***N.B.K.R Institute of Science & Technology***

***AUTONOMOUS***

***Vidyanagar***

***CERTIFICATE***

***Certified that this is a bonafide record of practical work done in the……………………………………………………………………… Laboratory by Mr./Ms.……………………………………………………. Roll No. ………………………… during the year ………………….........***

***Number of Experiments done:***

***Staff Member in-charge:***

***Examiner:***

***INDEX***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***S.NO*** | ***Date*** | ***Name of the Experiment*** | ***Page No*** | ***Grade*** |
| 1 |  | **Session-I:** Review of Java Concepts, Download and Install Android Studio, Android Setup, Application components, Resources, Activities, Services.  Develop a Hello World Program |  |  |
|  |  |  |  |  |
| 2 |  | **Session-II:** Android User Interfaces: UI Layouts, UI Controls, Styles and Themes.  Develop an application that uses GUI components, Font and Colours |  |  |
|  |  |  |  |  |
| 3 |  | **Session-III:** Android Event Handling, Drag and Drop, Notifications.  Develop a basic Calculator application that uses Layout Managers and event listeners |  |  |
|  |  |  |  |  |
| 4 |  | **Session-IV:** Alert Dialogues, Clipboard, Animation Develop an application that draws basic graphical primitives on the screen. |  |  |
|  |  |  |  |  |
| 5 |  | **Session-V:** Sending SMS, Phone calls. Develop an application for SMS, Phone Calls. |  |  |
|  |  |  |  |  |
| 6 |  | **Session-VI:** Android Audio Capture, Audio Manager, Audio Complete.  Develop a program for Audio Capture. |  |  |
|  |  |  |  |  |
| 7 |  | **Session-VII:** SqLite (CRUD).  Develop an Android Program to connect Database and Develop Database Operations using SQLite. |  |  |
|  |  |  |  |  |
| 8 |  | **Session-VIII:** Text to Speech  Develop an Application for Text to Speech. |  |  |
|  |  |  |  |  |
| 9 |  | **Session-IX:** Google Maps  Develop an application for identify the current location. |  |  |
|  |  |  |  |  |

# **Session-I:** Review of Java Concepts, Download and Install Android Studio, Android Setup,Application components, Resources, Activities, Services.

Develop a Hello World Program

**Introduction to Java:**

**JAVA** was developed by **James Gosling** at **Sun Microsystems** in the year **1991**, later acquired by Oracle Corporation. It is a simple programming language. Java makes writing, compiling, and debugging programming easy. It helps to create reusable code and modular programs.

[Java](https://www.geeksforgeeks.org/java/) is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to write once run anywhere that is compiled Java code can run on all platforms that support Java. Java applications are compiled to byte code that can run on any Java Virtual Machine. The syntax of Java is similar to C/C++.

***History***

Java’s history is very interesting. It is a programming language created in 1991. James Gosling, Mike Sheridan, and Patrick Naughton, a team of Sun engineers known as the **Green team** initiated the Java language in 1991. **Sun Microsystems** released its first public implementation in 1996 as **Java 1.0**.

It provides no-cost -run-times on popular platforms. Java1.0 compiler was re-written in Java by Arthur Van Hoff to strictly comply with its specifications. With the arrival of Java 2, new versions had multiple configurations built for different types of platforms.

In 1997, Sun Microsystems approached the ISO standards body and later formalized Java, but it soon withdrew from the process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

On November 13, 2006, Sun released much of its Java virtual machine as free, open- source software. On May 8, 2007, Sun finished the process, making all of its JVM’s core code available under open-source distribution terms.

The principles for creating java were simple, robust, secured, high performance, portable, multi- threaded, interpreted, dynamic, etc. **James Gosling in 1995** developed Java, who is known as the **Father of Java**. Currently, Java is used in mobile devices, internet programming, games, e-business, etc.

**Java programming language is named JAVA. Why?**

After the name OAK, the team decided to give a new name to it and the suggested words were Silk, Jolt, revolutionary, DNA, dynamic, etc. These all names were easy to spell and fun to say, but they all wanted the name to reflect the essence of technology. In accordance with James Gosling, **Java** the among the top names along with **Silk**, and since java was a unique name so most of them preferred it.

Java is the name of an **island** in Indonesia where the first coffee (named java coffee) was produced. And this name was chosen by James Gosling while having coffee near his office. Note that Java is just a name, not an acronym.

**Java Terminology**

Before learning Java, one must be familiar with these common terms of Java.

1. **Java Virtual Machine(JVM):** This is generally referred to as [JVM.](https://www.geeksforgeeks.org/jvm-works-jvm-architecture/#%3A~%3Atext%3DJVM(Java%20Virtual%20Machine)%20acts%2C(Write%20Once%20Run%20Anywhere)) There are three execution phases of a program. They are written, compile and run the program.
   * Writing a program is done by a java programmer like you and me.
   * The compilation is done by the **JAVAC** compiler which is a primary Java compiler included in the Java development kit (JDK). It takes Java program as input and generates bytecode as output.
   * In the Running phase of a program, **JVM** executes the bytecode generated by the compiler.
   * Now, we understood that the function of Java Virtual Machine is to execute the bytecode produced by the compiler. Every Operating System has a different JVM but the output they produce after the execution of bytecode is the same across all the operating systems. This is why Java is known as a **platform-independent language.**
2. **Bytecode in** the **Development process:** As discussed, the Javac compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. It is saved as **.class** file by the compiler. To view the bytecode, a disassembler like [javap](https://www.geeksforgeeks.org/javap-tool-in-java-with-examples/) can be used.
3. **Java Development Kit (JDK):** While we were using the term JDK, when we learn about bytecode and JVM . So, as the name suggests, it is a complete Java development kit that includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.
4. **Java Runtime Environment (JRE):** JDK includes JRE. JRE installation on our computers allows the java program to run, however, we cannot compile it. JRE includes a browser, JVM, applet supports, and plugins. For running the java program, a computer needs JRE.
5. **Garbage Collector:** In Java, programmers can’t delete the objects. To delete or recollect that memory JVM has a program called [Garbage Collector.](https://www.geeksforgeeks.org/garbage-collection-java/) Garbage Collectors can recollect the of objects that are not referenced. So, Java makes the life of a programmer easy by handling memory management. However, programmers should be careful about their code whether they are using objects that have been used for a long time. Because Garbage cannot recover the memory of objects being referenced.
6. **Class Path:** The [class path](https://www.geeksforgeeks.org/classpath-in-java/) is the file path where the java runtime and Java compiler look for **.class** files to load. By default, JDK provides many libraries. If you want to include external libraries, they should be added to the class path.

**Primary/Main Features of Java**

1. **Platform Independent:** Compiler converts source code to bytecode and then the JVM executes the bytecode generated by the compiler. This bytecode can run on any platform be it Windows, Linux, macOS which means if we compile a program on Windows, then we can run it on Linux and vice versa. Each operating system has a different JVM, but the output produced by all the OS is the same after the execution of bytecode. That is why we call java a platform-independent language.
2. **Object-Oriented Programming Language:** Organizing the program in the terms of collection of objects is a way of object-oriented programming, each of which represents an instance of the class. The four main concepts of Object-Oriented programming are:
   * Abstraction
   * Encapsulation
   * Inheritance
   * Polymorphism
3. **Simple:** Java is one of the simple languages as it does not have complex features like pointers, operator overloading, multiple inheritances, Explicit memory allocation.
4. **Robust:** Java language is robust that means reliable. It is developed in such a way that it puts a lot of effort into checking errors as early as possible, that is why the java compiler is able to detect even those errors that are not easy to detect by another programming language. The main features of java that make it robust are garbage collection, Exception Handling, and memory allocation.
5. **Secure:** In java, we don’t have pointers, and so we cannot access out-of-bound arrays i.e it shows **ArrayIndexOutOfBound Exception** if we try to do so. That’s why several security flaws like stack corruption or buffer overflow are impossible to exploit in Java.
6. **Distributed:** We can create distributed applications using the java programming language. Remote Method Invocation and Enterprise Java Beans are used for creating distributed applications in java. The java programs can be easily distributed on one or more systems that are connected to each other through an internet connection.
7. **Multithreading:** Java supports multithreading. It is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU.
8. **Portable:** As we know, java code written on one machine can be run on another machine. The platform-independent feature of java in which its platform-independent bytecode can be taken to any platform for execution makes java portable.
9. **High Performance:** Java architecture is defined in such a way that it reduces overhead during the runtime and at some time java uses Just In Time (JIT) compiler where the compiler compiles code on- demand basics where it only compiles those methods that are called making applications to execute faster.
10. **Dynamic flexibility:** Java being completely object-oriented gives us the flexibility to add classes, new methods to existing classes and even creating new classes through sub-classes. Java even supports functions written in other languages such as C, C++ which are referred to as native methods.
11. **Sandbox Execution:** Java programs run in a separate space that allows user to execute their applications without affecting the underlying system with help of a bytecode verifier. Bytecode verifier also provides additional security as its role is to check the code for any violation access.
12. **Write Once Run Anywhere:** As discussed above java application generates ‘.class’ file which corresponds to our applications(program) but contains code in binary format. It provides ease t architecture-neutral ease as bytecode is not dependent on any machine architecture. It is the primary reason java is used in the enterprising IT industry globally worldwide.
13. **Power of compilation and interpretation:** Most languages are designed with purpose either they are compiled language or they are interpreted language. But java integrates arising enormous power as Java compiler compiles the source code to bytecode and JVM executes this bytecode to machine OS- dependent executable code.

