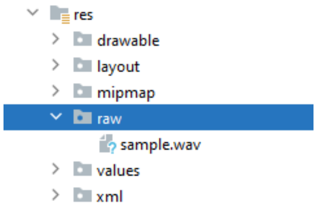
Adding a sound file to the project

Right click on res folder and select New-> Folder-> Raw resources folder

This will add a raw folder in the res folder

Right click on res->New->Folder->Raw Resources Folder

Now drag and drop sample.wav file in this folder



Creating a Service

File Menu -> New-> Service -> Service

Class Name: AudioService

Language: Kotlin

Click Finish

In the AudioService class create Mediaplayer variable and import Mediaplayer class by pressing Alt+Enter on Mediaplayer

Inside the class type onStartCommand and select it to add its code. Repeat the same for onDestroy method

Type the code in onStartCommand(), onDestroy() and onBind() as shown below to play and stop the media player on the start and stop of the service respectively

AudioService.kt

```
package com.example.sycskotlinpract3bservices

import android.app.Service
import android.content.Intent
import android.media.MediaPlayer
import android.os.IBinder

class AudioService : Service() {
lateinit var mPlayer: MediaPlayer

    override fun onBind(intent: Intent): IBinder {
        //TODO("Return the communication channel to the service.")
        throw UnsupportedOperationException("Not yet implemented");
    }

    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        mPlayer= MediaPlayer.create(this,R.raw.sample)
        mPlayer.start()
        return super.onStartCommand(intent, flags, startId);
    }

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

In MainActivity.kt file add two functions StartService() and StopService() as shown below

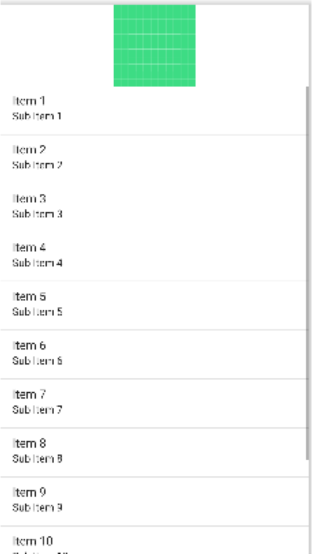
MainActivity.kt

```
package com.example.sycskotlinpract3bservices

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
    fun StartService(V: View){
        var i: Intent
        i=Intent(this,AudioService::class.java)
        startService(i)
    }
    fun StopService(V: View){
        var i: Intent
        i=Intent(this,AudioService::class.java)
        stopService(i)
    }
}
```

Add the names of StartService and StopService functions in the onClick property of btnStartService and btnStopService buttons

(Write MainActivity.kt and AudioService.kt file. Stick code of activity_main.xml and screenshots)

Practical 5:- Create an Android app to demonstrate different types of Menus



Item 1
Sub Item 1

Item 2
Sub Item 2

Item 3
Sub Item 3

Item 4
Sub Item 4

Item 5
Sub Item 5

Item 6
Sub Item 6

Item 7
Sub Item 7

Item 8
Sub Item 8

Item 9
Sub Item 9

Item 10

Options Menu

Right click on app-> New-> Android Resource File

File Name: optionmenus

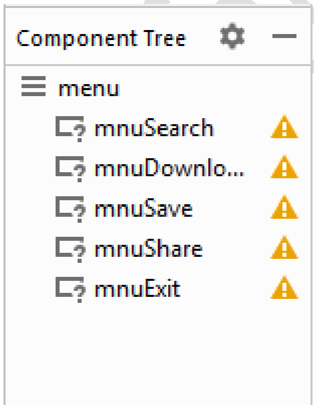
Resource Type: Menu

Directory Name: menu

Click OK

This creates menu folder under /res folder and menu.xml file under menu folder

Drag 5 MenuItems under menu and set their ID and Title appropriately



Component Tree

menu
- mnuSearch
- mnuDownlo...
- mnuSave
- mnuShare
- mnuExit

Optionmenus.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/mnuSearch"
        android:title="Search" />
    <item
        android:id="@+id/mnuDownload"
        android:title="Download" />
    <item
        android:id="@+id/mnuSave"
        android:title="Save" />
    <item
        android:id="@+id/mnuShare"
        android:title="Share" />
    <item
        android:id="@+id/mnuExit"
        android:title="Exit" />
</menu>
```

