Nagarjuna College of Engineering and Technology®

Department of Computer Science & Engineering

ANDROID PROGRAMMING

LAB MANUAL

Course Code: 21CSL582

5th Semester

<u>Prepared By:</u>

Mrs. Vidya V

Assistant Professor

Department of CSE

Laboratory Objectives: This laboratory will enable students to

1. Setting up the Android Application Development Environment, Outline the Android SDK features and the Development Framework.
2. Understand the view components and handling views.
3. Understanding activities, which are the major building blocks of app's user interface.
4. Create adaptive, responsive user interfaces that work across a wide range of devices.
5. Implementing different navigation paths through your application.

Descriptions

1. The installation procedure of the Android Studio/Java software must be demonstrated and carried out in groups.

2. Students can use the latest version of Android Studio/Java/Kotlin to execute these programs. Diagrams given are for representational purposes only, students are expected to improvise on them.

Programs List:

1	Create an application to design a Visiting Card. The Visiting card should have a company logo at the top right corner. The company name should be displayed in Capital letters, aligned to the center. Information like the name of the employee, job title, phone number, address, email, fax and the website address are to be displayed.
2	Develop an application using controls like Button and TextView, on click of Button 1 display “Welcome to NCET” and on Click of Button 2 display “Welcome to CSE”. Handle the two buttons in an Activity to demonstrate event handling using both Java and XML.
3.	Create an application to demonstrate all Activity life cycle callback methods. Display Toast message When each method invokes.
4.	Develop an android application with an Activity and accept data from the user in first Activity. On click of a Next button in first Activity transfer the data from first Activity to second Activity.
5.	Create an android application with Activity has three buttons. On click of Buton 1 open the web browser application, on click of Button 2 open the call application and on click of Button 3 open the map application.
6.	Develop an android application to Create a login Activity. It asks “username” and “password” from user. If username and password are valid, it displays Welcome message using new activity.

7.	Develop an android application to design a Simple Calculator application has two edit texts and four buttons. When you enter two numbers and click a button, the application performs the calculation for that button and displays the result.
8.	Develop a simple application with one EditText so that the user can write some text in it. Create a button called "Convert Text to Speech" that converts the user input text into voice.

INTRODUCTION TO Android Programming

Every day the new devices are incoming to the market with innovative options thanks to growing technology. The evolution of Mobile Application Development technology with new devices made our lives much easier.

In the smartphone world, simply having a running web site is not enough. Regarding a recent study, it has shown that about 45% and more of Google search happens using smartphones. The number is spectacular and there is a growth within the mobile business. Being obtainable on an internet-enabled device is needed for every and each business which has given the kicking start to mobile application development.

What is Mobile Development?

Mobile development, which is not about building phone apps, though it is a huge part of it. Actually, it’s doing any reasonably development for any kind of mobile devices such as developing apps for phones, tablets, smart watches, and every form of wearable devices that run any kind of mobile operating system.

Mobile development presents a reasonably distinctive chance for a one-person development team to build an actual, usable, significant app end-to-end during a comparatively short period. However, Mobile Apps Development represents more than just a chance for the solo-developer to create their own project as it is arguably the longer term of development, as mobile devices are getting larger and bigger parts of our lives.

Android is the dominant player in mobile development platforms space, it was a bit later participant to the game, first being released in Sept 2008, virtually a year later than iOS but it has managed to achieve a reasonably massive share of the mobile market.

Technically, Android is the mobile OS with the largest most dominant share of the market with around 80% share compared to iOS’s 18 % share. Those numbers are a bit deceiving since android may be a fragmented market consisting of the many different devices created by different manufacturers, running completely different versions of the Android OS.

Types of Application

1. Native Applications:

These are applications developed to be used on a particular platform or operating system such as Android, iOS etc. Native apps are usually written in languages that the platform accepts. They are also built using the specific Integrated Development Environment (IDE) for the given operating systems, such as Android Studio for Android Apps and XCode for iOS Apps.

The principal advantage of native apps is that they optimize the user experience. By being designed and developed specifically for that platform, they look and perform better.

Examples of some popular Native Applications are Instagram for Android, VLC media player for Android, WordPress for iOS, and 2048 game for iOS.

2. Hybrid Applications:

These are applications developed to be used across multiple platforms i.e can be deployed on both iOS and Android platforms. Hybrid mobile applications are built in a similar manner as websites. Both use a combination of technologies like HTML, CSS, and JavaScript. However, instead of targeting a mobile browser, hybrid applications target a WebView hosted inside a native container. This enables them to do things like access hardware capabilities of the mobile device.

Today, most hybrid mobile applications leverage Apache Cordova, a platform that provides a consistent set of JavaScript APIs to access device capabilities through plug-ins, which are built with native code. Examples of some popular Hybrid Applications are MarketWatch, Untappd, FanReact, and TripCase.

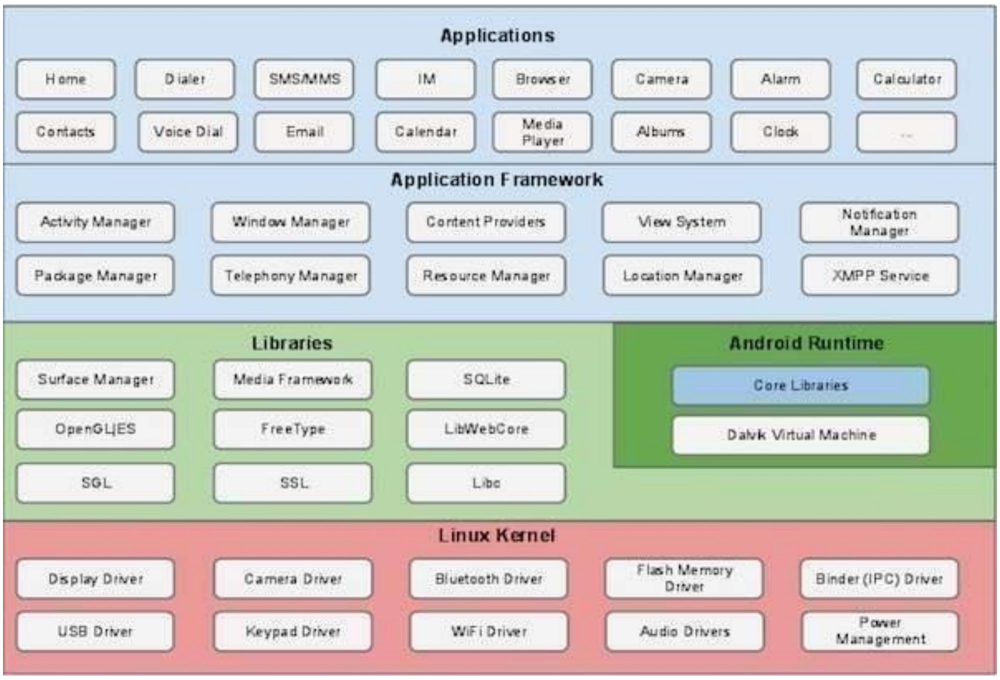
3. Progressive Web Applications (PWAs):

A Progressive Web App (PWA) is a web app that uses modern web capabilities to deliver an app-like experience to users without requiring them to install an app from the AppStore/PlayStore. They are usually accessible by a web URL which can always be pinned or saved on your phone’s home screen. PWAs are usually built using HTML, CSS, and JavaScript also.

Examples of some popular Progressive Web Applications are AliExpress’s PWA, Financial Times PWA, NASA’s PWA, and the recently just launched PayStack’s PWA.

Architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram. Fig 1.1 shows the Architecture of Android operating system.



Linux kernel

At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches. This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware. Libraries On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc. Android Libraries This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows −

- **android.app** − Provides access to the application model and is the cornerstone of all Android applications.
- **android.content** − Facilitates content access, publishing and messaging between applications and application components.
- **android.database** − Used to access data published by content providers and includes SQLite database management classes.
- **android.opengl** − A Java interface to the OpenGL ES 3D graphics rendering API.
- **android.os** − Provides applications with access to standard operating system services including messages, system services and inter-process communication.
- **android.text** − Used to render and manipulate text on a device display.
- **android.view** − The fundamental building blocks of application user interfaces.
- **android.widget** − A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
- **android.webkit** − A set of classes intended to allow web-browsing capabilities to be built into applications.

Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

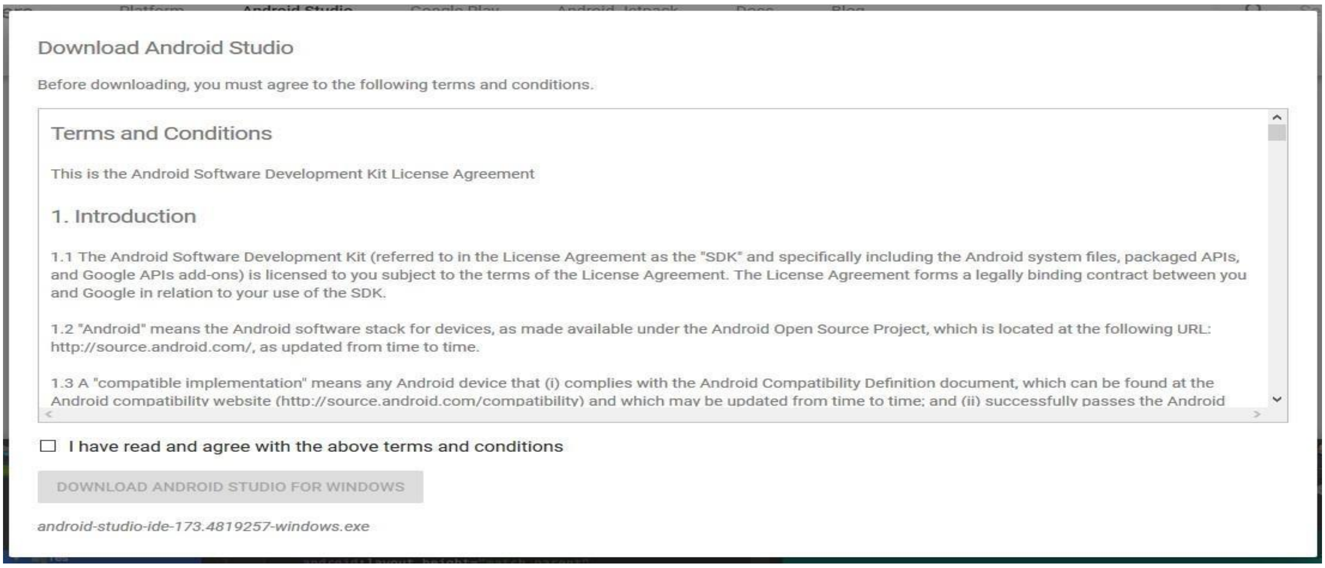
The Android framework includes the following key services −

- Activity Manager − Controls all aspects of the application life cycle and activity stack.
- Content Providers − Allows applications to publish and share data with other applications.
- Resource Manager − Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
- Notifications Manager − Allows applications to display alerts and notifications to the user.
- View System − An extensible set of views used to create application user interfaces.

Applications

You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games etc.

Install Android studio and Packages : https://developer.android.com/studio



Download Android Studio

Before downloading, you must agree to the following terms and conditions.

Terms and Conditions

This is the Android Software Development Kit License Agreement

1. Introduction

1.1 The Android Software Development Kit (referred to in the License Agreement as the "SDK" and specifically including the Android system files, packaged APIs, and Google APIs add-ons) is licensed to you subject to the terms of the License Agreement. The License Agreement forms a legally binding contract between you and Google in relation to your use of the SDK.

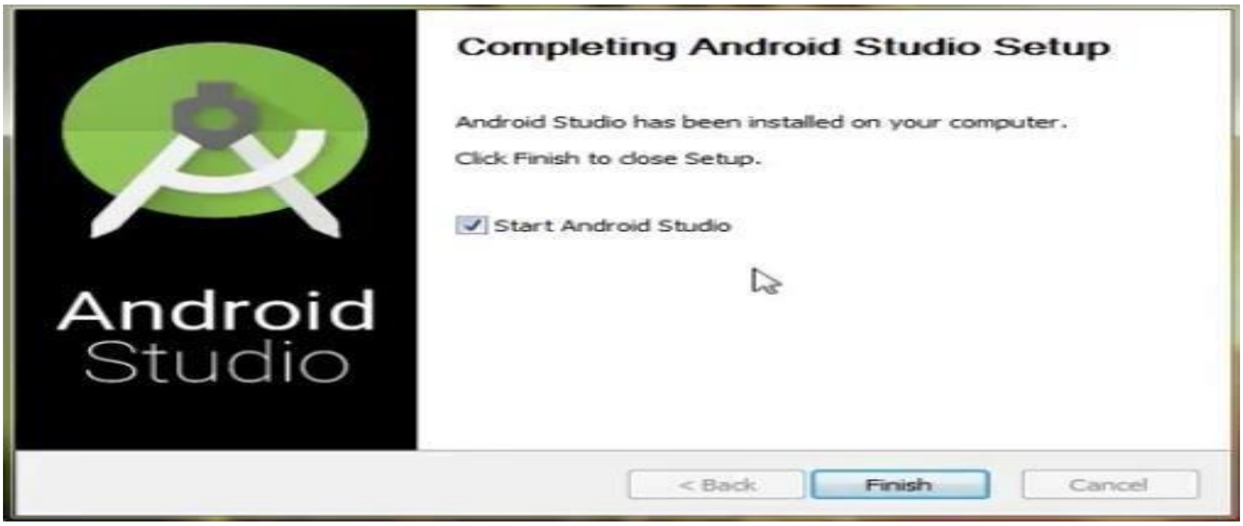
1.2 "Android" means the Android software stack for devices, as made available under the Android Open Source Project, which is located at the following URL: http://source.android.com/, as updated from time to time.

1.3 A "compatible implementation" means any Android device that (i) complies with the Android Compatibility Definition document, which can be found at the Android compatibility website (http://source.android.com/compatibility) and which may be updated from time to time; and (ii) successfully passes the Android

☐ I have read and agree with the above terms and conditions

android-studio-ide-173.4819257-windows.exe

Fig 1.2 Terms and conditions



Completing Android Studio Setup

Android Studio has been installed on your computer.

Click Finish to close Setup.