## Minimum Theoretical Background

**Step 1 -** Setup Java Development Kit (JDK) You can download the latest version of Java JDK from Oracle's Java site: Java SE Downloads. You will find instructions for installing JDK in downloaded files, follow the given instructions to install and configure the setup. Finally, set PATH and JAVA\_HOME environment variables to refer to the directory that contains java and javac, typically java\_install\_dir/bin and java\_install\_dir respectively. If you are running Windows and have installed the JDK in C:\jdk1.6.0\_15, you would have to put the following line in your C:\autoexec.batfile.

set PATH=C:\jdk1.6.0\_15\bin;%PATH% set JAVA\_HOME=C:\jdk1.6.0\_15

**Step 2 -** Setup Android SDK You can download the latest version of Android SDK from Android’s official website: [http://developer.android.com/sdk/index.html.](http://developer.android.com/sdk/index.html) If you are installing SDK on Windows machine, then you will find ainstaller\_rXXwindows. exe, so just download and run this exe which will launch Android SDK Tool Setup wizard to guide you throughout the installation, so just follow the instructions carefully. Finally, you will have Android SDK Tools installed on your machine. If you

are installing SDK either on Mac OS or Linux, check the instructions provided along with the downloaded android-sdk\_rXX-macosx.zip file for Mac OS and androidsdk\_rXX-linux.tgz file for Linux. This tutorial will consider that you are going to setup your environment on Windows machine having Windows 7 operating system.

**Step 3 -** Setup Android Development Tools (ADT) Plugin This step will help you in setting Android Development Tool plugin for Eclipse. Let's start with launching Eclipse and then, choose Help > Software Updates > Install New Software. This will display the following dialogue box.

**Step 4 -** Create Android Virtual Device to test your Android applications you will need a virtual Android device. So before we start writing our code, let us create an Android virtual device. Launch Android AVD Manager using Eclipse menu options Window > AVD Manager> which will launch Android AVD Manager. Use New button to create a new Android Virtual Device and enter the following information, before clicking Create AVD button.

Android Development Tools (ADT) is a plugin for the Android studio that is designed to give you a powerful, integrated environment in which to build Android applications. ADT extends the capabilities of Android studio to let you quickly set up new Android projects, create an application UI, add components based on the Android Framework API, debug your applications using the Android SDK tools, and even export signed (or unsigned) APKs in order to distribute your application. Developing in Android studio with ADT is highly recommended and is the fastest way to get started.

With the guided project setup, it provides, as well as tools integration, custom XML editors, and debug output pane, applications. ADT gives you an incredible boost in developing Android applications.

In android studio students must be aware of the directory structure and the control flow of the program. Program should be either executed on the android mobile phones or on the suitable emulators. To execute a simple program, like to display Hello World on screen syntax of writing a program in android is pre-requisite as the programming language used is JAVA only. The main activity code is a Java file MainActivity.java. This is the actual application file which ultimately gets converted to a Dalvik executable and runs your application.

Following is the default code generated by the application wizard for Hello World! application:

**AIM:** To develop a Hello World Program

**PROGRAM:**

**MainActivity.java**

package com.example.helloworld;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

TextView helloTextView = findViewById(R.id.helloTextView);

helloTextView.setText("Hello World!");

}

}

**activity\_main.xml**

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<TextView

android:id="@+id/helloTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Hello World!"

android:textSize="74sp"

android:layout\_centerInParent="true" />

</RelativeLayout>

**ActivityManifest.xml:**

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

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

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

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Hello\_World"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

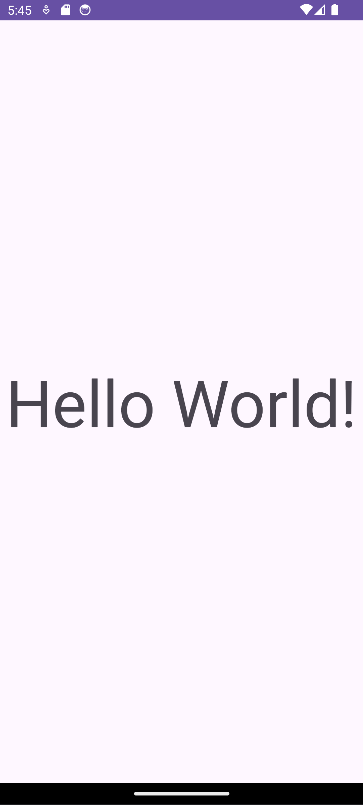
</application>

</manifest>

The Manifest File: Whatever component you develop as a part of your application, you must declare all its components in a manifest file called AndroidManifest.xml which resides at the root of the application project directory. This file works as an interface between Android OS and your application, so if you do not declare your component in this file, then it will not be considered by the OS. The <activity> tag is used to specify an activity and android:name attribute specifies the fully qualified class name of the Activity subclass and the android:label attributes specifies a string to use as the label for the activity. You can specify multiple activities using <activity> tags. The action for the intent filter is named android.intent.action.MAIN to indicate that this activity serves as the entry point for the application. The category for the intent- filter is named android.intent.category.LAUNCHER to indicate that the application can be launched from the device's launcher icon. The @string refers to the strings.xml file explained below. Hence, @string/app\_name refers to the app\_name string defined in the strings.xml file, which is "HelloWorld". Similar way, other strings get populated in the application. Following is the list of tags which you will use in your manifest file to specify different Android application components:

1. <activity>elements for activities
2. <service> elements for services
3. <receiver> elements for broadcast receivers
4. <provider> elements for content providers

**OUTPUT:**

****

**Result:**

Thus, the android application Hello World program is developed and tested using android studio.

# **Session-II:** Android User Interfaces: UI Layouts, UI Controls, Styles and Themes. Develop an application that uses GUI components, Font and Colors

## AIM:

To develop an android application that uses GUI components, Font and Colours using android studio.

## PROCEDURE:

1. Open eclipse or android studio and select new android project.
2. Give the project name and select next.
3. Choose the android version. Choose the lowest android version (Android 2.2) and select next.
4. Enter the package name. Package name must be two words separated by comma and click finish.
5. Go to package explorer on the left-hand side. Select the project.
6. Go to **“res“folder** and select layout. Double click the **activitymain.xml** file.
7. Now you can see the Graphics layout window.
8. Click the **activitymain.xml** file and type the code.
9. Go to project explorer and select **“src”** folder. Now select **mainactivity.java** file and type the code.
10. Go to **activitymain.xml** and right click. Select run as option and select run configuration.
11. Android output is present in the android emulator.

**PROGRAM:**

**MainActivity.java**

package com.example.gui;

import android.app.Activity;

import android.graphics.Color;

import android.graphics.Typeface;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

public class MainActivity extends Activity{

float font=24;

int i=1;

int fontType = Typeface.NORMAL; // Default font style

@Override

public void onCreate(Bundle savedInstance) {

super.onCreate(savedInstance);

setContentView(R.layout.activity\_main);

final TextView t1=(TextView)findViewById(R.id.textView1);

Button b1=(Button)findViewById(R.id.button1);

Button b2=(Button)findViewById(R.id.button2);

Button b3=(Button)findViewById(R.id.button3);

b1.setOnClickListener(new View.OnClickListener() {

public void onClick(View view) {

t1.setTextSize(font);

font=font+4;

if(font==40)

font=20;

}

});

b2.setOnClickListener(new View.OnClickListener() {

public void onClick(View view) {

switch(i){

case 1:

t1.setTextColor(Color.parseColor("#0000FF"));

break;

case 2:

t1.setTextColor(Color.parseColor("#00FF00"));

break;

case 3:

t1.setTextColor(Color.parseColor("#FF0000"));

break;

case 4:

t1.setTextColor(Color.parseColor("#F00000"));

break;

}

i++;

if(i==5)

i=1;

}

});

b3.setOnClickListener(new View.OnClickListener() {

public void onClick(View view) {

switch(fontType){

case Typeface.NORMAL:

t1.setTypeface(null, Typeface.BOLD);

fontType = Typeface.BOLD;

break;

case Typeface.BOLD:

t1.setTypeface(null, Typeface.ITALIC);

fontType = Typeface.ITALIC;

break;

case Typeface.ITALIC:

t1.setTypeface(null, Typeface.NORMAL);

fontType = Typeface.BOLD\_ITALIC;

break;

}

}

});

}

}

**activity\_main.xml**

<LinearLayout

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

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="vertical">

<TextView

android:id="@+id/textView1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_margin="20sp"

android:gravity="center"

android:text="Hello World!"

android:textSize="20sp"

android:textStyle="bold"/>

<Button

android:id="@+id/button1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:text="change font size"

android:textSize="20sp"/>

<Button

android:id="@+id/button2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:text="change color"

android:textSize="20sp"/>

<Button

android:id="@+id/button3"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:text="change font"

android:textSize="20sp"/>

</LinearLayout>

**AndroidManifest:**

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

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

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

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Gui"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

**Output:**

****

**Result:**

Thus, the android application GUI components, Font size, Font style and Font Colors is developed and tested using android studio.