PopUp Menu

Similarly make menu file popupMenu.xml having the following menus

Cut

Copy

paste

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/mnuSearch"
        android:title="Search" />
    <item
        android:id="@+id/mnuDownload"
        android:title="Download" />
    <item
        android:id="@+id/mnuSave"
        android:title="Save" />
</menu>
```

Strings.xml

```
<resources>
    <string name="app_name">Menuapp</string>
    <string-array name="fruits">
        <item>Apple</item>
        <item>Grapes</item>
        <item>Mango</item>
    </string-array>
</resources>
```

MainActivity.kt

In MainActivity.kt file press Ctrl+O and override the functions `onCreateOptionsMenu, onOptionsItemSelected, onCreateContextMenu and onContextItemSelected`

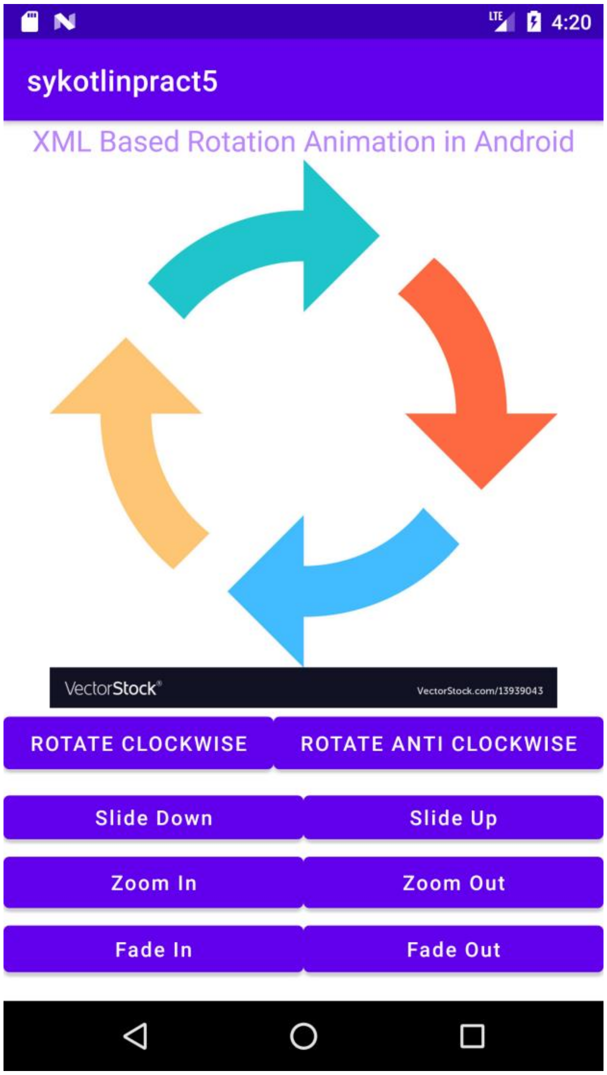
Now type the following code in onCreate() method and above functions

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

//Register listview for context menu
val listView=findViewById<ListView>(R.id.listView)
registerForContextMenu(listView)


//attach popup menu to image using OnClickListener . Inflate the popupmenu,
//set its OnMenuItemClickListener and show the popupmenu

val img = findViewById<ImageView>(R.id.image)

img.setOnClickListener {

      val popupMenu: PopupMenu = PopupMenu(this,img)

      popupMenu.menuInflater.inflate(R.menu.popupmenu, popupMenu.menu)

popupMenu.setOnMenuItemClickListener(PopupMenu.OnMenuItemClickListener{

item->

Toast.makeText(this,"You Clicked : " + item.title, Toast.LENGTH_SHORT).show()

      true

})
popupMenu.show()

}


 override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.optionmenu,menu)
        return super.onCreateOptionsMenu(menu)
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
when (item.itemId){
      R.id.mnuSearch ->
      Toast.makeText(this,"Search Selected",Toast.LENGTH_SHORT).show()
      R.id.mnuDownload ->
      Toast.makeText(this,"Download Selected",Toast.LENGTH_SHORT).show()
      R.id.mnuSave ->
      Toast.makeText(this,"Save Selected",Toast.LENGTH_SHORT).show()
      R.id.mnuShare ->
      Toast.makeText(this,"Share Selected",Toast.LENGTH_SHORT).show()
      R.id.mnuExit ->
      {     Toast.makeText(this,"Exit Selected",Toast.LENGTH_SHORT).show()
            finish()
      }
   }
   return super.onOptionsItemSelected(item)
 }
```

```
override fun onCreateContextMenu(
    menu: ContextMenu?,
    v: View?,
    menuInfo: ContextMenu.ContextMenuInfo?
) {
    val inflater = menuInflater
    inflater.inflate(R.menu.optionmenu, menu)

    super.onCreateContextMenu(menu, v, menuInfo)
}


override fun onContextItemSelected(item: MenuItem): Boolean {
   Toast.makeText(this,"Selected :"+item.title,Toast.LENGTH_SHORT).show()
    return super.onContextItemSelected(item)
}

}
```