☑ Start Android Studio

Fig 1.3 Download Completion Screen

Fig 1.4 Starting Android Studio

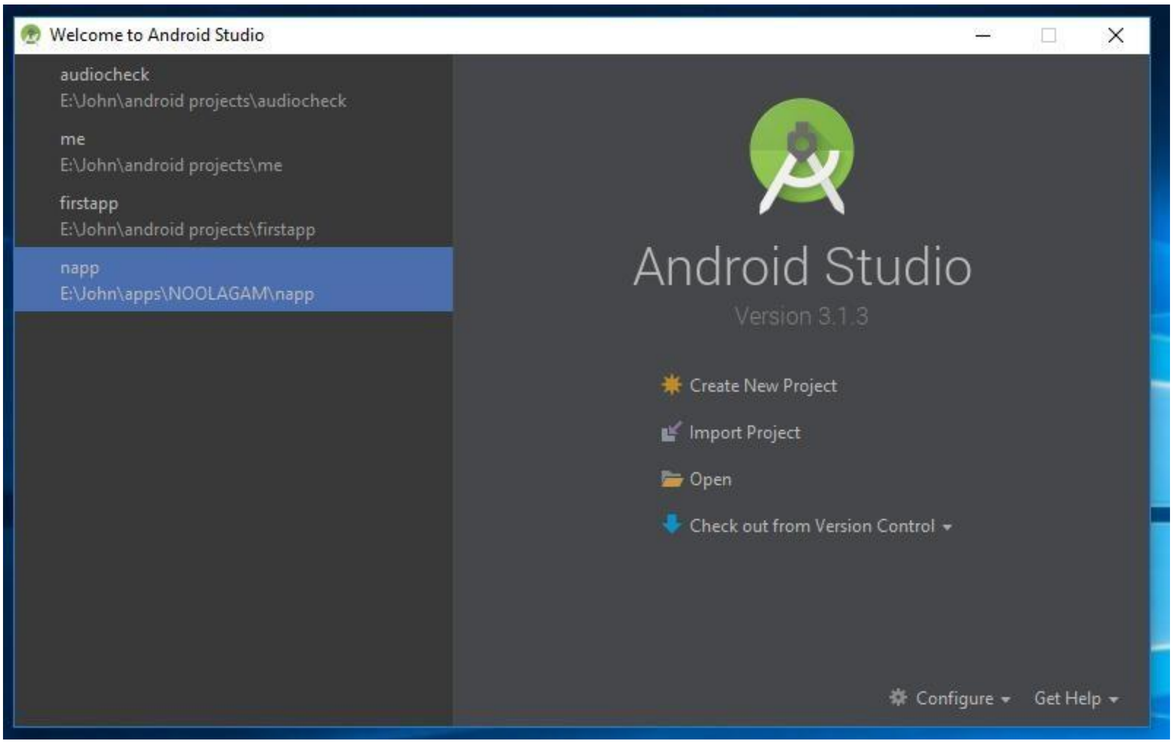
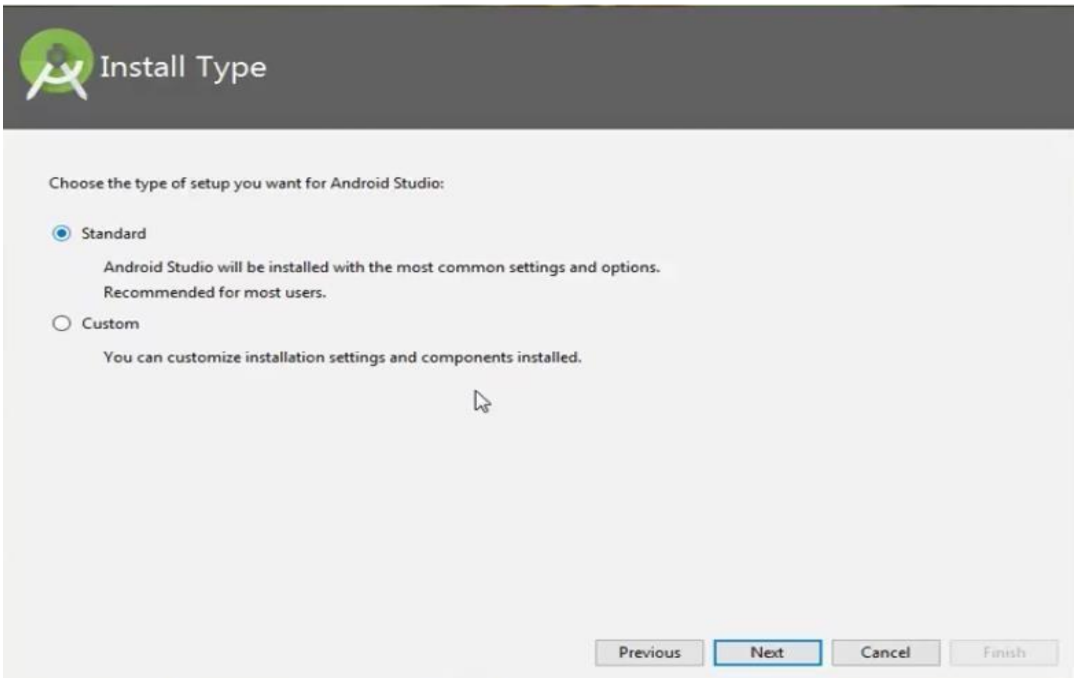

Welcome to Android Studio
audiocheck
E:\John\android projects\audiocheck
me
E:\John\android projects\me
firstapp
E:\John\android projects\firstapp
napp
E:\John\apps\NOOLAGAM\napp
Android Studio
Version 3.1.3
Create New Project
Import Project
Open
Check out from Version Control ▾
Configure ▾
Get Help ▾

Fig 1.5 Welcome Screen



Install Type

Choose the type of setup you want for Android Studio:

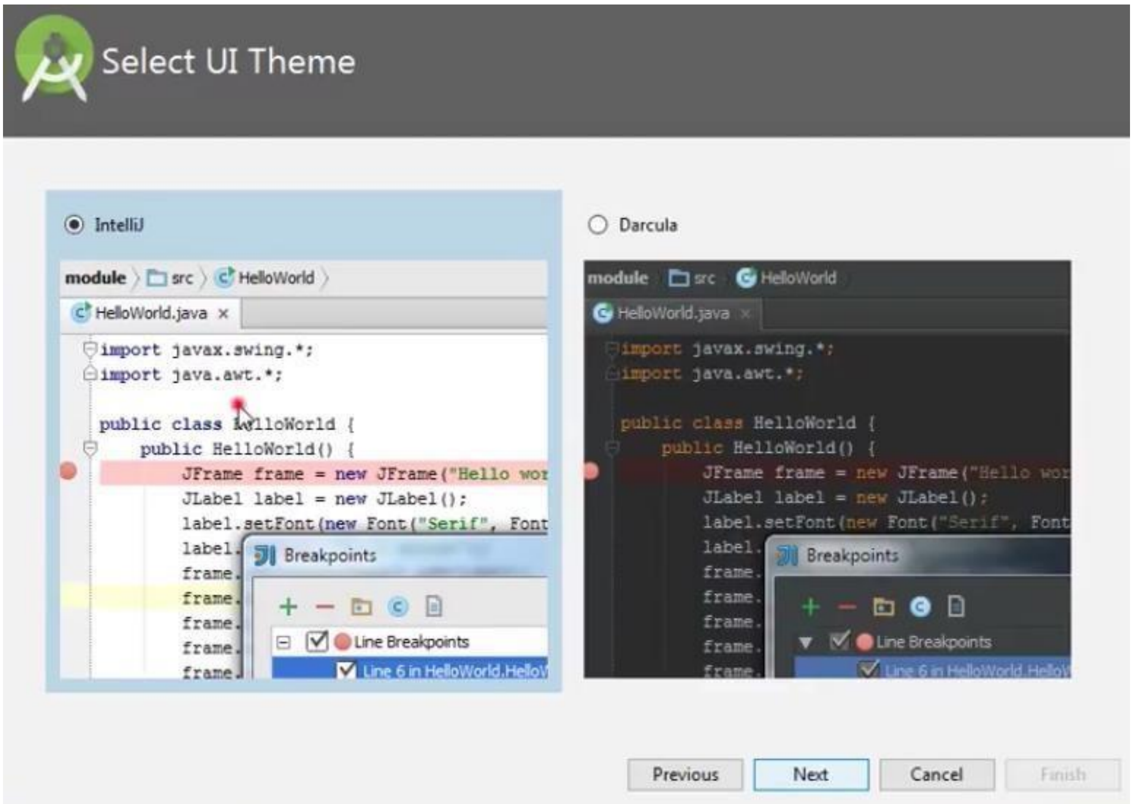
● Standard

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.

Fig 1.6 Selection of setup



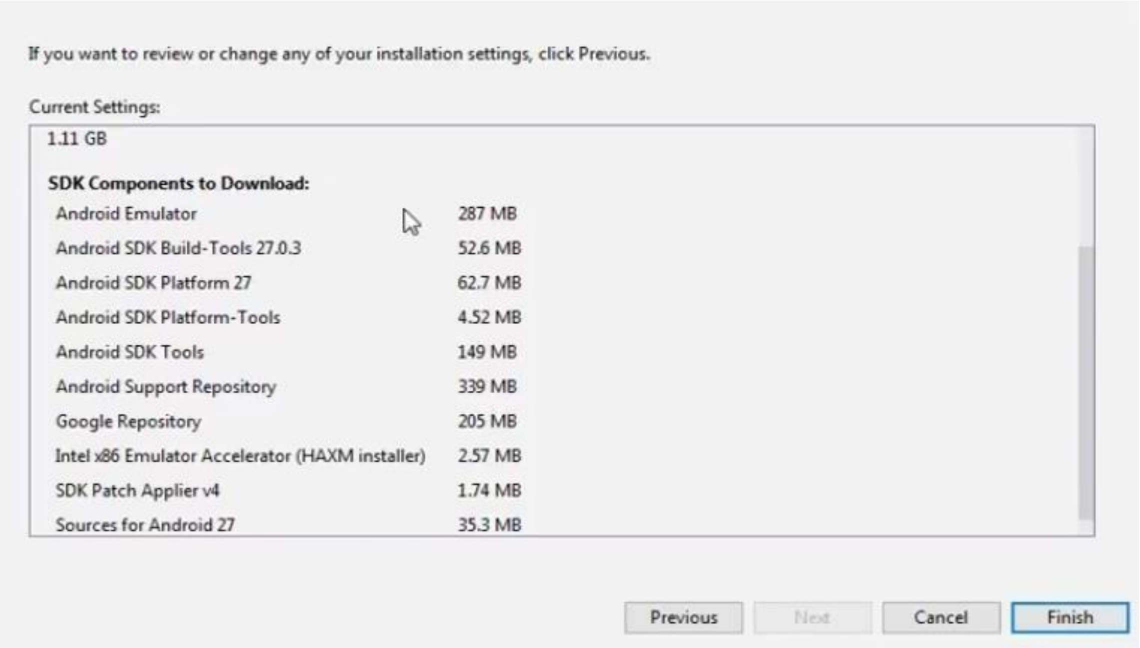
Select UI Theme

● IntelliJ

○ Darcula

```
import javax.swing.*;
import java.awt.*;

public class HelloWorld {
    public HelloWorld() {
        JFrame frame = new JFrame("Hello wor
        JLabel label = new JLabel();
        label.setFont(new Font("Serif", Font
```

Fig 1.7 Selection of Theme



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

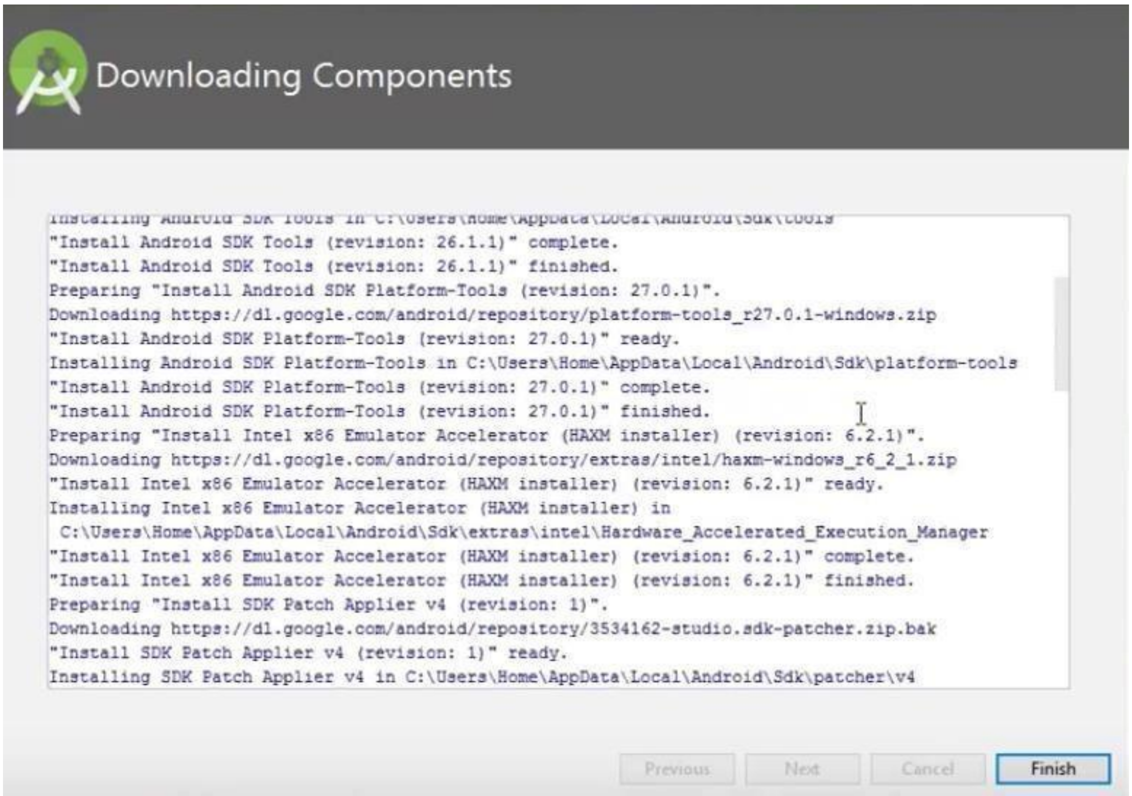
Current Settings:

1.11 GB

SDK Components to Download:

Android Emulator	287 MB
Android SDK Build-Tools 27.0.3	52.6 MB
Android SDK Platform 27	62.7 MB
Android SDK Platform-Tools	4.52 MB
Android SDK Tools	149 MB
Android Support Repository	339 MB
Google Repository	205 MB
Intel x86 Emulator Accelerator (HAXM installer)	2.57 MB
SDK Patch Applier v4	1.74 MB
Sources for Android 27	35.3 MB

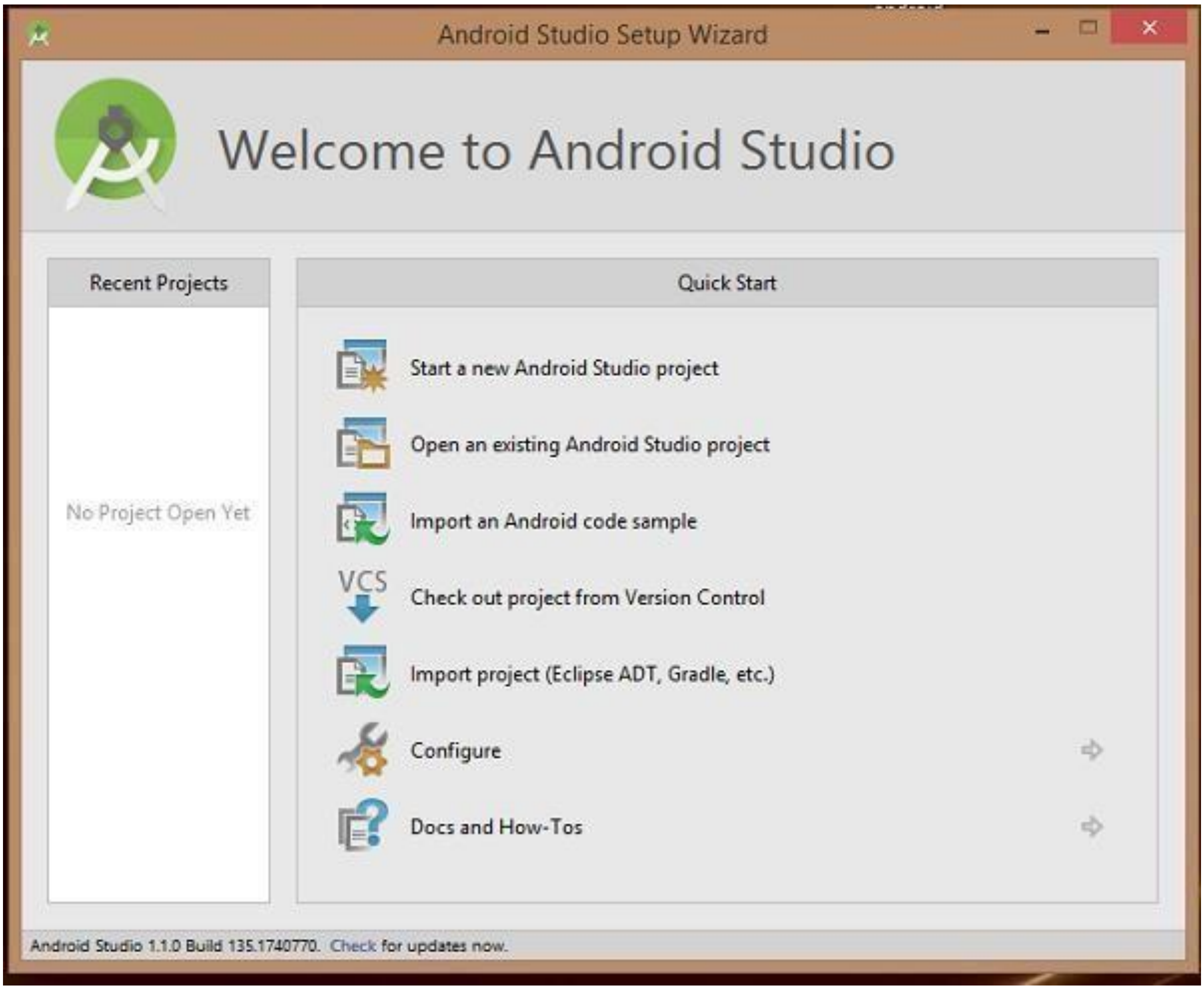
Fig 1.8 Downloading SDK components



```
Installing Android SDK Tools in C:\Users\Home\AppData\Local\Android\Sdk\tools
"Install Android SDK Tools (revision: 26.1.1)" complete.
"Install Android SDK Tools (revision: 26.1.1)" finished.
Preparing "Install Android SDK Platform-Tools (revision: 27.0.1)".
Downloading https://dl.google.com/android/repository/platform-tools_r27.0.1-windows.zip
"Install Android SDK Platform-Tools (revision: 27.0.1)" ready.
Installing Android SDK Platform-Tools in C:\Users\Home\AppData\Local\Android\Sdk\platform-tools
"Install Android SDK Platform-Tools (revision: 27.0.1)" complete.
"Install Android SDK Platform-Tools (revision: 27.0.1)" finished.
Preparing "Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)".
Downloading https://dl.google.com/android/repository/extras/intel/haxm-windows_r6_2_1.zip
"Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)" ready.
Installing Intel x86 Emulator Accelerator (HAXM installer) in
 C:\Users\Home\AppData\Local\Android\Sdk\extras\intel\Hardware_Accelerated_Execution_Manager
"Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)" complete.
"Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)" finished.
Preparing "Install SDK Patch Applier v4 (revision: 1)".
Downloading https://dl.google.com/android/repository/3534162-studio.sdk-patcher.zip.bak
"Install SDK Patch Applier v4 (revision: 1)" ready.
Installing SDK Patch Applier v4 in C:\Users\Home\AppData\Local\Android\Sdk\patcher\v4
```

Fig 1.9 Downloading SDK components

Creating a New Project in Android



Android Studio Setup Wizard

Welcome to Android Studio

Recent Projects

No Project Open Yet

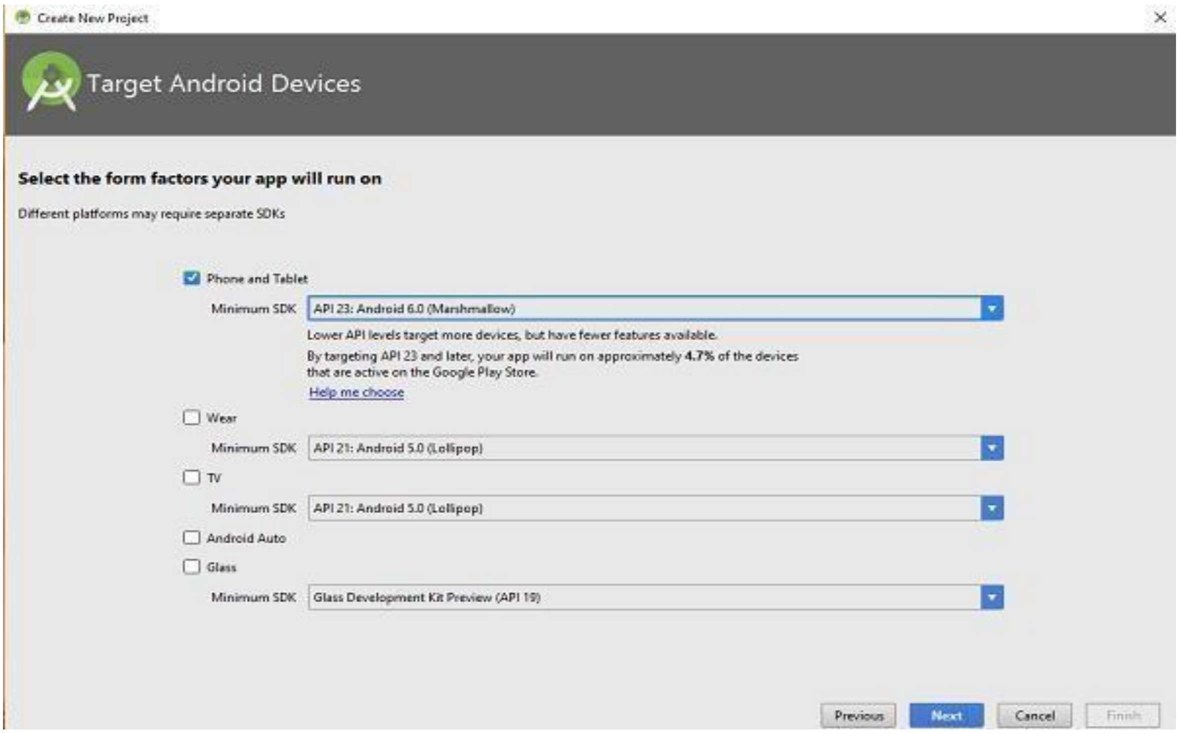
Quick Start

- Start a new Android Studio project
- Open an existing Android Studio project
- Import an Android code sample
- Check out project from Version Control
- Import project (Eclipse ADT, Gradle, etc.)
- Configure
- Docs and How-Tos

Android Studio 1.1.0 Build 135.1740770. Check for updates now.

Fig 1.10 Application name

After entered application name, it going to be called select the form factors your application runs on, here need to specify Minimum SDK, The next level of installation should contain selecting the activity to mobile, it specifies the default layout for Applications.



Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet

Minimum SDK: API 23: Android 6.0 (Marshmallow)

Lower API levels target more devices, but have fewer features available.

By targeting API 23 and later, your app will run on approximately **4.7%** of the devices that are active on the Google Play Store.

Help me choose

□ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

□ TV

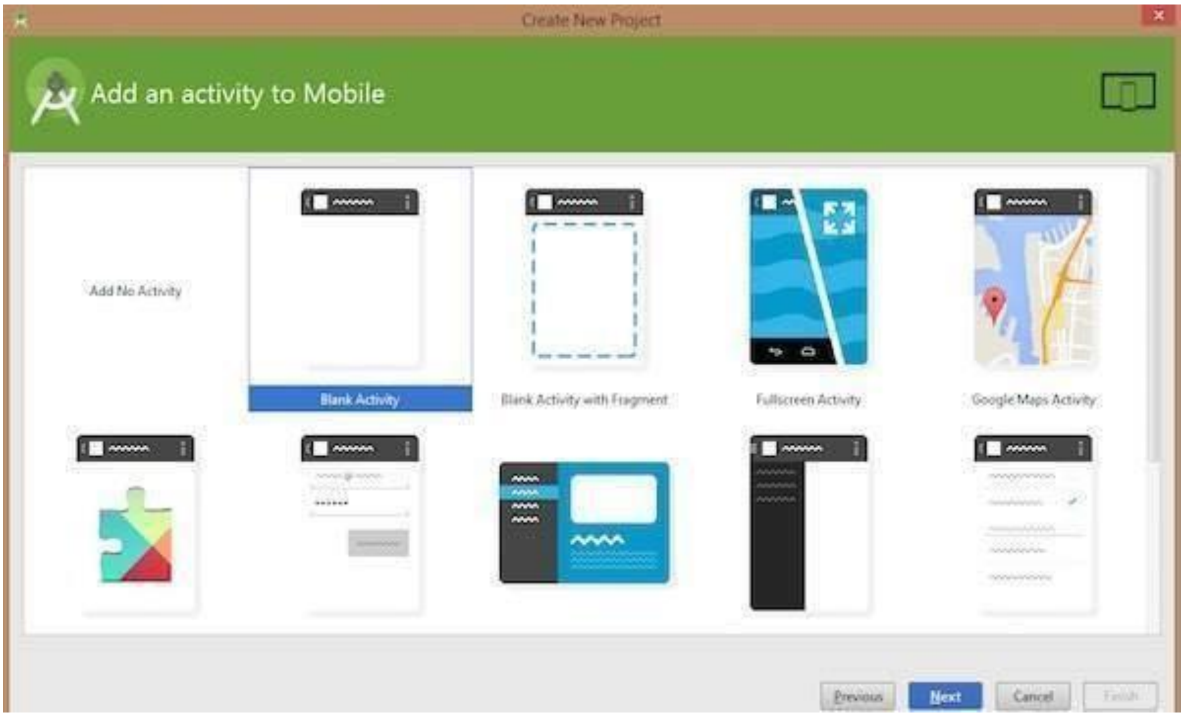
Minimum SDK: API 21: Android 5.0 (Lollipop)

□ Android Auto

□ Glass

Minimum SDK: Glass Development Kit Preview (API 19)

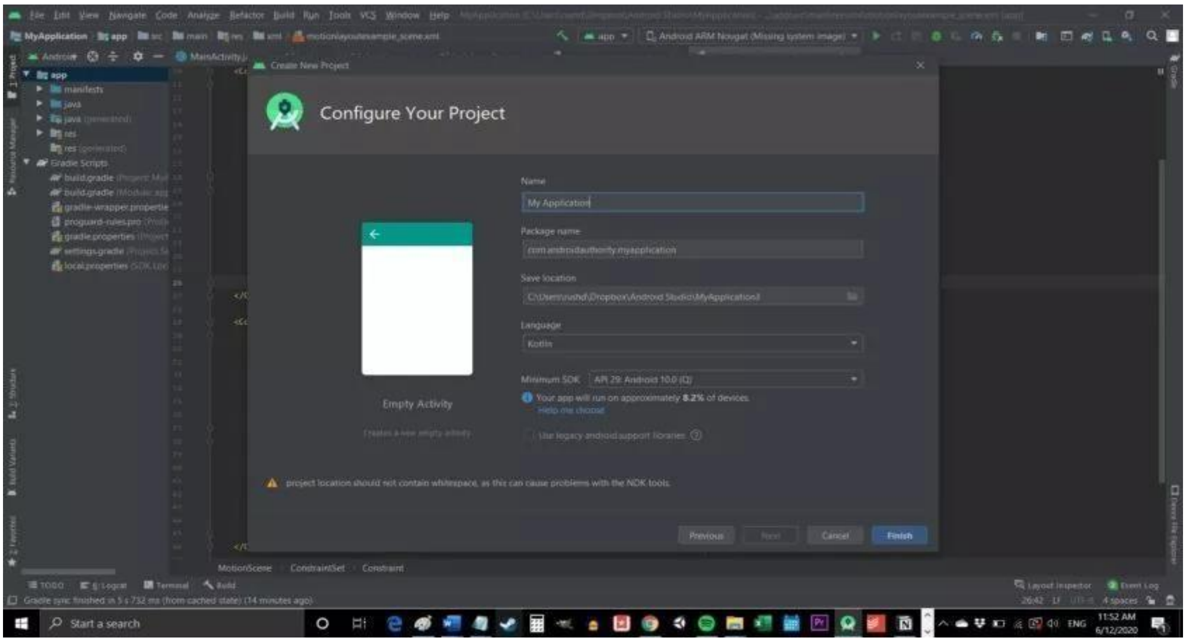
Fig 1.11 Selection of SDK



Create New Project

Add an activity to Mobile

Fig 1.12 Choosing the Activity



Create New Project

Configure Your Project

Name: My Application

Package name: com.androidauthority.myapplication

Save location: C:\Users\rushd\Dropbox\Android Studio\MyApplication4

Language: Kotlin

Minimum SDK: API 29: Android 10.0 (Q)