**Session-III:** Android Event Handling, Drag and Drop, Notifications.

# Develop a basic Calculator application that uses Layout Managers and event listeners.

## Aim:

To develop an android application that uses Layout Managers and event listeners using android studio.

## Procedure:

1. Open eclipse or android studio and select new android project.
2. Give project name and select next.
3. Choose the android version. Choose the lowest android version(Android 2.2) and select next.
4. Enter the package name. Package name must be two word separated by comma and click finish.
5. Go to package explorer in the left hand side. Select the project.
6. Go to **“res“** folder and select layout. Double click the **activitymain.xml** file.
7. Now you can see the Graphics layout window.
8. Click the **activitymain.xml** file and type the code.
9. Go to project explorer and select **“src”** folder. Now select **mainactivity.java** file and type the code.
10. Go to **activitymain.xml** and right click. Select run as option and select run configuration.
11. Android output is present in the android emulator.

## MainActivity.xml

package com.example.calculator;

import android.app.Activity;

import android.os.Bundle;

import android.text.TextUtils;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class MainActivity extends Activity implements View.OnClickListener {

EditText input1;

EditText input2;

Button addition;

Button subtraction;

Button multiplication;

Button division;

TextView tvResult;

String opera = " ";

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

input1 = (EditText) findViewById(R.id.etNum1);

input2 = (EditText) findViewById(R.id.etNum2);

addition = (Button) findViewById(R.id.btnAdd);

subtraction =(Button) findViewById(R.id.btnSub);

multiplication = (Button)findViewById(R.id.btnMult);

division = (Button)findViewById(R.id.btnDiv);

tvResult = (TextView) findViewById(R.id.tvResult);

addition.setOnClickListener(this);

subtraction.setOnClickListener(this);

multiplication.setOnClickListener(this);

division.setOnClickListener(this);

}

@Override

public void onClick(View v) {

float num1 = 0;

float num2 = 0;

float result = 0;

if (TextUtils.isEmpty(input1.getText().toString()) || TextUtils.isEmpty(input2.getText().toString()))

return;

num1 = Float.parseFloat(input1.getText().toString());

num2 = Float.parseFloat(input2.getText().toString());

int id = v.getId();

if(id==R.id.btnAdd){

opera = "+";

result = num1 + num2;

}

else if(id==R.id.btnSub){

opera = "-";

result = num1 - num2;

}

else if(id==R.id.btnMult){

opera = "\*";

result = num1 \* num2;

}

else if(id==R.id.btnDiv){

opera = "/";

result = num1 / num2;

}

else

{

Toast.makeText(getApplicationContext(),"invalid input",Toast.LENGTH\_LONG).show();

}

tvResult.setText(num1 + " " + opera + " " + num2 + " = " + result);

}

}

**activity\_main.xml:**

<?xmlVersion ="1.0" encoding="utf-8?>

<LinearLayout

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

android:orientation="vertical"

android:layout\_width="fill\_parent"

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

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/linearLayout1"

android:layout\_marginLeft="10pt"

android:layout\_marginRight="10pt"

android:layout\_marginTop="3pt">

<EditText

android:layout\_weight="1"

android:layout\_height="wrap\_content"

android:layout\_marginRight="5pt"

android:id="@+id/etNum1"

android:layout\_width="match\_parent"

android:inputType="numberDecimal">

</EditText>

<EditText

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:layout\_marginStart="5pt"

android:id="@+id/etNum2"

android:layout\_width="match\_parent"

android:inputType="numberDecimal">

</EditText>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/linearLayout2"

android:layout\_marginTop="3pt"

android:layout\_marginLeft="5pt"

android:layout\_marginRight="5pt">

<Button

android:layout\_height="wrap\_content"

android:layout\_width="match\_parent"

android:layout\_weight="1"

android:text="+"

android:textSize="15pt"

android:id="@+id/btnAdd">

</Button>

<Button

android:layout\_height="wrap\_content"

android:layout\_width="match\_parent"

android:layout\_weight="1"

android:text="-"

android:textSize="15pt"

android:id="@+id/btnSub">

</Button>

<Button

android:layout\_height="wrap\_content"

android:layout\_width="match\_parent"

android:layout\_weight="1"

android:text="\*"

android:textSize="15pt"

android:id="@+id/btnMult">

</Button>

<Button

android:layout\_height="wrap\_content"

android:layout\_width="match\_parent"

android:layout\_weight="1"

android:text="/"

android:textSize="15pt"

android:id="@+id/btnDiv">

</Button>

</LinearLayout>

<TextView

android:id="@+id/tvResult"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginLeft="5pt"

android:layout\_marginTop="3pt"

android:layout\_marginRight="5pt"

android:gravity="center\_horizontal"

android:textSize="12pt"></TextView>

</LinearLayout>

**AndroidManifest:**

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

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

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

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Calculator"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

**Output:**

A white sheet of paper

Description automatically generated

**Result:**

Thus, the android application Calculator is developed and tested using android studio.

**Session-IV:** Alert Dialogues, Clipboard, Animation

# Develop an application that draws basic graphical primitives on the screen.

## AIM:

To write an application that draws basic graphical primitives on the screen using android studio.

## PROCEDURE:

1. Open eclipse or android studio and select new android project.
2. Give project name and select next.
3. Choose the android version. Choose the lowest android version(Android 2.2) and select next.
4. Enter the package name. Package name must be two word separated by comma and click finish.
5. Go to package explorer in the left hand side. Select the project.
6. Go to **“res“** folder and select layout. Double click the **activitymain.xml** file.
7. Now you can see the Graphics layout window.
8. Click the **activitymain.xml** file and type the code.
9. Go to project explorer and select **“src”** folder. Now select **mainactivity.java** and create **samplecanvas.java** files and type the code.
10. Go to **activitymain.xml** and right click. Select run as option and select run configuration.
11. Android output is present in the android emulator.

## PROGRAM:

## MainActivity.xml

## package com.example.graphical;

## import android.content.Context;

## import android.graphics.Canvas;

## import android.graphics.Color;

## import android.graphics.Paint;

## import android.os.Bundle;

## import android.view.View;

## import ndroid.appcompat.app.AppCompatActivity;

## public class MainActivity extends AppCompatActivity

## {

## @Override

## protected void onCreate(Bundle savedInstanceState)

## {

## super.onCreate(savedInstanceState);

## setContentView(new myView(this));

## }

## private class myView extends View

## {

## public myView(Context Context)

## {

## super(Context);

## }

## @Override

## protected void onDraw(Canvas Canvas)

## {

## super.onDraw(Canvas);

## Paint paint = new Paint();

## paint.setTextSize(40);

## Canvas.drawText(“Circle”,52,30,paint);

## paint.setColor(Color.GREEN);

## Canvas.drawCircle(100,150,100,paint);

## paint.setColor(Color.RED);

## Canvas.drawText(“Rectangle”,255,30,paint);

## paint.setColor(Color.YELLOW);

## Canvas.drawRect(250,50,400,350,paint);

## paint.setColor(Color.GREEN);

## Canvas.drawText(“SQUARE”,55,430,paint);

## paint.setColor(Color.BLUE);

## Canvas.drawRect(50,450,150,550,paint);

## paint.setColor(Color.GREEN);

## Canvas.drawText(“LINE”,255,430,paint);

## paint.setColor(Color.CYAN);

## Canvas.drawLine(250,500,350,500,paint);

## }

## }

## }

**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>

**AndroidManifest:**

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

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

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

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Graphical"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

**Output:**



**Result:**

Thus, the application that draws basic graphical primitives on the screen is developed and tested using android studio.