Write all the above code in journal. For output take screenshot of your app and stick it

Practical 6:-
Create an Android application to demonstrate XML based animation



sykotlinpract5

XML Based Rotation Animation in Android

Add a New Resource file

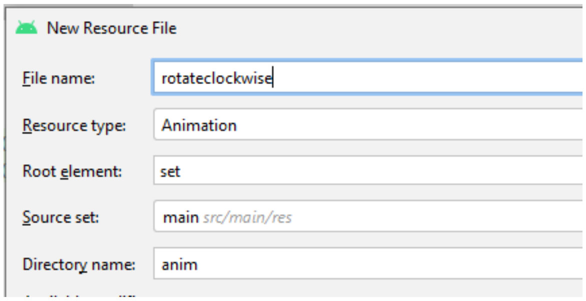
Click on app folder

File->New->Android Resource File

File name:- fade_in

Resource Type:- Animation

Similarly add other files (fade_out, Rotateclockwise , Rotateanticlockwise, Slide_down, Slide_up, Zoom_in, Zoom_out) or copy and paste in anim folder and rename the files



New Resource File

File name: rotateclockwise

Resource type: Animation

Root element: set

Source set: main src/main/res

Directory name: anim

Fade_in.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <alpha
        android:duration="1000"
        android:fromAlpha="0.1"
        android:toAlpha="1.0" />

</set>
```

Fade_out.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <alpha
        android:duration="1000"
        android:fromAlpha="1.0"
        android:toAlpha="0.1" />

</set>
```

Rotateclockwise.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <rotate
        android:fromDegrees="0"
        android:toDegrees="360"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="1000"/>

</set>
```

Rotateanticlockwise.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <rotate android:fromDegrees="360" android:toDegrees="0"
        android:pivotX="50%" android:pivotY="50%" android:duration="1000" />

</set>
```

Slide_down.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="1000"
        android:fromYDelta="-100%"
        android:toYDelta="0" />

</set>
```

Slide_up.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="1000"
        android:fromYDelta="0"
        android:toYDelta="-100%" />

</set>
```

Zoom_in.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <scale xmlns:android="http://schemas.android.com/apk/res/android"
        android:duration="1000"
        android:fromXScale="1"
        android:fromYScale="1"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="1.5"
        android:toYScale="1.5">
    </scale>

</set>
```

Zoom_out.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <scale
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:duration="1000"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="0.5"
        android:toYScale="0.5" >
    </scale>
</set>
```

Activity_main.xml

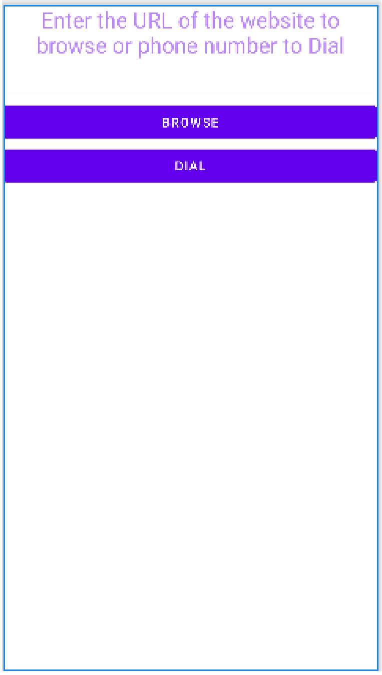
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="XML Based Rotation Animation in Android"
        android:textAlignment="center"
        android:textColor="@color/purple_200"
        android:textSize="20sp" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="376dp"
        app:srcCompat="@drawable/rotation"
        tools:srcCompat="@drawable/rotation" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="54dp"
        android:orientation="horizontal">

        <Button
            android:id="@+id/clockwise"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Rotate Clockwise" />
        <Button
            android:id="@+id/anticlockwise"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Rotate Anti Clockwise" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="42dp"
        android:orientation="horizontal">

        <Button
            android:id="@+id/slide_down"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Slide Down"
            android:textAllCaps="false" />