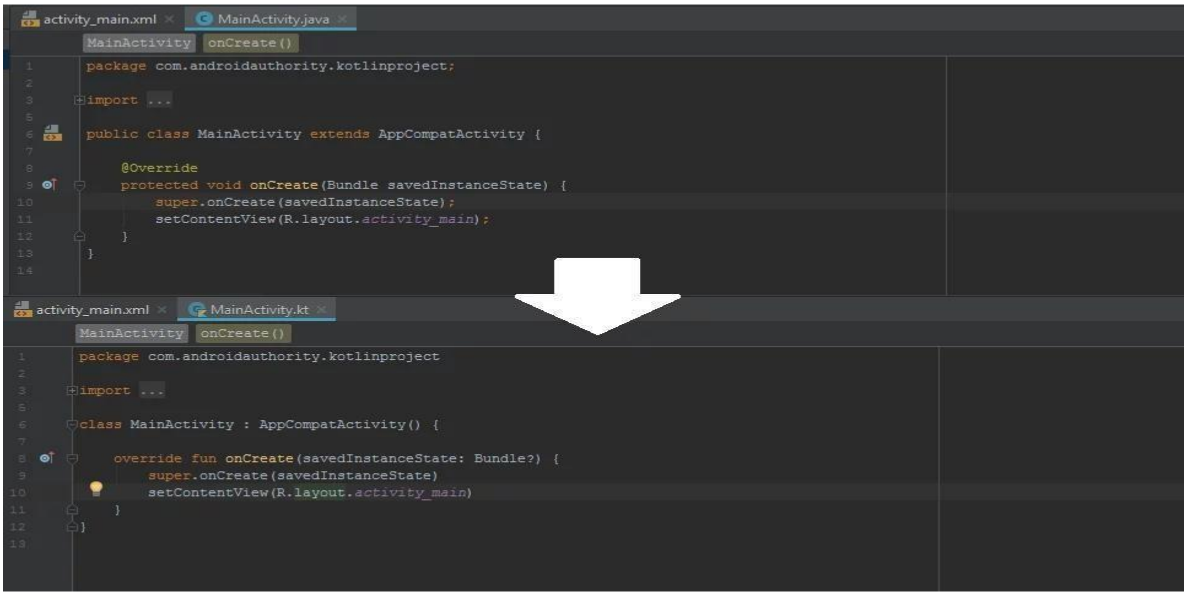
Your app will run on approximately **8.2%** of devices.
Help me choose

□ Use legacy android.support libraries

project location should not contain whitespace, as this can cause problems with the NDK tools.

Fig 1.13 Naming an application

As soon as Finish is clicked the application will open with the Main Application java and activity_xml files



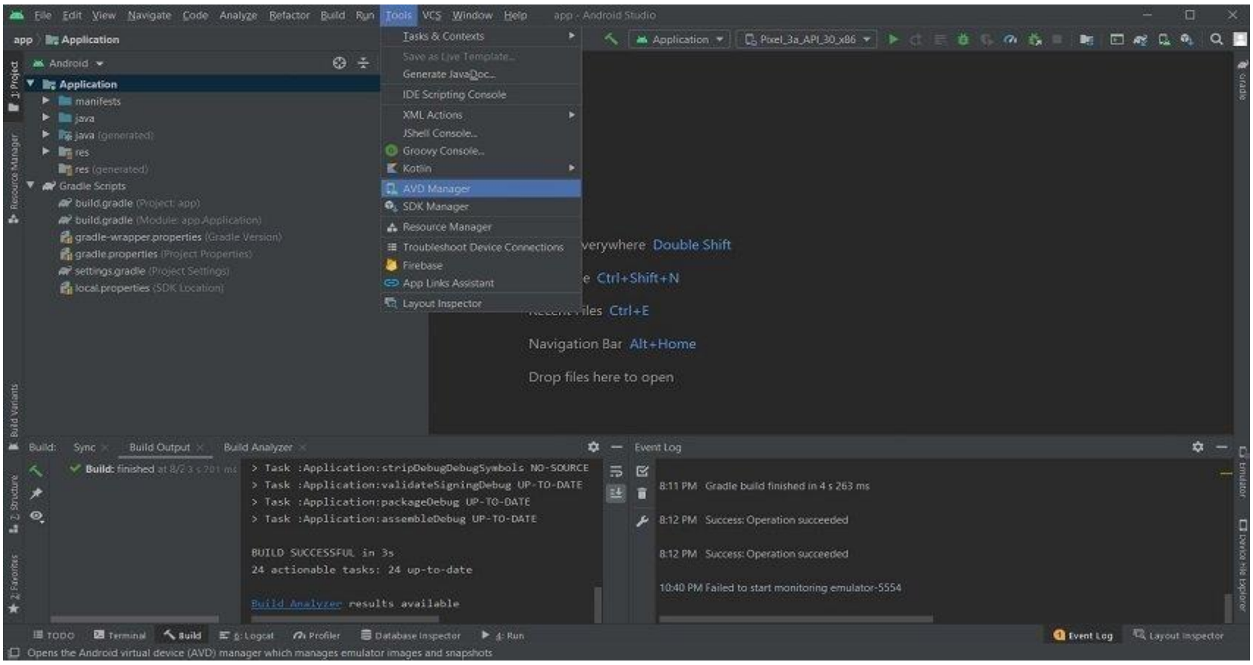
```
package com.androidauthority.kotlinproject;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

```
package com.androidauthority.kotlinproject

import ...

class MainActivity : AppCompatActivity() {

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

Fig 1.14 Creation of java and xml file

Creating emulator

1. In Android Studio, create an Android Virtual Device (AVD) that the emulator can use to install and run your app.
2. In the toolbar, select the AVD that you want to run your app on from the target device drop-down menu.
3. Click Run.


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help app - Android Studio
app Application
Application
Pixel_3a_API_30_x86
Android
Application
manifests
java
java (generated)
res
res (generated)
Gradle Scripts
build.gradle (Project: app)
build.gradle (Module: app.Application)
gradle-wrapper.properties (Gradle Version)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
Tasks & Contexts
Save as Live Template...
Generate JavaDoc...
IDE Scripting Console
XML Actions
JShell Console...
Groovy Console...
Kotlin
AVD Manager
SDK Manager
Resource Manager
Troubleshoot Device Connections
Firebase
App Links Assistant
Layout Inspector
verywhere Double Shift
e Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open
Build: Sync Build Output Build Analyzer
Event Log
TODO Terminal Build 6: Logcat Profiler Database Inspector 4: Run
Event Log Layout Inspector

```
Build: finished at 8/2 3 s 701 ms
> Task :Application:stripDebugDebugSymbols NO-SOURCE
> Task :Application:validateSigningDebug UP-TO-DATE
> Task :Application:packageDebug UP-TO-DATE
> Task :Application:assembleDebug UP-TO-DATE

BUILD SUCCESSFUL in 3s
24 actionable tasks: 24 up-to-date

Build Analyzer results available
```

```
8:11 PM  Gradle build finished in 4 s 263 ms

8:12 PM  Success: Operation succeeded

8:12 PM  Success: Operation succeeded

10:40 PM Failed to start monitoring emulator-5554
```

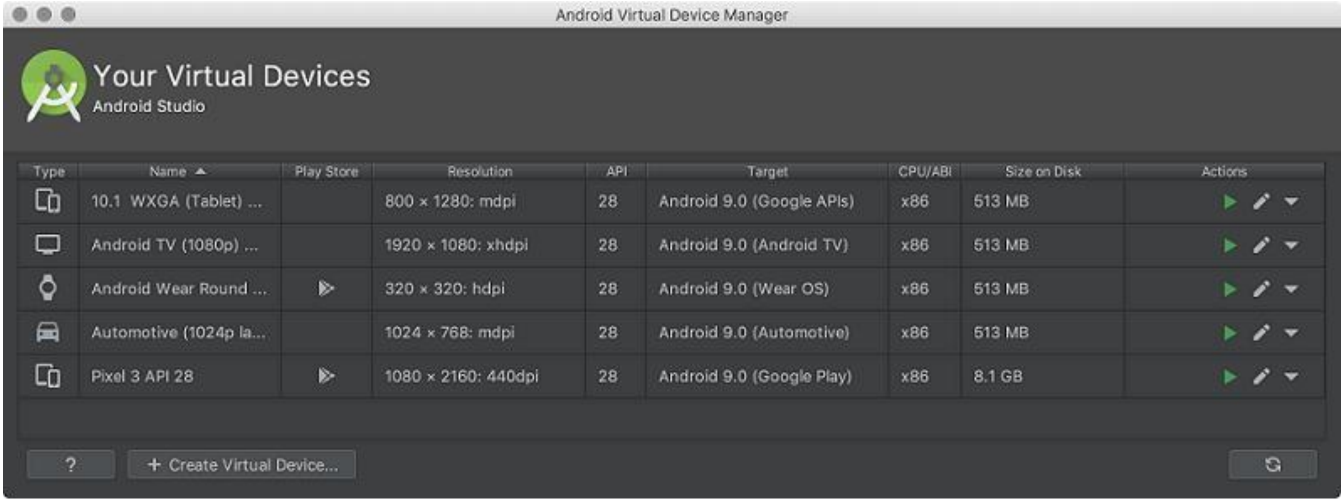
Opens the Android virtual device (AVD) manager which manages emulator images and snapshots

Fig 1.15 Selecting AVD Manager

Downloading a Virtual Device

1. In Android Studio, we need to create an Android Virtual Device (AVD) that the emulator can use to install and run your app. To create a new AVD:-

1.1 Open the AVD Manager by clicking **Tools > AVD Manager.**

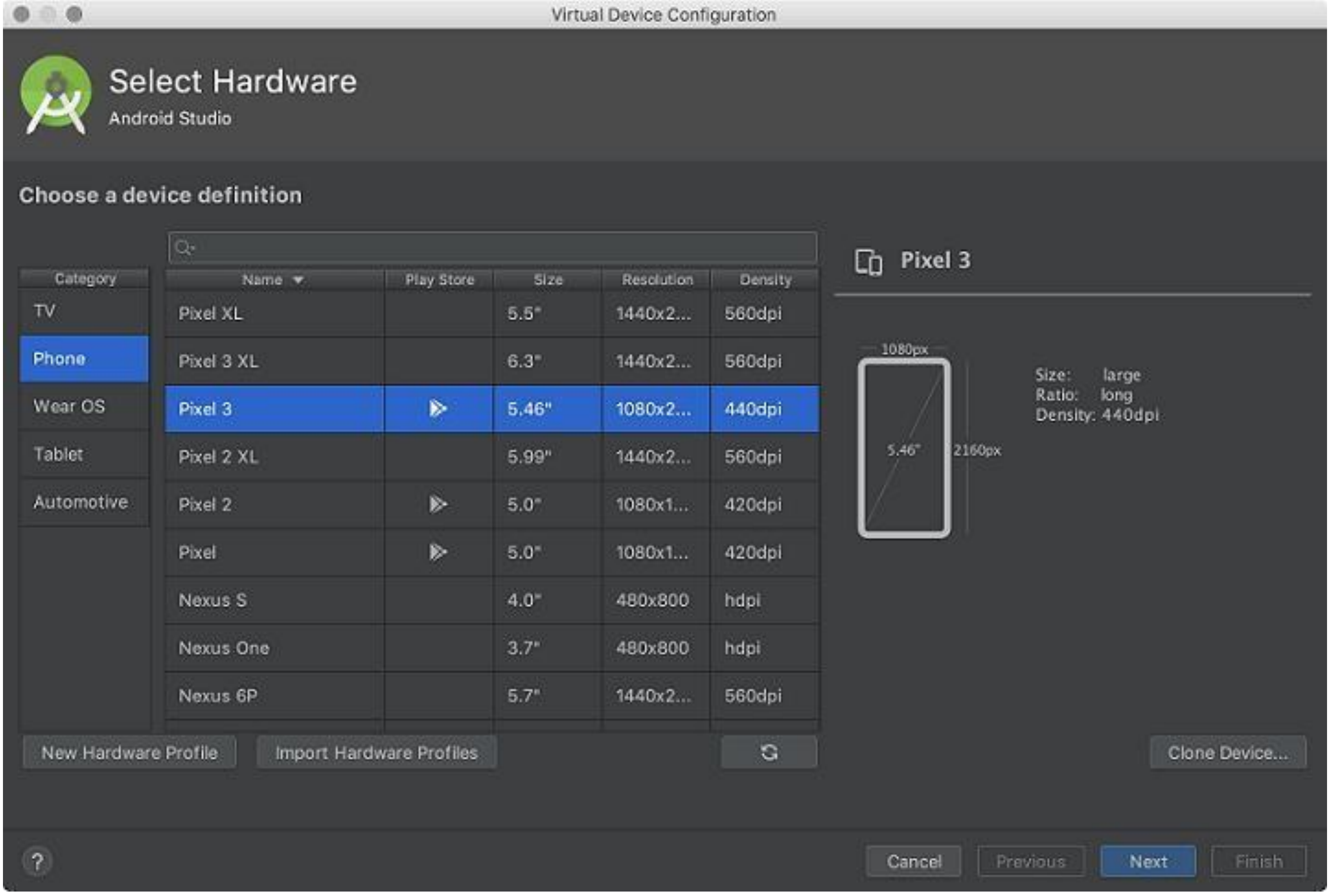


Type	Name ▲	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	10.1 WXGA (Tablet) ...		800 × 1280: mdpi	28	Android 9.0 (Google APIs)	x86	513 MB	
	Android TV (1080p) ...		1920 × 1080: xhdpi	28	Android 9.0 (Android TV)	x86	513 MB	
	Android Wear Round ...	▶	320 × 320: hdpi	28	Android 9.0 (Wear OS)	x86	513 MB	
	Automotive (1024p la...		1024 × 768: mdpi	28	Android 9.0 (Automotive)	x86	513 MB	
	Pixel 3 API 28	▶	1080 × 2160: 440dpi	28	Android 9.0 (Google Play)	x86	8.1 GB	

Fig 1.16 Downloading virtual device

Each virtual device comprises some hardware and software configurations. The system image of a virtual device represents its software components. When you set out to create a new virtual device, you’ll have to determine its hardware first. This is where you select settings like the screen size, screen resolution, screen pixel density, and RAM. You can define the hardware from scratch or use the default hardware options offered by Android Studio.In the hardware selection menu, you’ll also get to see a Play Store icon beside some of the hardware choices. The system image of these devices will have a Play Store integrated into their interfaces.