**Session-V:** Sending SMS, Phone calls.

# Develop an application for SMS and Phone Calls.

**SMS:**

**AIM:** To develop an application for SMS.

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | You will use Android Studio IDE to create an Android application and name it as *tutorialspoint* under a package *com.example.tutorialspoint*. |
| 2 | Modify *src/MainActivity.java* file and add required code to take care of sending sms. |
| 3 | Modify layout XML file *res/layout/activity\_main.xml* add any GUI component if required. I'm adding a simple GUI to take mobile number and SMS text to be sent and a simple button to send SMS. |
| 4 | No need to define default string constants at res/values/strings.xml. Android studio takes care of default constants. |
| 5 | Modify *AndroidManifest.xml* as shown below |
| 6 | Run the application to launch Android emulator and verify the result of the changes done in the application. |

**Program:**

**MainActivity.java**

package com.example.sms;

import android.os.Bundle;

import android.telephony.SmsManager;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

EditText phonenumber, message;

Button send;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

send = findViewById(R.id.button);

phonenumber = findViewById(R.id.editText);

message = findViewById(R.id.editText2);

send.setOnClickListener(new View.OnClickListener() {

public void onClick(View view) {

String number = phonenumber.getText().toString();

String msg = message.getText().toString();

if (isValidIndianNumber(number)) {

try {

SmsManager smsManager = SmsManager.getDefault();

smsManager.sendTextMessage(number, null, msg, null, null);

Toast.makeText(getApplicationContext(), "Message Sent", Toast.LENGTH\_LONG).show();

} catch (Exception e) {

Toast.makeText(getApplicationContext(), "Failed to send message", Toast.LENGTH\_LONG).show();

}

} else {

Toast.makeText(getApplicationContext(), "Please enter a valid Indian number", Toast.LENGTH\_LONG).show();

}

}

});

}

private boolean isValidIndianNumber(String number) {

return number.startsWith("+91");

}

}

**AndroidManifest.xml**

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

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

xmlns:dist="http://schemas.android.com/apk/distribution"

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

package="com.example.sms">

<uses-feature

android:name="android.hardware.telephony"

android:required="false" />

<uses-permission android:name="android.permission.SEND\_SMS"/>

<dist:module dist:instant="true" />

<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/Theme.Sms">

<activity

android:name="com.example.sms.MainActivity"

android:exported="true"

tools:ignore="MissingClass">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

**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:orientation="vertical"

android:layout\_marginTop="140dp"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<EditText

android:id="@+id/editText"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:ems="10"

android:hint="Enter number"

android:inputType="textPersonName" />

<EditText

android:id="@+id/editText2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:ems="10"

android:hint="Enter message"

android:inputType="textPersonName" />

<Button

android:id="@+id/button"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

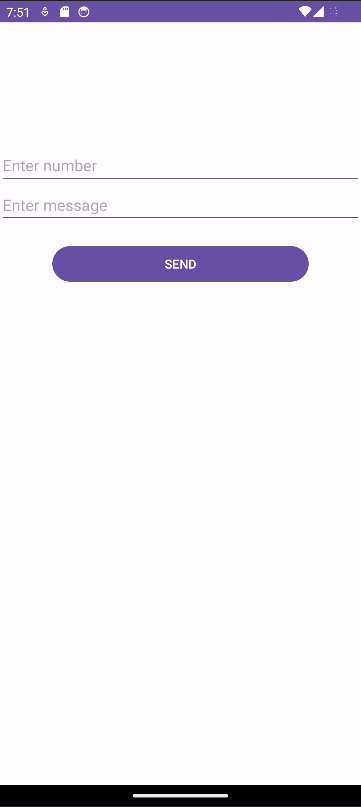
android:layout\_marginLeft="60dp"

android:layout\_marginRight="60dp"

android:text="SEND" />

</LinearLayout>

**Output:**

****

**Result:**

The application that SMS on the screen is developed and tested using android studio.

**PHONE CALLS:**

**AIM:** To develop an application for Phone Calls

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | You will use Android studio IDE to create an Android application and name it as *My Application* under a package *com.example.saira\_000.myapplication*. |
| 2 | Modify *src/MainActivity.java* file and add required code to take care of making a call. |
| 3 | Modify layout XML file *res/layout/activity\_main.xml* add any GUI component if required. I'm adding a simple button to Call 91-000-000-0000 number |
| 4 | No need to define default string constants.Android studio takes care of default constants. |
| 5 | Modify *AndroidManifest.xml* as shown below |
| 6 | Run the application to launch Android emulator and verify the result of the changes done in the application. |

**PROGRAM:**

**MainActivity.java:**

package com.example.phone\_calls;

import android.Manifest;

import android.content.Intent;

import android.content.pm.PackageManager;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.widget.EditText;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

public class MainActivity extends AppCompatActivity {

private static final int REQUEST\_CALL\_PHONE = 1;

private EditText editTextPhoneNumber;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

editTextPhoneNumber = findViewById(R.id.editTextPhoneNumber);

}

public void makePhoneCall(View view) {

String phoneNumber = editTextPhoneNumber.getText().toString().trim();

if (!phoneNumber.isEmpty()) {

if (ContextCompat.checkSelfPermission(MainActivity.this, Manifest.permission.CALL\_PHONE) != PackageManager.PERMISSION\_GRANTED) {

ActivityCompat.requestPermissions(MainActivity.this, new String[]{Manifest.permission.CALL\_PHONE}, REQUEST\_CALL\_PHONE);

} else {

startPhoneCall(phoneNumber);

}

} else {

Toast.makeText(this, "Please enter a phone number", Toast.LENGTH\_SHORT).show();

}

}

private void startPhoneCall(String phoneNumber) {

Intent intent = new Intent(Intent.ACTION\_CALL);

intent.setData(Uri.parse("tel:" + phoneNumber));

startActivity(intent);

}

@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults);

if (requestCode == REQUEST\_CALL\_PHONE) {

if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

String phoneNumber = editTextPhoneNumber.getText().toString().trim();

if (!phoneNumber.isEmpty()) {

startPhoneCall(phoneNumber);

} else {

Toast.makeText(this, "Please enter a phone number", Toast.LENGTH\_SHORT).show();

}

} else {

Toast.makeText(this, "Permission Denied", Toast.LENGTH\_SHORT).show();

}

}

}

}

**activity\_main.xml:**

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<EditText

android:id="@+id/editTextPhoneNumber"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter Phone Number"

android:inputType="phone" />

<Button

android:id="@+id/btnCall"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Make Phone Call"

android:layout\_below="@id/editTextPhoneNumber"

android:layout\_centerHorizontal="true"

android:onClick="makePhoneCall" />

</RelativeLayout>

**ActivityManifest:**

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

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

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

<uses-feature

android:name="android.hardware.telephony"

android:required="false" />

<uses-permission android:name="android.permission.CALL\_PHONE" />

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Phone\_calls"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

**OUTPUT:**

A screenshot of a phone

Description automatically generated

**RESULT:**

The application that Phone Calls on the screen is developed and tested using android studio.

# **Session-VI:** Android Audio Capture, Audio Manager, Audio Complete.

Develop a program for Audio Capture.

|  |  |
| --- | --- |
| **Steps** | **Description** |
| 1 | You will use Android studio IDE to create an Android application and name it as AudioCapture under a package com.example.sairamkrishna.myapplication. |
| 2 | Modify src/MainActivity.java file to add AudioCapture code |
| 3 | Modify layout XML file res/layout/activity\_main.xml add any GUI component if required. |
| 4 | Modify AndroidManifest.xml to add necessary permissions. |
| 5 | Run the application and choose a running android device and install the application on it and verify the results. |

**Program:**

**MainActivity.java**

package com.example.audio;

import android.Manifest;

import android.content.pm.PackageManager;

import android.media.MediaRecorder;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

import java.io.IOException;

public class MainActivity extends AppCompatActivity {

private static final int REQUEST\_PERMISSION\_CODE = 100;

private MediaRecorder mediaRecorder;

private String audioFilePath;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

// Request necessary permissions

requestPermissions();

// Set the path to store the audio file

audioFilePath = getExternalCacheDir().getAbsolutePath() + "/audio.3gp";

}

private void requestPermissions() {

if (ContextCompat.checkSelfPermission(this, Manifest.permission.RECORD\_AUDIO) != PackageManager.PERMISSION\_GRANTED ||

ContextCompat.checkSelfPermission(this, Manifest.permission.WRITE\_EXTERNAL\_STORAGE) != PackageManager.PERMISSION\_GRANTED) {

ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.RECORD\_AUDIO, Manifest.permission.WRITE\_EXTERNAL\_STORAGE}, REQUEST\_PERMISSION\_CODE);

}

}

public void startRecording(View view) {

try {

mediaRecorder = new MediaRecorder();

mediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);

mediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.THREE\_GPP);

mediaRecorder.setOutputFile(audioFilePath);

mediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR\_NB);

mediaRecorder.prepare();

mediaRecorder.start();

// Toggle visibility of buttons

Button startButton = findViewById(R.id.startRecordingButton);

Button stopButton = findViewById(R.id.stopRecordingButton);

startButton.setVisibility(View.GONE);

stopButton.setVisibility(View.VISIBLE);

Toast.makeText(this, "Recording started", Toast.LENGTH\_SHORT).show();

} catch (IOException e) {

e.printStackTrace();

}

}

public void stopRecording(View view) {

mediaRecorder.stop();

mediaRecorder.release();

mediaRecorder = null;

// Toggle visibility of buttons

Button startButton = findViewById(R.id.startRecordingButton);

Button stopButton = findViewById(R.id.stopRecordingButton);

startButton.setVisibility(View.VISIBLE);

stopButton.setVisibility(View.GONE);

Toast.makeText(this, "Recording stopped", Toast.LENGTH\_SHORT).show();

}

}

**activity\_main.xml**:

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<Button

android:id="@+id/startRecordingButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Start Recording"

android:layout\_centerInParent="true"

android:onClick="startRecording" />

<Button

android:id="@+id/stopRecordingButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Stop Recording"

android:layout\_below="@id/startRecordingButton"

android:layout\_centerHorizontal="true"

android:onClick="stopRecording"

android:visibility="gone" />

</RelativeLayout>

**ActivityManifest**:

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

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

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

<uses-permission android:name="android.permission.RECORD\_AUDIO" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Audio"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

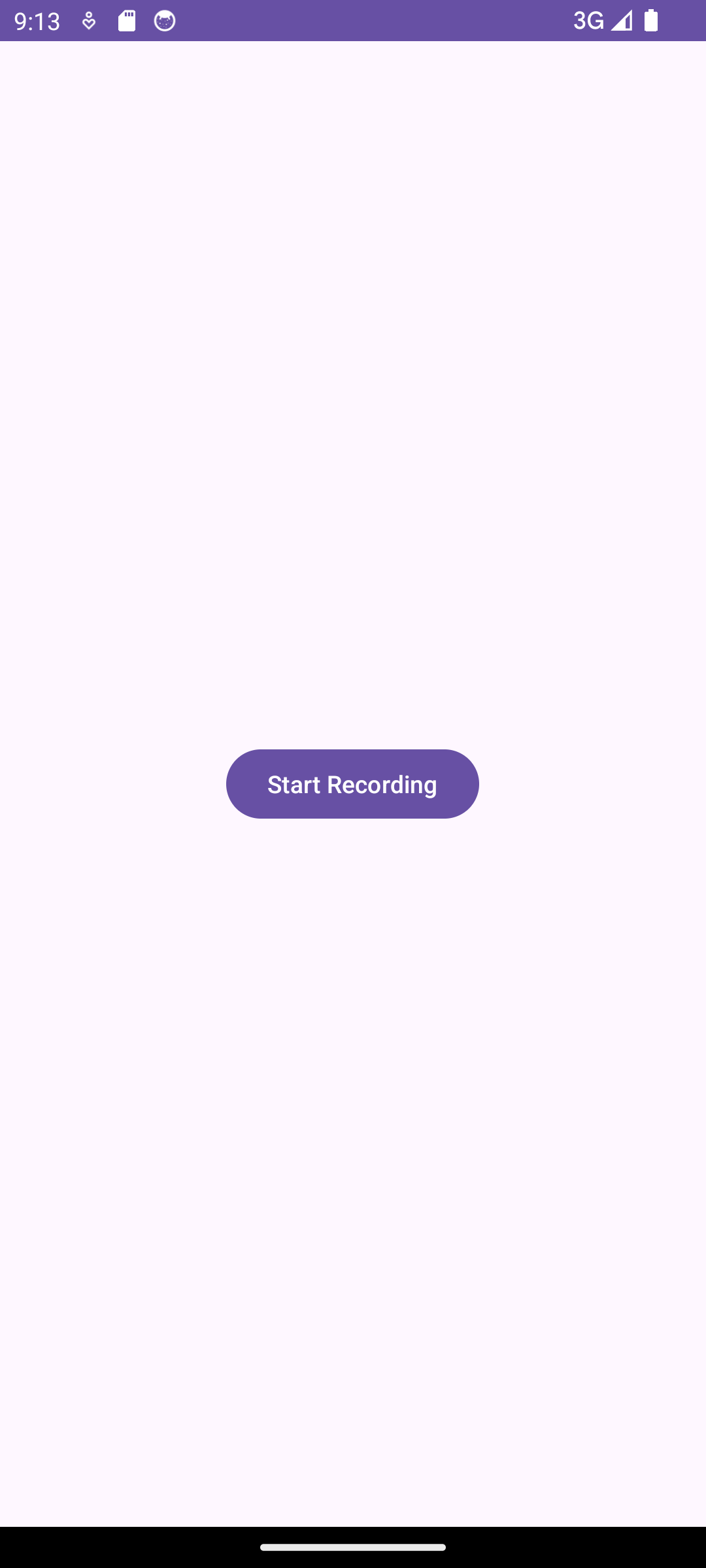
</intent-filter>

</activity>

</application>

</manifest>

**OUTPUT:**

****

**RESULT:**

The application that Audio capture on the screen is developed and tested using android studio.

**Session-VII:**SqLite (CRUD).

# Develop an Android Program to connect Database and Develop Database Operations using SQLite.

It creates basic contacts applications that allows insertion, deletion and modification of contacts.

|  |  |
| --- | --- |
| **Steps** | **Description** |
| 1 | You will use Android studio to create an Android application under a package com.example.sairamkrishna.myapplication. |
| 2 | Modify src/MainActivity.java file to get references of all the XML components and populate the contacts on listView. |
| 3 | Create new src/DBHelper.java that will manage the database work |
| 4 | Create a new Activity as DisplayContact.java that will display the contact on the screen |
| 5 | Modify the res/layout/activity\_main to add respective XML components |
| 6 | Modify the res/layout/activity\_display\_contact.xml to add respective XML components |
| 7 | Modify the res/values/string.xml to add necessary string components |
| 8 | Modify the res/menu/display\_contact.xml to add necessary menu components |
| 9 | Create a new menu as res/menu/mainmenu.xml to add the insert contact option |
| 10 | Run the application and choose a running android device and install the application on it and verify the results. |

Following is the content of the modified **MainActivity.java**. package com.example.sairamkrishna.myapplication;

import android.content.Context; import android.content.Intent;

import android.support.v7.app.ActionBarActivity; import android.os.Bundle;

import android.view.KeyEvent; import android.view.Menu; import android.view.MenuItem; import android.view.View;

import android.widget.AdapterView; import android.widget.ArrayAdapter;

import android.widget.AdapterView.OnItemClickListener; import android.widget.ListView;

import java.util.ArrayList; import java.util.List;

public class MainActivity extends ActionBarActivity {

public final static String EXTRA\_MESSAGE = "MESSAGE"; private ListView obj;

DBHelper mydb;

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_main);

mydb = new DBHelper(this);

ArrayList array\_list = mydb.getAllCotacts();

ArrayAdapter arrayAdapter=new ArrayAdapter(this,android.R.layout.simple\_list\_item\_1, array\_list);

obj = (ListView)findViewById(R.id.listView1); obj.setAdapter(arrayAdapter); obj.setOnItemClickListener(new OnItemClickListener(){

@Override

public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,long arg3) {

// TODO Auto-generated method stub int id\_To\_Search = arg2 + 1;

Bundle dataBundle = new Bundle(); dataBundle.putInt("id", id\_To\_Search);

Intent intent = new Intent(getApplicationContext(),DisplayContact.class);

intent.putExtras(dataBundle); startActivity(intent);

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present. getMenuInflater().inflate(R.menu.menu\_main, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item){ super.onOptionsItemSelected(item);

switch(item.getItemId()) {

case R.id.item1:Bundle dataBundle = new Bundle(); dataBundle.putInt("id", 0);

Intent intent = new Intent(getApplicationContext(),DisplayContact.class); intent.putExtras(dataBundle);

startActivity(intent); return true;

default:

return super.onOptionsItemSelected(item);

}

}

public boolean onKeyDown(int keycode, KeyEvent event) { if (keycode == KeyEvent.KEYCODE\_BACK) {

moveTaskToBack(true);