        <Button
            android:id="@+id/slide_up"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Slide Up"
            android:textAllCaps="false" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="47dp"
        android:orientation="horizontal">

        <Button
            android:id="@+id/zoom_in"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Zoom In"
            android:textAllCaps="false" />

        <Button
            android:id="@+id/zoom_out"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Zoom Out"
            android:textAllCaps="false" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="73dp"
        android:orientation="horizontal">

        <Button
            android:id="@+id/fade_in"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Fade In"
            android:textAllCaps="false" />

        <Button
            android:id="@+id/fade_out"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Fade Out"
            android:textAllCaps="false" />

    </LinearLayout>

</LinearLayout>
```

MainActivity.kt

```
package com.example.sykotlinpract5

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.Handler
import android.view.View
import android.view.animation.AnimationUtils
import android.widget.Button
import android.widget.ImageView

class MainActivity : AppCompatActivity() {

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

        val fadein=findViewById<Button>(R.id.fade_in)
        val fadeout=findViewById<Button>(R.id.fade_out)
        val zoomin=findViewById<Button>(R.id.zoom_in)
        val zoomout=findViewById<Button>(R.id.zoom_out)
        val slidedown=findViewById<Button>(R.id.slide_down)
        val slideup=findViewById<Button>(R.id.slide_up)
        val clockwise=findViewById<Button>(R.id.clockwise)
        val anticlockwise=findViewById<Button>(R.id.anticlockwise)
        val imageView=findViewById<ImageView>(R.id.imageView)
        fadein.setOnClickListener {
           imageView.visibility = View.VISIBLE
            val animationFadeIn = AnimationUtils.loadAnimation(this,
R.anim.fade_in)
            imageView.startAnimation(animationFadeIn)
        }


        fadeout.setOnClickListener {
            val animationFadeOut = AnimationUtils.loadAnimation(this,
R.anim.fade_out)
            imageView.startAnimation(animationFadeOut)

            imageView.visibility = View.GONE
        }


        zoom_in.setOnClickListener {
            val animationZoomIn = AnimationUtils.loadAnimation(this,
R.anim.zoom_in)
            imageView.startAnimation(animationZoomIn)
        }
        zoom_out.setOnClickListener {
            val animationZoomOut = AnimationUtils.loadAnimation(this,
R.anim.zoom_out)
            imageView.startAnimation(animationZoomOut)
        }

        slide_down.setOnClickListener {
            val animationSlideDown = AnimationUtils.loadAnimation(this,
R.anim.slide_down)
            imageView.startAnimation(animationSlideDown)
        }
        slide_up.setOnClickListener {
            val animationSlideUp = AnimationUtils.loadAnimation(this,
R.anim.slide_up)
            imageView.startAnimation(animationSlideUp)
        }
        clockwise.setOnClickListener {
            val animationclockwise = AnimationUtils.loadAnimation(this,
R.anim.rotateclockwise)
            imageView.startAnimation(animationclockwise)
        }
        anticlockwise.setOnClickListener {
            val animationanticlockwise = AnimationUtils.loadAnimation(this,
R.anim.rotateanticlockwise)
            imageView.startAnimation(animationanticlockwise)
        }
    }

}
```

Practical 7:-

Create an Android application to demonstrate implicit intents



Enter the URL of the website to browse or phone number to Dial

Activity_main.xml

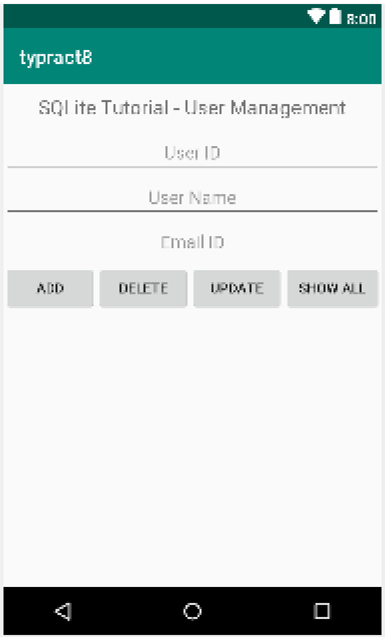
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Enter the URL of the website to browse or phone number
to Dial"
        android:textAlignment="center"
        android:textColor="@color/purple_200"
        android:textSize="24sp" />

    <EditText
        android:id="@+id/edtURL"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName" />
    <Button
        android:id="@+id/btnBrowse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Browse" />

    <Button
        android:id="@+id/btnDial"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Dial" />

</LinearLayout>
```