Once you’re done selecting the hardware configurations, click next and you’ll be taken to the system image menu.A system image comprises the Android version, its API level, and ABI. These selections have to be in accordance with your project.Click on Next and the download will start Click on Create **Virtual** Device, at the bottom of the AVD Manager dialog. Then **Select Hardware** page appears.



Virtual Device Configuration

Select Hardware
Android Studio

Choose a device definition

Category
TV
Phone
Wear OS
Tablet
Automotive

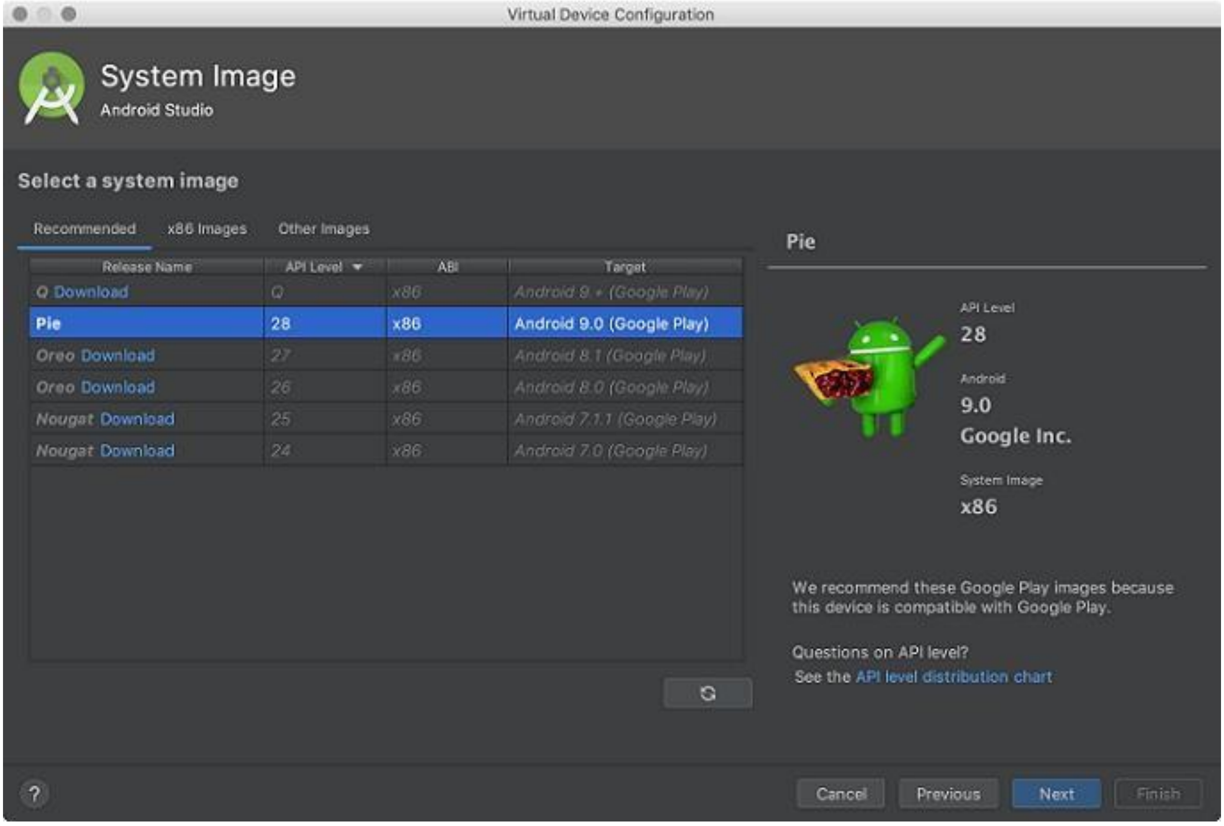
Name ▾	Play Store	Size	Resolution	Density
Pixel XL		5.5"	1440x2...	560dpi
Pixel 3 XL		6.3"	1440x2...	560dpi
Pixel 3	▶	5.46"	1080x2...	440dpi
Pixel 2 XL		5.99"	1440x2...	560dpi
Pixel 2	▶	5.0"	1080x1...	420dpi
Pixel	▶	5.0"	1080x1...	420dpi
Nexus S		4.0"	480x800	hdpi
Nexus One		3.7"	480x800	hdpi
Nexus 6P		5.7"	1440x2...	560dpi

Pixel 3

Size: large
Ratio: long
Density: 440dpi

Fig 1.17 Selecting Type of device

Select a hardware profile and then click **Next.** If we don 't see the hardware profile we want, then we can create or import a hardware profile. The **System Image** page appears.



System Image
Android Studio

Select a system image

Release Name	API Level	ABI	Target
Q Download	Q	x86	Android 9.+ (Google Play)
Pie	28	x86	Android 9.0 (Google Play)
Oreo Download	27	x86	Android 8.1 (Google Play)
Oreo Download	26	x86	Android 8.0 (Google Play)
Nougat Download	25	x86	Android 7.1.1 (Google Play)
Nougat Download	24	x86	Android 7.0 (Google Play)

Pie

API Level
28

Android
9.0
Google Inc.

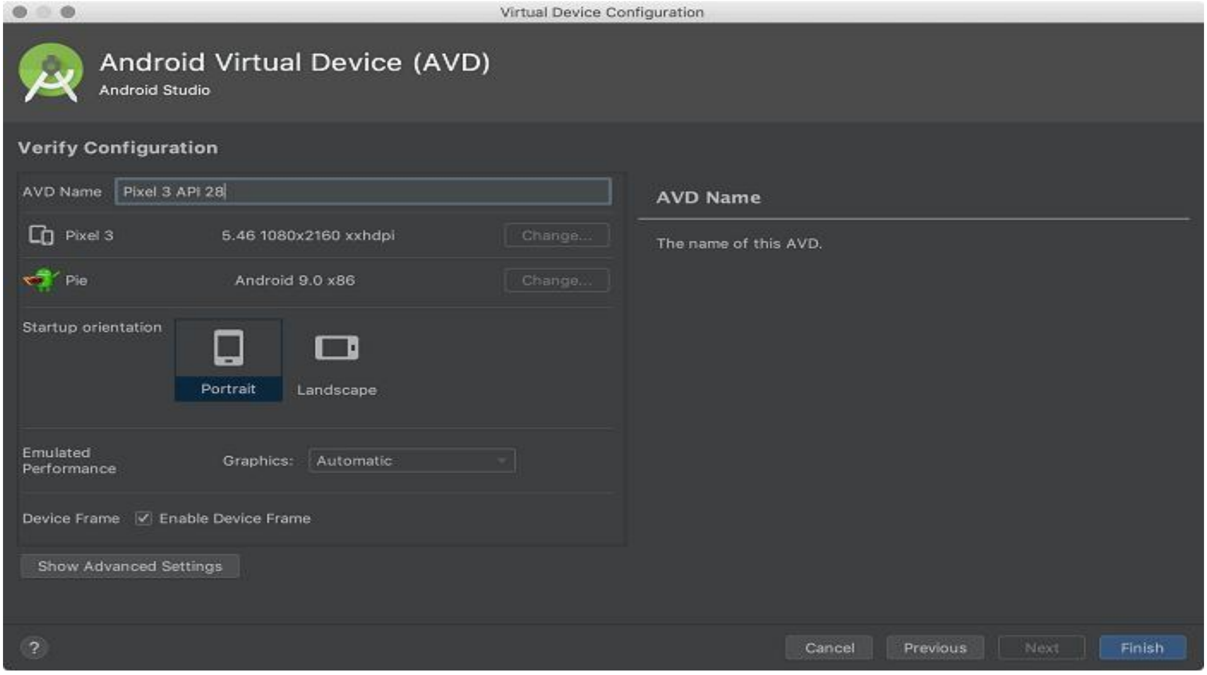
System Image
x86

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

Questions on API level?
See the API level distribution chart

Fig 1.18 System image of Virtual Device

Select the system image for the particular API level and click Next. This leads to open a Verify Configuration page.



Android Virtual Device (AVD)
Android Studio

Verify Configuration

AVD Name: Pixel 3 API 28

Pixel 3 5.46 1080x2160 xxhdpi Change...

Pie Android 9.0 x86 Change...

Startup orientation: Portrait Landscape

Emulated Performance Graphics: Automatic

Device Frame ☑ Enable Device Frame

AVD Name

The name of this AVD.

Fig 1.19 Type of Orientation

Change AVD properties if needed, and then click Finish. In the toolbar, choose the AVD, which we want to run our app from the target device from the drop-down menu.

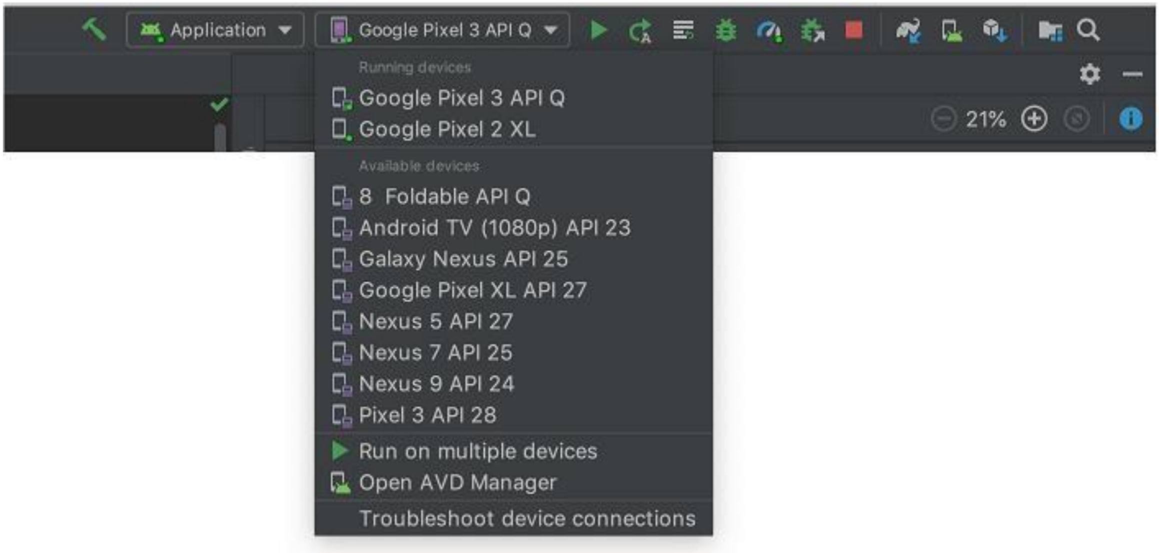


Fig 1.20 List of Devices

Click on Run by selecting the device.

To start the emulator:
1. Open the AVD Manager.
2. Double-click an AVD, or click **Run**
While the emulator is running, we can run the Android Studio project and select the emulator as the target device. We can also drag an APKs file to install on an emulator, and then run them.

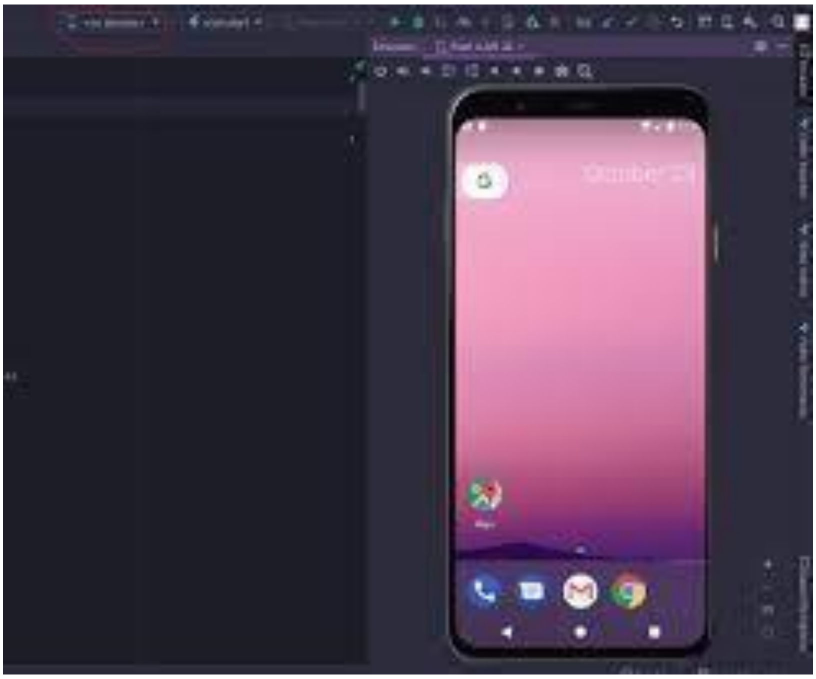


Fig 1.21 : Screenshot of System Emulator

Steps for Creating the First Android Application

Android applications are written in Java, and use XML extensively. We shall assume that you have basic knowledge of Java and XML. Go to "Android Guides" @ https://developer.android.com/guide/index.html. Read "Building your first app".