}

return super.onKeyDown(keycode, event);

}

}

Following is the modified content of display contact activity **DisplayContact.java**

package com.example.sairamkrishna.myapplication;

import android.os.Bundle; import android.app.Activity; import android.app.AlertDialog;

import android.content.DialogInterface; import android.content.Intent;

import android.database.Cursor;

import android.view.Menu; import android.view.MenuItem; import android.view.View;

import android.widget.Button; import android.widget.TextView; import android.widget.Toast;

public class DisplayContact extends Activity { int from\_Where\_I\_Am\_Coming = 0;

private DBHelper mydb ;

TextView name ; TextView phone; TextView email; TextView street; TextView place;

int id\_To\_Update = 0;

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_display\_contact);

name = (TextView) findViewById(R.id.editTextName); phone = (TextView) findViewById(R.id.editTextPhone); email = (TextView) findViewById(R.id.editTextStreet); street = (TextView) findViewById(R.id.editTextEmail); place = (TextView) findViewById(R.id.editTextCity);

mydb = new DBHelper(this);

Bundle extras = getIntent().getExtras(); if(extras !=null) {

int Value = extras.getInt("id");

if(Value>0){

//means this is the view part not the add contact part. Cursor rs = mydb.getData(Value);

id\_To\_Update = Value; rs.moveToFirst();

String nam = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS\_COLUMN\_NAME)); String phon = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS\_COLUMN\_PHONE)); String emai = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS\_COLUMN\_EMAIL)); String stree = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS\_COLUMN\_STREET)); String plac = rs.getString(rs.getColumnIndex(DBHelper.CONTACTS\_COLUMN\_CITY));

if (!rs.isClosed()) { rs.close();

}

Button b = (Button)findViewById(R.id.button1); b.setVisibility(View.INVISIBLE);

name.setText((CharSequence)nam); name.setFocusable(false); name.setClickable(false);

phone.setText((CharSequence)phon); phone.setFocusable(false); phone.setClickable(false);

email.setText((CharSequence)emai); email.setFocusable(false); email.setClickable(false);

street.setText((CharSequence)stree); street.setFocusable(false); street.setClickable(false);

place.setText((CharSequence)plac); place.setFocusable(false); place.setClickable(false);

}

}

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present. Bundle extras = getIntent().getExtras();

if(extras !=null) {

int Value = extras.getInt("id"); if(Value>0){

getMenuInflater().inflate(R.menu.display\_contact, menu);

} else{

getMenuInflater().inflate(R.menu.menu\_main menu);

}

}

return true;

}

public boolean onOptionsItemSelected(MenuItem item) { super.onOptionsItemSelected(item); switch(item.getItemId()) {

case R.id.Edit\_Contact:

Button b = (Button)findViewById(R.id.button1); b.setVisibility(View.VISIBLE); name.setEnabled(true); name.setFocusableInTouchMode(true); name.setClickable(true);

phone.setEnabled(true); phone.setFocusableInTouchMode(true); phone.setClickable(true);

email.setEnabled(true); email.setFocusableInTouchMode(true); email.setClickable(true);

street.setEnabled(true); street.setFocusableInTouchMode(true); street.setClickable(true);

place.setEnabled(true); place.setFocusableInTouchMode(true); place.setClickable(true);

return true;

case R.id.Delete\_Contact:

AlertDialog.Builder builder = new AlertDialog.Builder(this); builder.setMessage(R.string.deleteContact)

.setPositiveButton(R.string.yes, new DialogInterface.OnClickListener() { public void onClick(DialogInterface dialog, int id) {

mydb.deleteContact(id\_To\_Update); Toast.makeText(getApplicationContext(), "Deleted Successfully",

Toast.LENGTH\_SHORT).show();

Intent intent = new Intent(getApplicationContext(),MainActivity.class); startActivity(intent);

}

})

.setNegativeButton(R.string.no, new DialogInterface.OnClickListener() { public void onClick(DialogInterface dialog, int id) {

// User cancelled the dialog

}

});

AlertDialog d = builder.create(); d.setTitle("Are you sure"); d.show();

return true; default:

return super.onOptionsItemSelected(item);

}

}

public void run(View view) {

Bundle extras = getIntent().getExtras(); if(extras !=null) {

int Value = extras.getInt("id"); if(Value>0){

if(mydb.updateContact(id\_To\_Update,name.getText().toString(), phone.getText().toString(), email.getText().toString(),

street.getText().toString(), place.getText().toString())){ Toast.makeText(getApplicationContext(), "Updated", Toast.LENGTH\_SHORT).show();

Intent intent = new Intent(getApplicationContext(),MainActivity.class); startActivity(intent);

} else{

Toast.makeText(getApplicationContext(), "not Updated", Toast.LENGTH\_SHORT).show();

}

} else{

if(mydb.insertContact(name.getText().toString(), phone.getText().toString(),

email.getText().toString(), street.getText().toString(), place.getText().toString())){

Toast.makeText(getApplicationContext(), "done",

Toast.LENGTH\_SHORT).show();

} else{

Toast.makeText(getApplicationContext(), "not done",

Toast.LENGTH\_SHORT).show();

}

Intent intent = new Intent(getApplicationContext(),MainActivity.class); startActivity(intent);

}

}

}

}

Following is the content of Database class **DBHelper.java**

package com.example.sairamkrishna.myapplication; import java.util.ArrayList;

import java.util.HashMap; import java.util.Hashtable;

import android.content.ContentValues; import android.content.Context; import android.database.Cursor;

import android.database.DatabaseUtils;

import android.database.sqlite.SQLiteOpenHelper; import android.database.sqlite.SQLiteDatabase;

public class DBHelper extends SQLiteOpenHelper {

public static final String DATABASE\_NAME = "MyDBName.db"; public static final String CONTACTS\_TABLE\_NAME = "contacts"; public static final String CONTACTS\_COLUMN\_ID = "id";

public static final String CONTACTS\_COLUMN\_NAME = "name"; public static final String CONTACTS\_COLUMN\_EMAIL = "email"; public static final String CONTACTS\_COLUMN\_STREET = "street"; public static final String CONTACTS\_COLUMN\_CITY = "place"; public static final String CONTACTS\_COLUMN\_PHONE = "phone"; private HashMap hp;

public DBHelper(Context context) { super(context, DATABASE\_NAME , null, 1);

}

@Override

public void onCreate(SQLiteDatabase db) {

// TODO Auto-generated method stub db.execSQL(

"create table contacts " +

"(id integer primary key, name text,phone text,email text, street text,place text)"

);

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

// TODO Auto-generated method stub db.execSQL("DROP TABLE IF EXISTS contacts"); onCreate(db);

}

public boolean insertContact (String name, String phone, String email, String street,String place) { SQLiteDatabase db = this.getWritableDatabase();

ContentValues contentValues = new ContentValues(); contentValues.put("name", name); contentValues.put("phone", phone); contentValues.put("email", email); contentValues.put("street", street); contentValues.put("place", place); db.insert("contacts", null, contentValues);

return true;

}

public Cursor getData(int id) {

SQLiteDatabase db = this.getReadableDatabase();

Cursor res = db.rawQuery( "select \* from contacts where id="+id+"", null ); return res;

}

public int numberOfRows(){

SQLiteDatabase db = this.getReadableDatabase();

int numRows = (int) DatabaseUtils.queryNumEntries(db, CONTACTS\_TABLE\_NAME); return numRows;

}

public boolean updateContact (Integer id, String name, String phone, String email, String street,String place) {

SQLiteDatabase db = this.getWritableDatabase(); ContentValues contentValues = new ContentValues(); contentValues.put("name", name); contentValues.put("phone", phone); contentValues.put("email", email); contentValues.put("street", street); contentValues.put("place", place);

db.update("contacts", contentValues, "id = ? ", new String[] { Integer.toString(id) } ); return true;

}

public Integer deleteContact (Integer id) { SQLiteDatabase db = this.getWritableDatabase(); return db.delete("contacts",

"id = ? ",

new String[] { Integer.toString(id) });

}

public ArrayList<String> getAllCotacts() { ArrayList<String> array\_list = new ArrayList<String>();

//hp = new HashMap();

SQLiteDatabase db = this.getReadableDatabase();

Cursor res = db.rawQuery( "select \* from contacts", null ); res.moveToFirst();

while(res.isAfterLast() == false){ array\_list.add(res.getString(res.getColumnIndex(CONTACTS\_COLUMN\_NAME))); res.moveToNext();

}

return array\_list;