MainActivity.kt

```
package com.example.sykotlinpract6

import android.app.ProgressDialog
import android.content.Intent
import android.net.Uri
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import androidx.appcompat.app.AppCompatActivity
import androidx.core.content.ContextCompat.startActivity


class MainActivity : AppCompatActivity() {

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

        val edtURL = findViewById<EditText>(R.id.edtURL)
        val btnBrowse = findViewById<Button>(R.id.btnBrowse)
        val btnDial = findViewById<Button>(R.id.btnDial)

        btnBrowse.setOnClickListener {
            val pd = ProgressDialog(this)
            pd.setMessage("Searching Webpage...")
            pd.setProgressStyle(ProgressDialog.STYLE_SPINNER)
            pd.progress = 0
            pd.show()
            val s = edtURL.text.toString()
            val uri= Uri.parse(s)
            val browseIntent = Intent(Intent.ACTION_VIEW,uri)
            startActivity(browseIntent)
        }
        btnDial.setOnClickListener {
            val pd = ProgressDialog(this)
            pd.setMessage("Dialling Number...")
            pd.setProgressStyle(ProgressDialog.STYLE_SPINNER)
            pd.progress = 0
            pd.show()
            val p = edtURL.text.toString()
            val dialIntent = Intent(Intent.ACTION_DIAL)
            dialIntent.data = Uri.parse("tel:" + p)
            startActivity(dialIntent)
        }

    }
}
```

Write the code of MainActivity.kt file. Stick the rest

Practical 8:- Create a suitable Android application to store and retrieve data in the SQLite database.



Activity_main.xml

Change the Layout to LinearLayout. Set its property

Orientation: vertical

Add a TextView and set its property

textSize: 20sp

text: SQLite Tutorial- User Management

Add another LinearLayout(vertical)

Add three EditTexts (Plain Text) to this layout and set their properties:-

1) ID: edtUserId
   Hint: User ID
2) ID: edtName
   Hint: User Name
3) ID: edtEmailid
   Hint: Email ID

Add another LinearLayout(horizontal)

Add four Buttons to this layout and set their properties as:-

1) ID: btnAddUser
   Text: Add
   onClick: addUser
2) ID: btnDeleteUser
   Text: Delete
   onClick: deleteUser
3) ID: btnUpdateUser
   Text: Update
   onClick: updateUser
4) ID: btnShowAll
   Text: Show All
   onClick: showAllUsers

Now add a TextView under the main LinearLayout (vertical) and set its property

ID: txtResult

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              xmlns:app="http://schemas.android.com/apk/res-auto"
              xmlns:tools="http://schemas.android.com/tools"
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:orientation="vertical"
              android:gravity="center"
              tools:context="com.example.rafatkhan.typract8.MainActivity">

    <TextView android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="SQLite Tutorial - User Management"
              android:textSize="20dp" android:padding="10dp" />

    <LinearLayout android:layout_width="match_parent"
                  android:layout_height="wrap_content"
                  android:orientation="vertical">

        <EditText android:id="@+id/edtUserId"
                  android:hint="User ID"
                  android:gravity="center"
                  android:layout_width="match_parent"
                  android:layout_height="wrap_content" />

        <EditText android:id="@+id/edtName"
                  android:hint="User Name"
                  android:gravity="center"
                  android:layout_width="match_parent"
                  android:layout_height="wrap_content" />

        <EditText android:id="@+id/edtEmailid"
                  android:hint="Email ID"
                  android:gravity="center"
                  android:layout_width="match_parent"
                  android:layout_height="wrap_content" />
    </LinearLayout>

    <LinearLayout android:layout_width="match_parent"
                  android:layout_height="wrap_content"
                  android:orientation="horizontal">

        <Button android:id="@+id/btnAddUser"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:onClick="addUser"
                android:text="Add" />

        <Button android:id="@+id/btnDeleteUser"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:onClick="deleteUser"
                android:text="Delete" />
        <Button android:id="@+id/btnUpdateUser"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:onClick="updateUser"
                android:text="Update" />

        <Button android:id="@+id/btnShowAll"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:onClick="showAllUsers"
                android:text="Show All" />
    </LinearLayout>
    <TextView android:id="@+id/txtResult"
              android:layout_width="match_parent"
              android:layout_height="wrap_content" />