Creating a New Android Project

Step1: Launch Android Studio.

Step 2: Select File -> New-> New Project

Step 3: In "Choose your project", select "Phone and Tablet" tab -> "Empty Activity" ->Next.

Step 4: In "Configure your project" ->Set "Name" to "Hello Android" (this will be the "Title" in your phone's application menu) ->The "Package name" and "Save Location" will be updated automatically -> In "Language", select "Java" -> Leave the "Minimum API Level" and the rest to default -> Finish.

It could take a few minutes to set up your first app. Watch the "progress bar" at the bottom status bar. Once the progress bar indicates completion, a hello-world app is created by default.

Set up Emulator (Android Virtual Device (AVD))

To run your Android application under the emulator, you need to first create an Android Virtual Devices (AVD). An AVD models a specific device. You can create AVDs to emulate different android devices (e.g., phone/tablet, android version, screen size, and etc.).

Step 1: In Android studio, select "Tools" -> AVD Manager.

Step 2: Click "Create Virtual Device".

Step 3: In "Choose a device definition" -> In "Category", choose "Phone" -> In "Name", choose "2.7 QVGA" (the smallest device available - you can try a bigger device later) -> Next.

Step 4: In "System Image: Recommended" -> Select the version with the highest API level ->Click "Download" -> Next.

Step 5: In "AVD Name", enter "2.7 QVGA API 27" (default)   Finish.

Running the Android Application on Emulator

Step 1: Select the "Run" menu -> "Run app" -> Under "Available Virtual Devices", select "2.7 QVGA API 27" -> OK.

Step 2: It may take a few MINUTES to fire up the app on the emulator. You first see a Google logo -> then "Android" -> then the "wallpaper" ->then the "Hello, world!" message.

Step 3: DO NOT CLOSE THE EMULATOR, as it really takes a long time to start. You could always re-run the app (or run a new app) on the same emulator. Try re-run the app by selecting "Run" menu -> "Run app".

Program 1

Create an application to design a Visiting Card. The Visiting card should have a company logo at the top right corner. The company name should be displayed in Capital letters, aligned to the center. Information like the name of the employee, job title, phone number, address, email, fax and the website address are to be displayed.

1. Create a New Android Project with Empty Activity.
2. Open activity_main.xml file from res layout folder.
3. Set all constraints to Constraint Layout and Text View.
4. Use View background property to draw the line
5. Add Image to drawable folder and reference the image in the layout using @drawable/<image_name>
6. Use android:layout_gravity/android:gravity properties to center the components.

Following is the content of the modified res/layout/activity_main.xml.

XML Code: activity_main.xml

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


    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentEnd="true"
        android:layout_marginEnd="319dp"

        android:text="NCET"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
        android:textSize="22sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.055" />
```

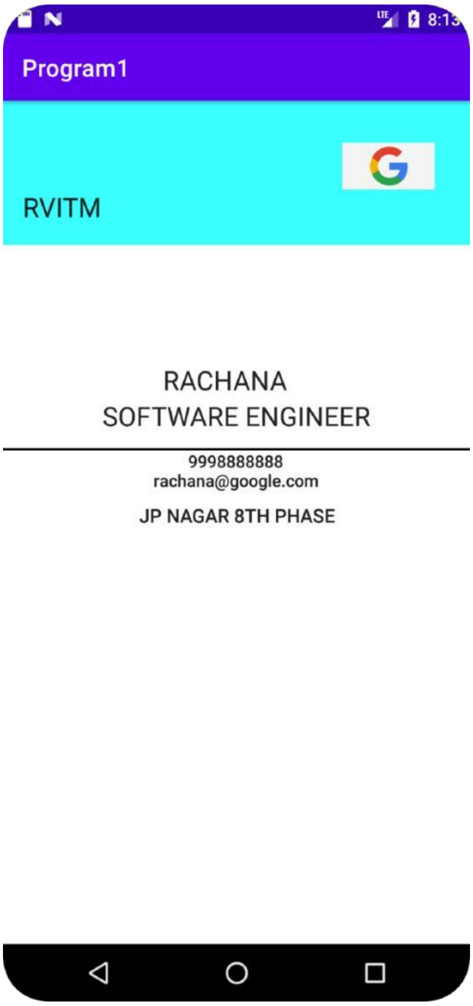
```
<ImageView

    android:id="@+id/imageView3"
    android:layout_width="170dp"
    android:layout_height="39dp"
    android:layout_alignParentTop="true"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.044"
    app:srcCompat="@drawable/ncet" />

<TextView
    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="RACHANA"
    android:textAppearance="@style/TextAppearance.AppCompat.Body1"
    android:textSize="22sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.555"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.221" />

<TextView
    android:id="@+id/textView5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:fontFamily="sans-serif"
    android:text="Assistant Professor"
    android:textAppearance="@style/TextAppearance.AppCompat.Body2"
    android:textSize="22sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.583"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.316" />

<TextView
    android:id="@+id/textView6"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="rachana@google.com"
```

```
        android:textAppearance="@style/TextAppearance.AppCompat.Body2"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.575"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.456" />

    <TextView

        android:id="@+id/textView7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="9998888888"
        android:textAppearance="@style/TextAppearance.AppCompat.Body2"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.543"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.383" />

    <TextView

        android:id="@+id/textView8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

        android:includeFontPadding="true"
        android:text="Devanahalli, Bangalore"
        android:textAppearance="@style/TextAppearance.AppCompat.Body2"
        android:textSize="16sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.591"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.52" />


</androidx.constraintlayout.widget.ConstraintLayout>
```

JAVA Code: MainActivity.java

```
package com.example.program1;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState)
        { super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

OUTPUT



RVITM

RACHANA
SOFTWARE ENGINEER

9998888888
rachana@google.com

JP NAGAR 8TH PHASE

Program 2:

Develop an application using controls like Button and TextView, on click of Button display "Welcome to NCET" and on Click of Button 2 display "Welcome to CSE". Handle the two buttons in an Activity to demonstrate event handling using both Java and XML.

1. Create a New Android Project with EmptyActivity.
2. Open activity_main.xml file from res layout folder, check/add Constraint Layout as the root view.
3. Create Layout using Drag and Drop frame work.
4. OpenMainActivty.javafile,Overrideon Create() method and bring activity_main.xmlfile on screen using setContentView() and bring the view references using findViewById() method.
5. Add Listeners to Button ClickEvent:
6. Create a class which implements OnClickListenerinterface.
7. Override onClick() method of OnClickListenerInterface.
8. Register the button for click event by calling setOnClickListener() method of View class and pass the object of the class that implemented OnClickListenerInterface.
9. Create onClick attribute in XML and implement the method in Java.

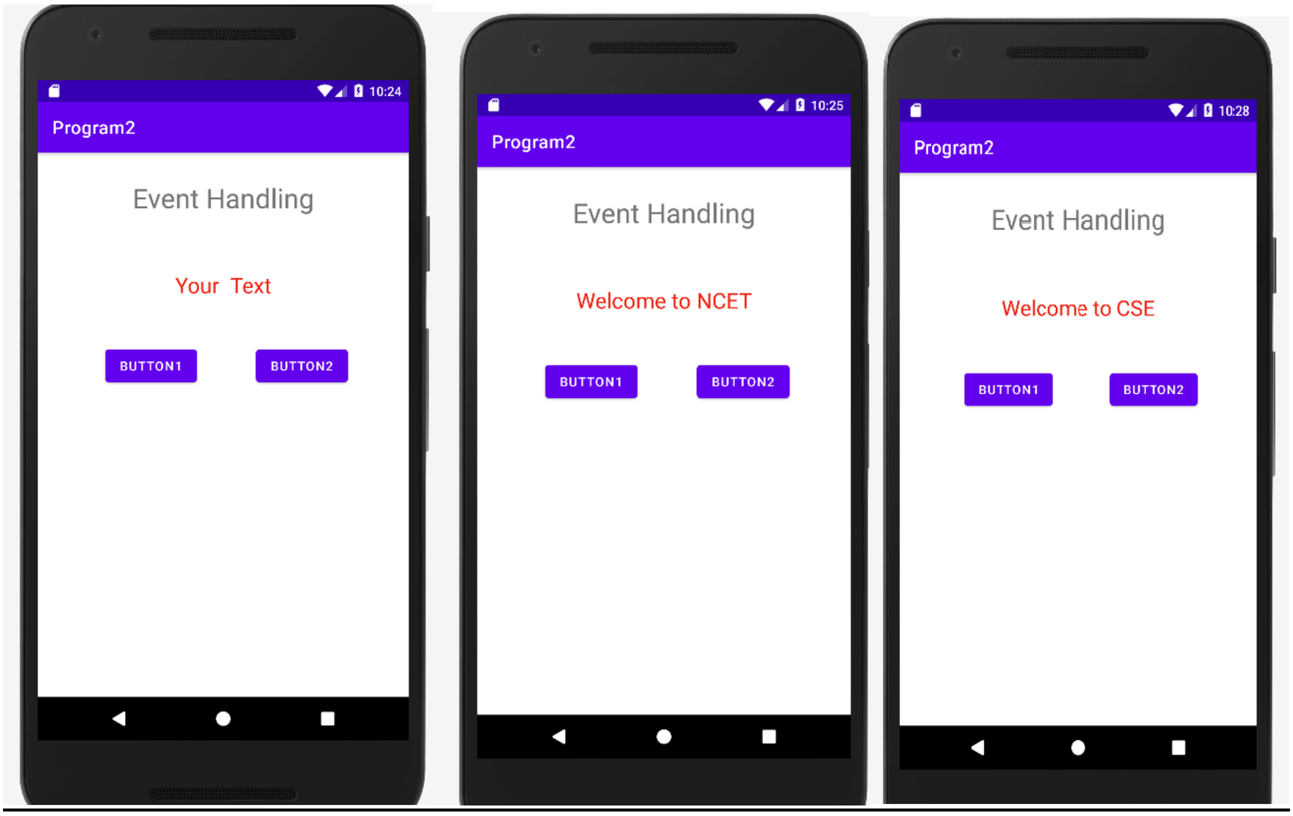
XML Code: Activity_Main.xml

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

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Event Handling"
        android:textColor="#FF5722"
        android:textSize="30sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.082" />
    <TextView
        android:id="@+id/t1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Your Text"
        android:textSize="30sp"
        android:textColor="#3F51B5"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.526"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.221" />

    <Button
        android:id="@+id/b1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button1"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.138"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.421" />

    <Button
        android:id="@+id/b2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button2"
        android:onClick="dosomething"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.87"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.421" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Java Code: MainActivity.java

```
package com.example.aProgram2;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity implements View.OnClickListener
{
   TextView tv;
   Button bt;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      tv=(TextView) findViewById(R.id.t1);
      bt=(Button) findViewById(R.id.b1);
      bt.setOnClickListener(this);

   }

   @Override
   public void onClick(View view) {
      tv.setText("Welcome to NCET");
   }

   public void dosomething(View view) {
      tv.setText("Welcome to CSE");
   }
}
```

OUTPUT



Event Handling

Your Text

Event Handling

Welcome to NCET

Event Handling

Welcome to CSE

Program 3:

Create an application to demonstrate all Activity life cycle callback methods. Display Toast message When each method invokes.

1. Create a New Android Project with Empty Activity.
2. Override all Activity life cycle methods.
   onCreate(), onStart(), onResume(), onPause(), onStop, onRestart(), onDestroy()
3. Invoke Toast. makeText method to display Toast.
4. Use android:layout_gravity/android:gravity properties to center the components.

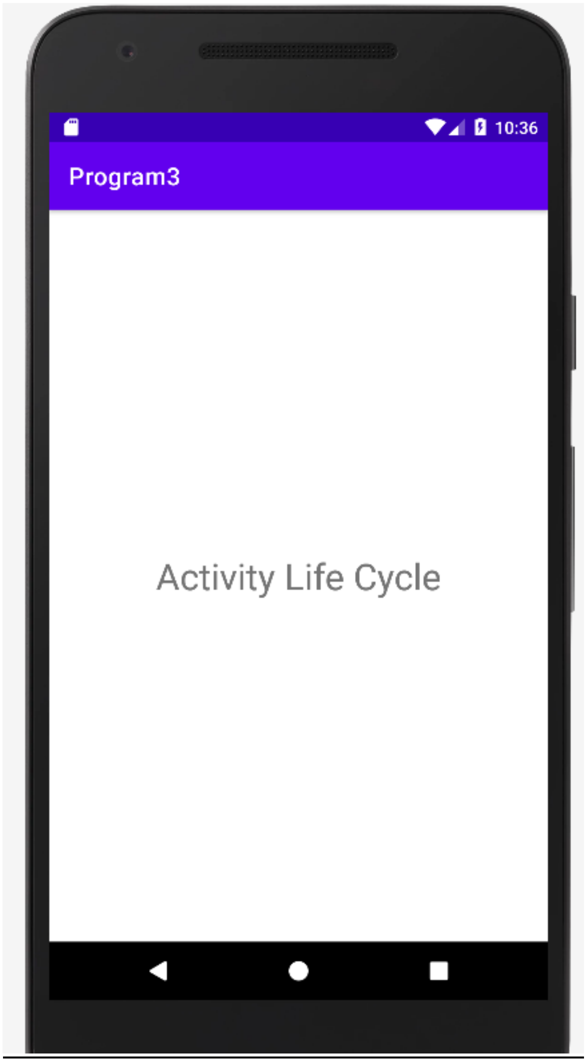
XML Code: activity_main.xml

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Java code:MainActivity.java

```
package com.example.aProgram3;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    protected void onStart() {
        super.onStart();
        Toast.makeText(this, "onstart", Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onResume() {
        super.onResume();
        Toast.makeText(this, "onresume", Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onPause() {
        super.onPause();
        Toast.makeText(this, "onpause", Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onStop() {
        super.onStop();
        Toast.makeText(this, "onstop", Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onRestart() {
        super.onRestart();
        Toast.makeText(this, "onrestart", Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        Toast.makeText(this, "ondestroy", Toast.LENGTH_LONG).show();
    }
}
```

OUTPUT



10:36

Program3

Activity Life Cycle

Program 4:

Develop an android application with an Activity and accept data from the user in first Activity. On click of a Next button in first Activity transfer the data from first Activity to second Activity.