}

}

Following is the content of the **res/layout/activity\_main.xml**

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

<RelativeLayout xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) xmlns:t[ools="http://schemas.android.com/tools"](http://schemas.android.com/tools) android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin" android:paddingRight="@dimen/activity\_horizontal\_margin" android:paddingTop="@dimen/activity\_vertical\_margin" android:paddingBottom="@dimen/activity\_vertical\_margin" tools:context=".MainActivity">

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:id="@+id/textView" android:layout\_alignParentTop="true" android:layout\_centerHorizontal="true" android:textSize="30dp" android:text="Data Base" />

<TextView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Tutorials Point" android:id="@+id/textView2" android:layout\_below="@+id/textView" android:layout\_centerHorizontal="true" android:textSize="35dp" android:textColor="#ff16ff01" />

<ImageView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:id="@+id/imageView" android:layout\_below="@+id/textView2" android:layout\_centerHorizontal="true" android:src="@drawable/logo"/>

<ScrollView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:id="@+id/scrollView" android:layout\_below="@+id/imageView" android:layout\_alignParentLeft="true" android:layout\_alignParentStart="true" android:layout\_alignParentBottom="true" android:layout\_alignParentRight="true" android:layout\_alignParentEnd="true">

<ListView android:id="@+id/listView1" android:layout\_width="match\_parent" android:layout\_height="wrap\_content"

android:layout\_centerHorizontal="true" android:layout\_centerVertical="true" >

</ListView>

</ScrollView>

</RelativeLayout>

Following is the content of the **res/layout/activity\_display\_contact.xml**

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

<ScrollView xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) xmlns:t[ools="http://schemas.android.com/tools"](http://schemas.android.com/tools) android:id="@+id/scrollView1"

android:layout\_width="match\_parent" android:layout\_height="wrap\_content" tools:context=".DisplayContact" >

<RelativeLayout android:layout\_width="match\_parent" android:layout\_height="370dp"

android:paddingBottom="@dimen/activity\_vertical\_margin" android:paddingLeft="@dimen/activity\_horizontal\_margin" android:paddingRight="@dimen/activity\_horizontal\_margin" android:paddingTop="@dimen/activity\_vertical\_margin">

<EditText

android:id="@+id/editTextName" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_alignParentLeft="true" android:layout\_marginTop="5dp" android:layout\_marginLeft="82dp" android:ems="10" android:inputType="text" >

</EditText>

<EditText android:id="@+id/editTextEmail" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/editTextStreet" android:layout\_below="@+id/editTextStreet" android:layout\_marginTop="22dp" android:ems="10" android:inputType="textEmailAddress" />

<TextView android:id="@+id/textView1" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignBottom="@+id/editTextName" android:layout\_alignParentLeft="true" android:text="@string/name" android:textAppearance="?android:attr/textAppearanceMedium" />

<Button android:id="@+id/button1"

android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_alignLeft="@+id/editTextCity" android:layout\_alignParentBottom="true" android:layout\_marginBottom="28dp" android:onClick="run" android:text="@string/save" />

<TextView android:id="@+id/textView2" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignBottom="@+id/editTextEmail" android:layout\_alignLeft="@+id/textView1" android:text="@string/email" android:textAppearance="?android:attr/textAppearanceMedium" />

<TextView

android:id="@+id/textView5" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_alignBottom="@+id/editTextPhone" android:layout\_alignLeft="@+id/textView1" android:text="@string/phone" android:textAppearance="?android:attr/textAppearanceMedium" />

<TextView android:id="@+id/textView4" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_above="@+id/editTextEmail" android:layout\_alignLeft="@+id/textView5" android:text="@string/street" android:textAppearance="?android:attr/textAppearanceMedium" />

<EditText android:id="@+id/editTextCity" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignRight="@+id/editTextName" android:layout\_below="@+id/editTextEmail" android:layout\_marginTop="30dp" android:ems="10"

android:inputType="text" />

<TextView android:id="@+id/textView3" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignBaseline="@+id/editTextCity" android:layout\_alignBottom="@+id/editTextCity" android:layout\_alignParentLeft="true" android:layout\_toLeftOf="@+id/editTextEmail" android:text="@string/country" android:textAppearance="?android:attr/textAppearanceMedium" />

<EditText android:id="@+id/editTextStreet" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/editTextName" android:layout\_below="@+id/editTextPhone" android:ems="10"

android:inputType="text" >

<requestFocus />

</EditText>

<EditText android:id="@+id/editTextPhone" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/editTextStreet" android:layout\_below="@+id/editTextName" android:ems="10" android:inputType="phone|text" />

</RelativeLayout>

</ScrollView>

Following is the content of the **res/value/string.xml**

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

<resources>

<string name="app\_name">Address Book</string>

<string name="action\_settings">Settings</string>

<string name="hello\_world">Hello world!</string>

<string name="Add\_New">Add New</string>

<string name="edit">Edit Contact</string>

<string name="delete">Delete Contact</string>

<string name="title\_activity\_display\_contact">DisplayContact</string>

<string name="name">Name</string>

<string name="phone">Phone</string>

<string name="email">Email</string>

<string name="street">Street</string>

<string name="country">City/State/Zip</string>

<string name="save">Save Contact</string>

<string name="deleteContact">Are you sure, you want to delete it.</string>

<string name="yes">Yes</string>

<string name="no">No</string>

</resources>

Following is the content of the **res/menu/main\_menu.xml**

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

<menu xmlns:android="[http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) >

<item android:id="@+id/item1" android:icon="@drawable/add" android:title="@string/Add\_New" >

</item>

</menu>

Following is the content of the **res/menu/display\_contact.xml**

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

<menu xmlns:android="[http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) >

<item

android:id="@+id/Edit\_Contact" android:orderInCategory="100" android:title="@string/edit"/>

<item

android:id="@+id/Delete\_Contact" android:orderInCategory="100" android:title="@string/delete"/>

</menu>

This is the defualt **AndroidManifest.xml** of this project

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

<manifest xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) package="com.example.sairamkrishna.myapplication" >

<application android:allowBackup="true" android:icon="@mipmap/ic\_launcher" android:label="@string/app\_name" android:theme="@style/AppTheme" >

<activity android:name=".MainActivity" android:label="@string/app\_name" >

<intent-filter>

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

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

</intent-filter>

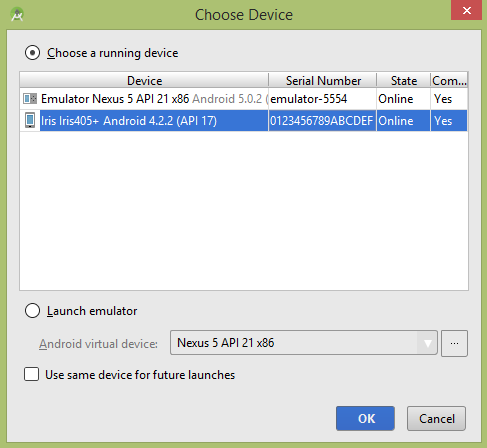
</activity>

<activity android:name=".DisplayContact"/>

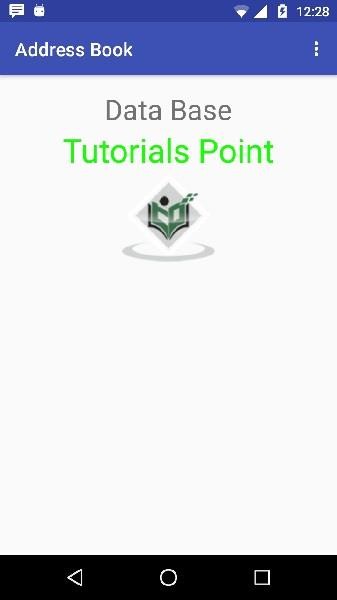
</application>

</manifest>

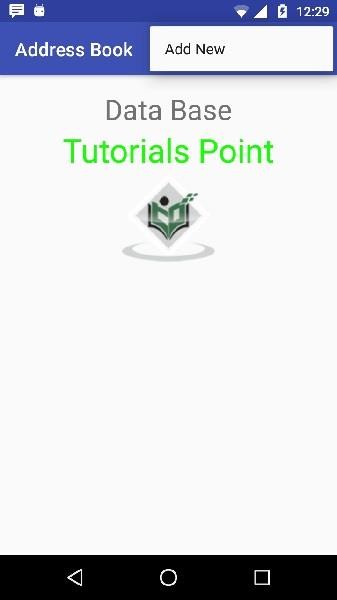
Let's try to run your application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio , open one of your project's activity files and click Run  icon from the tool bar. Before starting your application,Android studio will display following window to select an option where you want to run your Android application.