</LinearLayout>
```

DBHelper Class

Right click on app/java folder and select New-> Kotlin File/Class

Name: DBHelper

Kind: Class

Click OK

Extend class DBHelper with SQLiteOpenHelper Class

Click on red bulb and select the constructor with four parameters Context, String, Cursorfactory and Int.

Change the values to **SQLiteOpenHelper(context, DATABASE_NAME, null, 1)**

Now right click inside the class and select **Generate-> Override Methods** or Click Ctrl+O. Now select the methods onCreate and onUpgrade (Using Ctrl) and click OK. Comment ToDo lines in both.

Also add your own new functions insertData, updateData, deleteData and get() with appropriate parameters and return type as required

```
package com.example.rafatkhan.typract8

import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHelper (context: Context) :
        SQLiteOpenHelper(context, DATABASE_NAME, null, 1)
{
    companion object {
        val DATABASE_NAME="USER.DB"
        val TABLE_NAME="USERS"
        val COL_0="USERID"
        val COL_1="USERNAME"
        val COL_2="EMAILID"
    }

    override fun onCreate(db: SQLiteDatabase) {
        db.execSQL("CREATE TABLE $TABLE_NAME( USERID TEXT PRIMARY KEY , USERNAME
TEXT, EMAILID TEXT)" )
    }

    override fun onUpgrade(db: SQLiteDatabase, p1: Int, p2: Int) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME)
        onCreate(db)

    }

    fun insertData (id: String, name: String, emailid: String ){
        val db=this.writableDatabase
        val contentValues= ContentValues()
       contentValues.put(COL_0,id)
        contentValues.put(COL_1,name)
        contentValues.put(COL_2,emailid)
        db.insert(TABLE_NAME,null,contentValues)
    }

    fun updateData(id : String, name: String, emailid: String){
        val db=this.writableDatabase
        val contentValues= ContentValues()
        contentValues.put(COL_0,id)
        contentValues.put(COL_1,name)
        contentValues.put(COL_2,emailid)
        db.update(TABLE_NAME,contentValues,"USERID = ?", arrayOf(id))
    }

    fun deleteData(id: String) {
        val db=this.writableDatabase
        db.delete(TABLE_NAME, "USERID=?", arrayOf(id))
    }



    val allData: Cursor
        get(){
            val db=this.writableDatabase
            val res=db.rawQuery("SELECT * FROM " + TABLE_NAME,null)
            return res
        }

}
```

MainActivity.kt

Create an object of DBHelper class. Create functions addUser, deleteUser, updateUser and showAllUsers

```
package com.example.rafatkhan.typract8

import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {
private var dbHelper = DBHelper(this)
val btnAddUser=findViewById<Button>(R.id.btnAddUser)
val btnUpdateUser=findViewById<Button>(R.id.btnUpdateUser)
val btnDeleteUser=findViewById<Button>(R.id.btnDeleteUser)
val btnShowAll=findViewById<Button>(R.id.btnShowAll)
val edtUserId=findViewById<EditText>(R.id.edtUserId)
val edtName=findViewById<EditText>(R.id.edtName)
val edtEmailId=findViewById<EditText>(R.id.edtEmailid)
val txtResult = findViewById<TextView>(R.id.txtResult)


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

        btnAddUser.setOnClickListener(){
            var userid = edtUserId.text.toString()
            var name = edtName.text.toString()
            var emailid = edtEmailId.text.toString()
            dbHelper.insertData(userid,name, emailid)
            edtEmailId.setText("")
            this.edtName.setText("")
            this.edtUserId.setText("")
            this.txtResult.text = "Added user   "
        }
        btnUpdateUser.setOnClickListener(){
            var userid = edtUserId.text.toString()
            var name = edtName.text.toString()
            var emailid = edtEmailId.text.toString()
            dbHelper.updateData(userid,name,emailid)
            this.txtResult.text = "Updated user  "
        }

        btnDeleteUser.setOnClickListener(){
            var userid = this.edtUserId.text.toString()
            dbHelper.deleteData(userid)
            this.txtResult.text = "Deleted user  "
        }
        btnShowAll.setOnClickListener(){
            val res = dbHelper.allData
            val buffer = StringBuffer()
            if (res.count == 0) {
                txtResult.text = "No Data Found"

            }
            else
            {
                while (res.moveToNext()) {
                    buffer.append("ID : " + res.getString(0) + "\n")
                    buffer.append("Name : " + res.getString(1) + "\n")
                    buffer.append("Email ID : " + res.getString(2) + "\n")
                }
                txtResult.text = buffer.toString()
            }

        }

    }

}
```