XML code: activity_main.xml

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

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Explicit Intent"
        android:textSize="30sp"

        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.139" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="dosomething"
        android:text="Next"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.499" />

    <EditText
        android:id="@+id/edit"
        android:layout_width="380dp"
        android:layout_height="45dp"
        android:ems="10"
        android:hint="Enter your name"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.291" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Java code: MainActivity.java

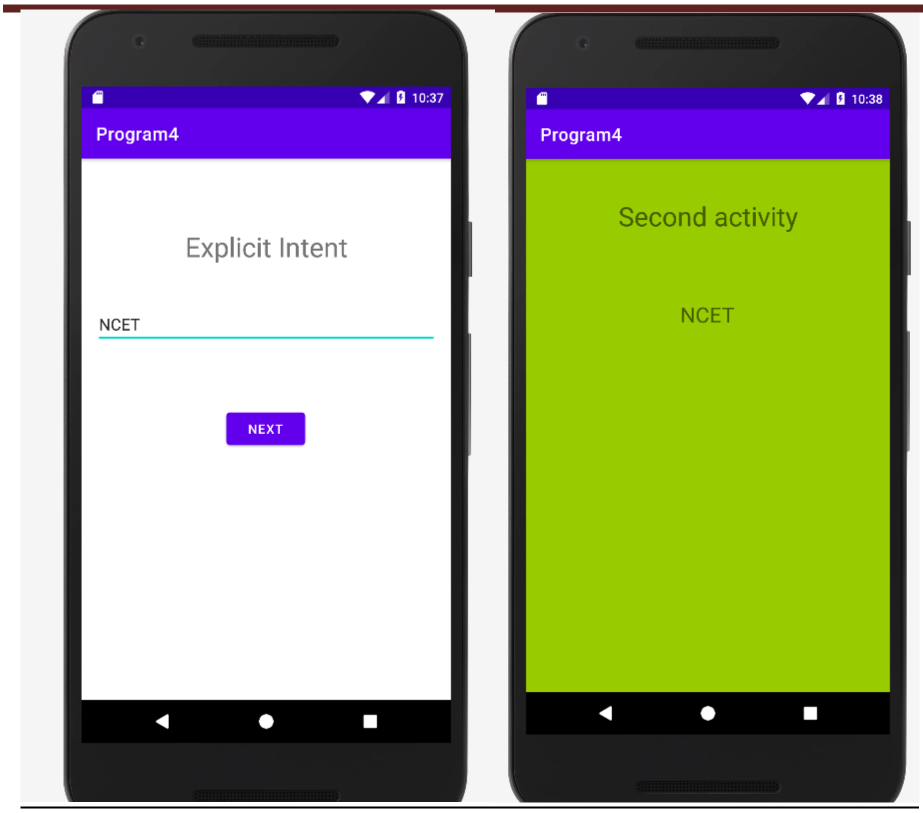
```
package com.example.a6aexplicitintent;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {
   EditText e;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e=(EditText) findViewById(R.id.edit);
    }

    public void dosomething(View view) {
        Intent i1=new Intent(this,Second.class);
        i1.putExtra("user",e.getText().toString());
        startActivity(i1);
    }
}
```

Second Activity XML code: activity_second.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#99cc00"
    tools:context=".Second">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second activity"
        android:textSize="30sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.079" />

    <TextView
        android:id="@+id/result"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.28" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Second Activity Java code: Second.java

```
package com.example.a6aexplicitintent;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.TextView;

public class Second extends AppCompatActivity {
   TextView t;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        t=(TextView) findViewById(R.id.result);
       Bundle b1= getIntent().getExtras();
       String s1=b1.getString("user");
       t.setText(s1);
    }
}
```


10:37
Program4
Explicit Intent
NCET
NEXT
10:38
Program4
Second activity
NCET

Program 5:

Create an android application with Activity has three buttons. On click of Buton 1 open the web browser application, on click of Button 2 open the call application and on click of Button 3 open the map application.

XML Code: activity_main.xml

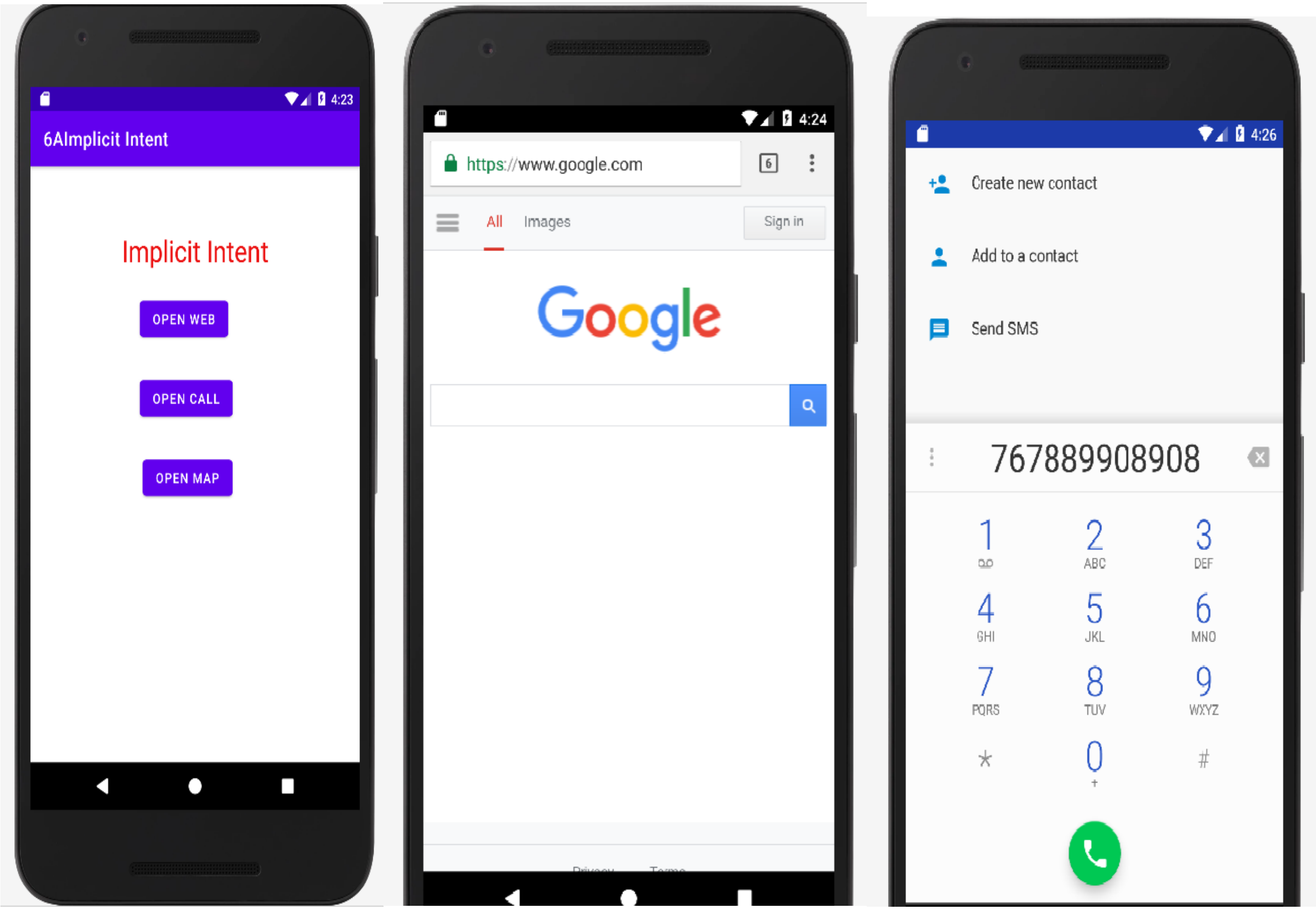
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
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

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Implicit Intent"
        android:textColor="#EC1010"
        android:textSize="30sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.116" />

    <Button
        android:id="@+id/b1"
        android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="openweb"
    android:text="open web"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.454"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.235" />

<Button
    android:id="@+id/b2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="opendial"
    android:text="open call"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.462"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.38" />

<Button
    android:id="@+id/b3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="openmap"
    android:text="open map"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
```

```
        app:layout_constraintHorizontal_bias="0.469"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.525" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Java code: MainActivity.java

```
package com.example.a6aimplicitintent;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void openweb(View view) {
             Intent i1=new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.google.com"));
              startActivity(i1);
  }
    public void opendial(View view) {
           Intent i2=new Intent(Intent.ACTION_VIEW,Uri.parse("tel:767889908908"));
              startActivity(i2);}
        public void openmap(View view) {
        Intent i3=new Intent(Intent.ACTION_VIEW,Uri.parse("geo:20.593,78.9629"));
        startActivity(i3);
    }
}
```

Output



Program 6:

Develop an android application to Create a login Activity. It asks “username” and “password” from user. If username and password are valid, it displays Welcome message using new activity.

XML Code:activity_main.xml

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

    <EditText
        android:id="@+id/etName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:autofillHints=""
        android:ems="10"
        android:hint="USER NAME"
        android:inputType="textPersonName"
        app:layout_constraintTop_toBottomOf="@id/etName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.326" />
    <EditText
        android:id="@+id/etPassword"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:autofillHints=""
        android:ems="10"
        android:hint="PASSWORD"
        android:inputType="numberPassword"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
```

```
 app:layout_constraintTop_toBottomOf="@+id/etName"
        app:layout_constraintVertical_bias="0.064" />

 <Button
        android:onClick="Login"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="SIGN IN"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etPassword"
        app:layout_constraintVertical_bias="0.098" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Java code: MainActivity.java

```
package com.example.loginform;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    EditText uname,password;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        uname= (EditText) findViewById(R.id.etName) ;
        password=(EditText) findViewById(R.id.etPassword);

    }
    public void Login(View view) {
        if(uname.getText().toString().equals("Admin")&&password.getText().toString().equals("1234"))
        {
            Intent i=new Intent(this,Second.class);
            startActivity(i);
            Toast.makeText(this, " Login Successful", Toast.LENGTH_SHORT).show();

        }
        else
        {
           Toast.makeText(this, "Login Failure", Toast.LENGTH_SHORT).show();
        }

    }
}
```

Second.java

```
package com.example.loginform;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;

public class Second  extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.show);
    }
}
```

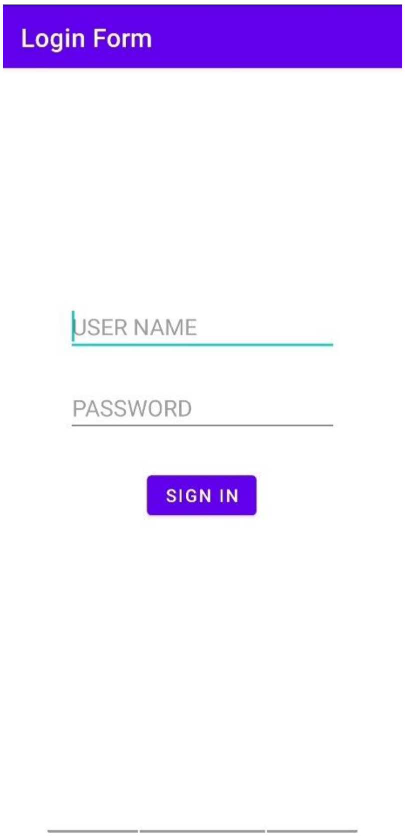
activity_Second.xml

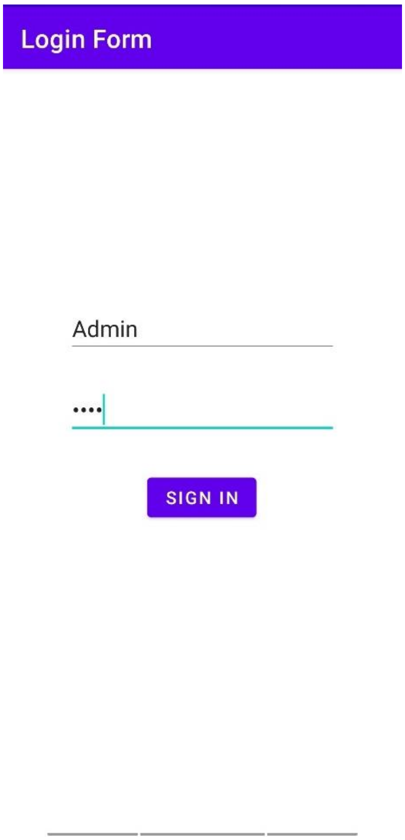
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="WELCOME"
        tools:layout_editor_absoluteX="146dp"
        tools:layout_editor_absoluteY="196dp"
        tools:ignore="MissingConstraints" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Note*

Create activity.xml and mainactivity.java file then go to Project java folder→ Right click→new->Activity→Empty activity→ give file name as second.java and activity_Second.xml

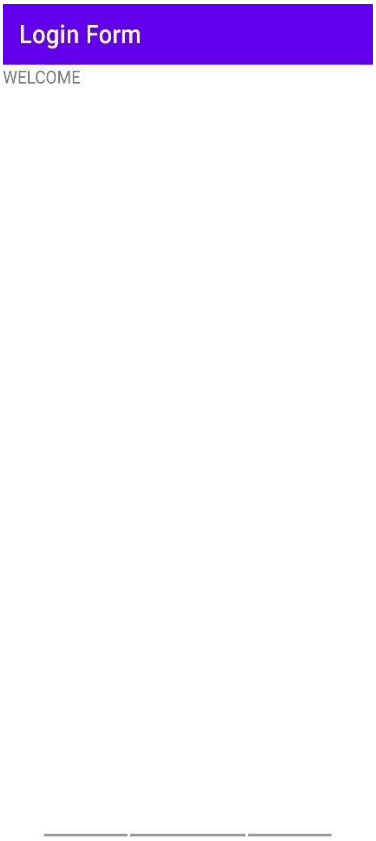
OUTPUT





Admin

••••



WELCOME

Program 7:

Develop an android application to design a Simple Calculator application has two edit texts and four buttons. When you enter two numbers and click a button, the application performs the calculation for that button and displays the result.

XML Code: activity_main.xml

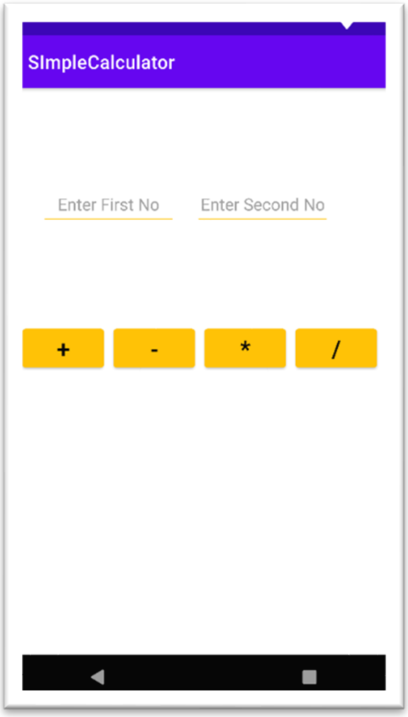
```
<?xmlversion="1.0"encoding="utf-8"?>
    <androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal"
        tools:context=".MainActivity">


            <EditText
                android:id="@+id/firstNo"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="30dp"
                android:layout_marginTop="100dp"
                android:gravity="center"
                android:inputType="number"
                android:backgroundTint="#FFBF00"
                android:ems="7"
                android:hint="EnterFirstNo"
                android:minHeight="48dp"/>

            <EditText
                android:id="@+id/secondtNo"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="20dp"
                android:layout_marginTop="100dp"
                android:backgroundTint="#FFBF00"
                android:ems="7"
                android:inputType="number"
                android:gravity="center"
                android:hint="EnterSecondNo"
                android:minHeight="48dp"/>


        <TextView
            android:id="@+id/output"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
    android:layout_marginTop="36dp"
    android:gravity="center"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout"/>


    <Button
      android:onClick="add"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="+"
      android:textColor="@color/black"
      android:backgroundTint="#FFBF00"
      android:gravity="center"
      android:textSize="25dp"
      android:layout_marginLeft="10dp"
     android:layout_marginTop="10dp"/>

    <Button
      android:onClick="sub"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="@color/black"
      android:text="-"
      android:backgroundTint="#FFBF00"
      android:gravity="center"
      android:textSize="25dp"
      android:layout_marginLeft="10dp
     android:layout_marginTop="10dp"/>


<Button
      android:onClick="mul"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="@color/black"
      android:text="*"
      android:gravity="center"
      android:textSize="25dp"
      android:backgroundTint="#FFBF00"
      android:layout_marginLeft="10dp"
      android:layout_marginTop="10dp"/>

    <Button
      android:onClick="div"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
```

```
            android:text="/"
            android:gravity="center"
            android:backgroundTint="#FFBF00"
            android:textColor="@color/black"
            android:layout_marginLeft="10dp"
            android:textSize="25dp
            android:layout_marginTop="10dp"/>

    </androidx.constraintlayout.widget.ConstraintLayout>
```

Java Code:Mainactivity.java

```
package com.example.simplecalculator;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity
{
    int number1, number2;
    TextView output;
    EditText no1,no2;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
         no1 = (EditText) findViewById(R.id.firstNo);
         no2 = (EditText) findViewById(R.id.secondtNo);
         output =(TextView) findViewById(R.id.output);
    }

    public void add(View view)
    {
        number1 = Integer.parseInt(no1.getText().toString());
        number2 = Integer.parseInt(no2.getText().toString());
        output.setText(String.valueOf(number1+number2));
    }
    public void sub(View view)
    {
        number1 = Integer.parseInt(no1.getText().toString());
        number2 = Integer.parseInt(no2.getText().toString());
        output.setText(String.valueOf(number1-number2));
     }
    public void mul(View view)
    {
        number1 = Integer.parseInt(no1.getText().toString());
        number2 = Integer.parseInt(no2.getText().toString());
        output.setText(String.valueOf(number1*number2));
    }
    public void div(View view)
    {
        number1 = Integer.parseInt(no1.getText().toString());
        number2 = Integer.parseInt(no2.getText().toString());
        output.setText(String.valueOf(number1/number2));
    }
}
```

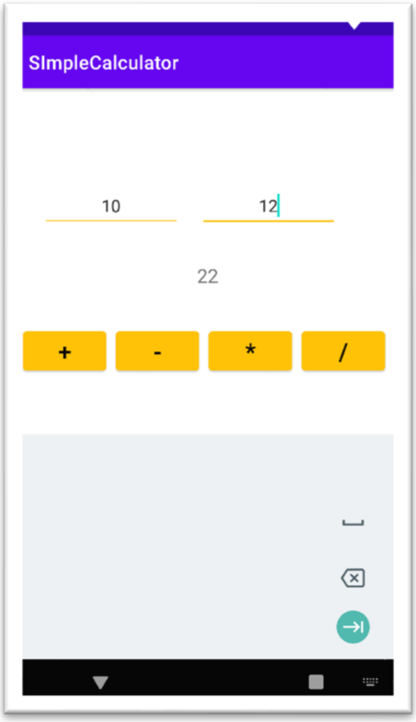
OUTPUT



SImpleCalculator

Enter First No

Enter Second No

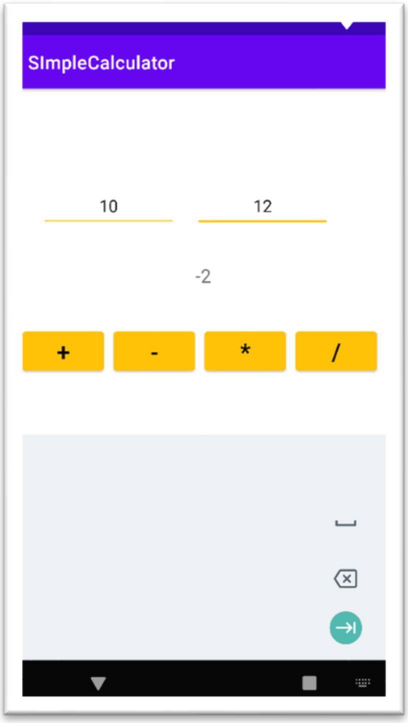


SImpleCalculator

10

12

22



SImpleCalculator

10

12

-2

Program 8:

Develop a simple application with one EditText so that the user can write some text in it. Create a button called "Convert Text to Speech" that converts the user input text into voice.

XML Code:activitymain.xml

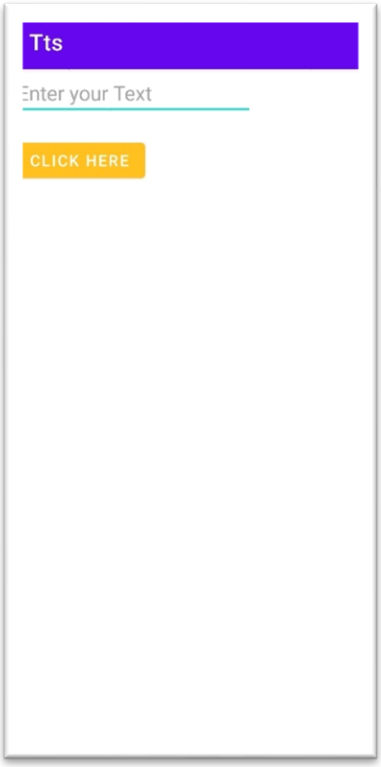
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
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

    <EditText
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="100dp"
        android:ems="10"
        android:hint="Enter your Text"
        android:minHeight="48dp"
        tools:ignore="MissingConstraints" />

    <Button
        android:onClick="voice"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="20dp"
        android:text="Click Here"
        android:backgroundTint="#FFBE1A"
        app:layout_constraintTop_toBottomOf="@id/text"
        tools:ignore="MissingConstraints" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Java Code:Mainactivity.java

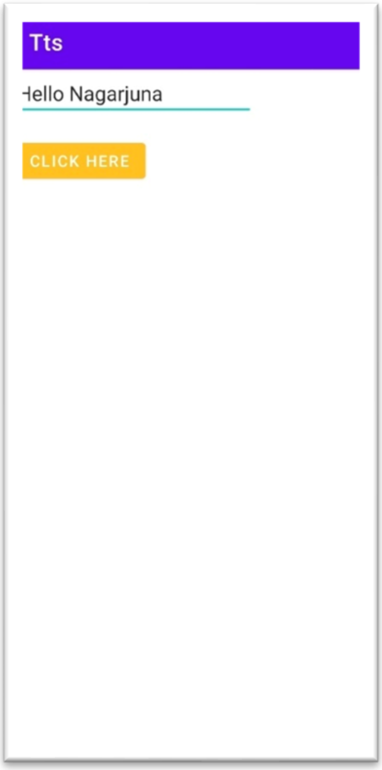
```
package com.example.tts;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.speech.tts.TextToSpeech;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity
```

```
{

    EditText textData;
    TextToSpeech tts;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void voice(View view)
    {
        textData = (EditText) findViewById(R.id.text);
        tts = new TextToSpeech(getApplicationContext(), new TextToSpeech.OnInitListener()
        {
            @Override
            public void onInit(int i)
            {
                if (i == TextToSpeech.SUCCESS)
                {
                    tts.speak(textData.getText().toString(), TextToSpeech.QUEUE_ADD, null, null);
                }
            }

        });
    }
}
```

OUTPUT



Enter your Text



Hello Nagarjuna

VIVA QUESTIONS

1. What is Android?

It is an open-sourced operating system that is used primarily on mobile devices, such as cell phones and tablets. It is a Linux kernel-based system that's been equipped with rich components that allows developers to create and run apps that can perform both basic and advanced functions.</p>

2. What Is the Google Android SDK?

The Google Android SDK is a tool set that developers need in order to write apps on Android enabled devices. It contains a graphical interface that emulates an Android driven handheld environment, allowing them to test and debug their codes.

3. How many key components are there in Android Architecture

Android Architecture is made up of 4 key components:

4. Describe the Android Framework.

The Android Framework is an important aspect of the Android Architecture. Here you can find all the classes and methods that developers would need in order to write applications on the Android environment.

5. What is AAPT?

AAPT is short for Android Asset Packaging Tool. This tool provides developers with the ability to deal with zip-compatible archives, which includes creating, extracting as well as viewing its contents.

6. What is the importance of having an emulator within the Android environment?

The emulator lets developers "play" around an interface that acts as if it were an actual mobile device. They can write and test codes, and even debug. Emulators are a safe place for testing codes especially if it is in the early design phase.

7. What is the use of an activity Creator?

An activity Creator is the first step towards the creation of a new Android project. It is made up of a shell script that will be used to create new file system structure necessary for writing codes within the Android IDE.

8. Describe Activities.

Activities are what you refer to as the window to a user interface. Just as you create windows in order to display output or to ask for an input in the form of dialog boxes, activities play the same role, though it may not always be in the form of a user interface.

9. What are Intents?

Intents displays notification messages to the user from within the Android enabled device. It can be used to alert the user of a particular state that occurred. Users can be made to respond to intents.

10. Differentiate Activities from Services.

Activities can be closed, or terminated anytime the user wishes. On the other hand, services are designed to run behind the scenes, and can act independently. Most services.

11. What items are important in every Android project?

These are the essential items that are present each time an Android project is created:
Android Manifest.xml
build.xml
bin/
src/
res/
assets/

12. What is the importance of XML-based layouts?

The use of XML-based layouts provides a consistent and somewhat standard means of setting GUI definition format. In common practice, layout details are placed in XML files while other items are placed in source files.

13. What are containers?

Containers, as the name itself implies, holds objects and widgets together, depending on which specific items are needed and in what particular arrangement that is wanted. Containers may hold labels, fields, buttons, or even child containers, as examples.

14. What is Orientation?

Orientation, which can be set using set Orientation(), dictates if the Linear Layout is represented as a row or as a column. Values are set as either HORIZONTAL or VERTICAL.

15. What is the importance of Android in the mobile market?

Developers can write and register apps that will specifically run under the Android environment. This means that every mobile device that is Android enabled will be able to support and run these apps. With the growing popularity of Android mobile devices, developers can take advantage of this trend by creating and uploading their apps on the Android Market for distribution to anyone who wants to download it.

16. What do you think are some disadvantages of Android?

Given that Android is an open-source platform, and the fact that different Android operating systems have been released on different mobile devices, there's no clear cut policy to how applications can adapt with various OS versions and upgrades.
–One app that runs on this particular version of Android OS may or may not run on another version.
–Another disadvantage is that since mobile devices such as phones and tabs come in different sizes and forms, it poses a challenge for developers to create apps that can adjust correctly to the right screen size and other varying features and specs.

17. What is adb?

Adb is short for "Android Debug Bridge". It allows developers the power to execute remote shell commands. Its basic function is to allow and control communication towards and from the emulator port.

18. What are the four essential states of an activity?

Active – if the activity is at the foreground

Paused – if the activity is at the background and still visible
Stopped – if the activity is not visible and therefore is hidden or obscured by another activity Destroyed – when the activity process is killed or completed terminated

19. What is ANR?
ANR is short for Application Not Responding. This is actually a dialog that appears to the user whenever an application have been unresponsive for a long period of time.

20. Which elements can occur only once and must be present?
Among the different elements, the and elements must be present and can occur only once. The rest are optional, and can occur as many times as needed.

21. How are escape characters used as attribute?
Escape characters are preceded by double backslashes. For example, a newline character is created using

22. What is the importance of settings permissions in app development?
Permissions allow certain restrictions to be imposed primarily to protect data and code. Without these, codes could be compromised, resulting to defects in functionality.

23. What is the function of an intent filter?
Because every component needs to indicate which intents they can respond to, intent filters are used to filter out intents that these components are willing to receive. One or more intent filters are possible, depending on the services and activities that is going to make use of it

24. **Enumerate the three key loops when monitoring an activity?**
Entire lifetime – activity happens between on Create and on Destroy Visible lifetime – activity happens between on Start and on Stop Foreground lifetime – activity happens between on Resume and on Pause

25. When is the on Stop(. method invoked?
A call to on Stop method happens when an activity is no longer visible to the user, either because another activity has taken over or if in front of that activity.

26. Is there a case wherein other qualifiers in multiple resources take precedence over locale?
Yes, there are actually instances wherein some qualifiers can take precedence over locale. There are two known exceptions, which are the MCC (mobile country code. and MNC (mobile network code. Qualifiers).

.