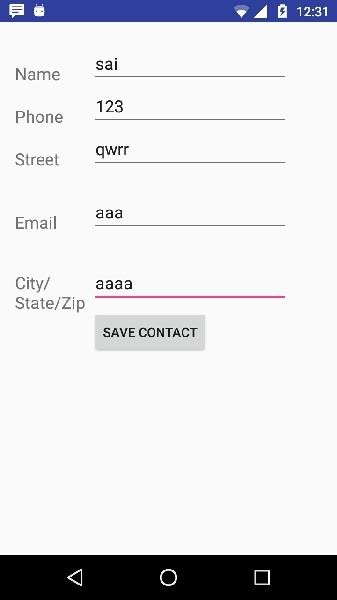
Select your mobile device as an option and then check your mobile device which will display following screen −



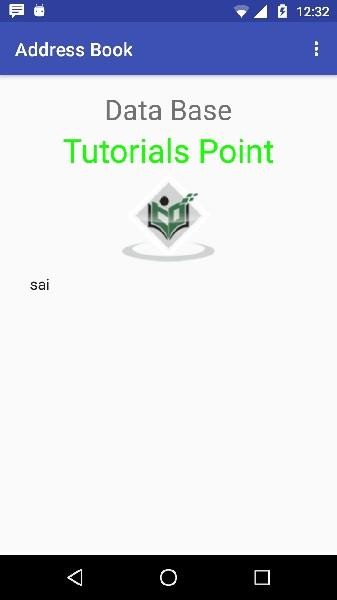
Now open your optional menu, it will show as below image: **Optional menu appears different places on different versions**



Click on the add button of the menu screen to add a new contact. It will display the following screen −



It will display the following fields. Please enter the required information and click on save contact. It will bring you back to main screen.



**Session-VIII:** Text to Speech

# Develop an Application for Text to Speech.

|  |  |
| --- | --- |
| Steps | Description |
| 1 | You will use Android studio to create an Android application under a package com.example.sairamkrishna.myapplication. |
| 2 | Modify src/MainActivity.java file to add TextToSpeech code. |
| 3 | Modify layout XML file res/layout/activity\_main.xml add any GUI component if required. |
| 4 | Run the application and choose a running android device and install the application on it and verify the results. |

**MainActivity.java**

package com.example.textspeech;

import android.os.Bundle;

import android.speech.tts.TextToSpeech;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.Locale;

public class MainActivity extends AppCompatActivity {

private TextToSpeech textToSpeech;

private EditText editText;

private Button btnSpeak;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

editText = findViewById(R.id.editText);

btnSpeak = findViewById(R.id.btnSpeak);

textToSpeech = new TextToSpeech(getApplicationContext(), new TextToSpeech.OnInitListener() {

@Override

public void onInit(int status) {

if (status != TextToSpeech.ERROR) {

textToSpeech.setLanguage(Locale.US);

} else {

Toast.makeText(MainActivity.this, "Initialization failed", Toast.LENGTH\_SHORT).show();

}

}

});

btnSpeak.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String text = editText.getText().toString();

if (!text.isEmpty()) {

textToSpeech.speak(text, TextToSpeech.QUEUE\_FLUSH, null, null);

} else {

Toast.makeText(MainActivity.this, "Please enter some text", Toast.LENGTH\_SHORT).show();

}

}

});

}

@Override

protected void onDestroy() {

super.onDestroy();

if (textToSpeech != null) {

textToSpeech.stop();

textToSpeech.shutdown();

}

}

}

**activity\_main.xml**

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<EditText

android:id="@+id/editText"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter text to speak"

android:layout\_margin="16dp"

android:inputType="textMultiLine"

android:minLines="3" />

<Button

android:id="@+id/btnSpeak"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/editText"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="16dp"

android:text="Speak" />

</RelativeLayout>

**AndroidManifest.xml**

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

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

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

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

<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />

<uses-permission android:name="android.permission.WAKE\_LOCK" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.READ\_CONTACTS" />

<uses-permission android:name="android.permission.READ\_CALENDAR" />

<uses-permission android:name="com.android.vending.BILLING" />

<uses-permission android:name="android.permission.RECORD\_AUDIO" />

<uses-permission android:name="android.permission.MODIFY\_AUDIO\_SETTINGS" />

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.TextSpeech"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

Let's try to run your application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run Eclipse Run Icon icon from the toolbar. Before starting your application, android studio will display following window to select an option where you want to run your Android application.

Android Text To Speech Tutorial

Select your mobile device as an option and then check your mobile device which will display following screen.

Android Text To Speech Tutorial

Now just type some text in the field and click on the text to speech button below. A notification would appear and text will be spoken. It is shown in the image below −

Android Text To Speech Tutorial

Now type something else and repeat the step again with different locale. You will again hear sound. This is shown below −

Let's try to run your application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run  icon from the toolbar. Before starting your application, android studio will display following window to select an option where you want to run your Android application.

**OUTPUT**:

A white rectangle with small dots

Description automatically generated

**RESULT:**

The application that Text Speech on the screen is developed and tested using android studio.

**Session-IX:** Google Maps

# Develop an application for identify the current location.

**AIM:**

To develop an application for identify the current location.

**PROGRAM:**

**MainActivity:**

package com.example.current\_location;

import android.Manifest;

import android.content.pm.PackageManager;

import android.location.Location;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

import com.google.android.gms.location.FusedLocationProviderClient;

import com.google.android.gms.location.LocationCallback;

import com.google.android.gms.location.LocationRequest;

import com.google.android.gms.location.LocationResult;

import com.google.android.gms.location.LocationServices;

public class MainActivity extends AppCompatActivity {

private static final int REQUEST\_PERMISSION\_CODE = 1001;

private FusedLocationProviderClient fusedLocationClient;

private LocationCallback locationCallback;

private boolean isTracking = false;

private Button startStopButton;

private TextView latitudeTextView, longitudeTextView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

startStopButton = findViewById(R.id.startStopButton);

latitudeTextView = findViewById(R.id.latitudeTextView);

longitudeTextView = findViewById(R.id.longitudeTextView);

fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);

if (checkPermission()) {

startLocationUpdates();

} else {

requestPermission();

}

}

private boolean checkPermission() {

return ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION) == PackageManager.PERMISSION\_GRANTED &&

ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_COARSE\_LOCATION) == PackageManager.PERMISSION\_GRANTED;

}

private void requestPermission() {

ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.ACCESS\_FINE\_LOCATION, Manifest.permission.ACCESS\_COARSE\_LOCATION}, REQUEST\_PERMISSION\_CODE);

}

public void startStopTracking(View view) {

if (!isTracking) {

startLocationUpdates();

startStopButton.setText("Stop Tracking");

isTracking = true;

} else {

stopLocationUpdates();

startStopButton.setText("Start Tracking");

isTracking = false;

}

}

private void startLocationUpdates() {

LocationRequest locationRequest = LocationRequest.create();

locationRequest.setPriority(LocationRequest.PRIORITY\_HIGH\_ACCURACY);

locationRequest.setInterval(5000); // 5 seconds

locationCallback = new LocationCallback() {

@Override

public void onLocationResult(LocationResult locationResult) {

if (locationResult == null) {

return;

}

for (Location location : locationResult.getLocations()) {

// Update UI with location data

latitudeTextView.setText("Latitude: " + location.getLatitude());

longitudeTextView.setText("Longitude: " + location.getLongitude());

}

}

};

if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION) != PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {

// TODO: Consider calling

// ActivityCompat#requestPermissions

// here to request the missing permissions, and then overriding

// public void onRequestPermissionsResult(int requestCode, String[] permissions,

// int[] grantResults)

// to handle the case where the user grants the permission. See the documentation

// for ActivityCompat#requestPermissions for more details.

return;

}

fusedLocationClient.requestLocationUpdates(locationRequest, locationCallback, null);

}

private void stopLocationUpdates() {

fusedLocationClient.removeLocationUpdates(locationCallback);

}

@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults);

if (requestCode == REQUEST\_PERMISSION\_CODE) {

if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION\_GRANTED &&

grantResults[1] == PackageManager.PERMISSION\_GRANTED) {

startLocationUpdates();

} else {

Toast.makeText(this, "Permission denied. Cannot track location.", Toast.LENGTH\_SHORT).show();

}

}

}

}

**activity\_main.xml:**

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<Button

android:id="@+id/startStopButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerInParent="true"

android:text="Start Tracking"

android:onClick="startStopTracking"/>

<TextView

android:id="@+id/latitudeTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/startStopButton"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="16dp"

android:text="Latitude: "

android:textSize="16sp"/>

<TextView

android:id="@+id/longitudeTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/latitudeTextView"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="8dp"

android:text="Longitude: "

android:textSize="16sp"/>

</RelativeLayout>

**ActivityManifest.xml:**

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

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

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

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.Current\_location"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

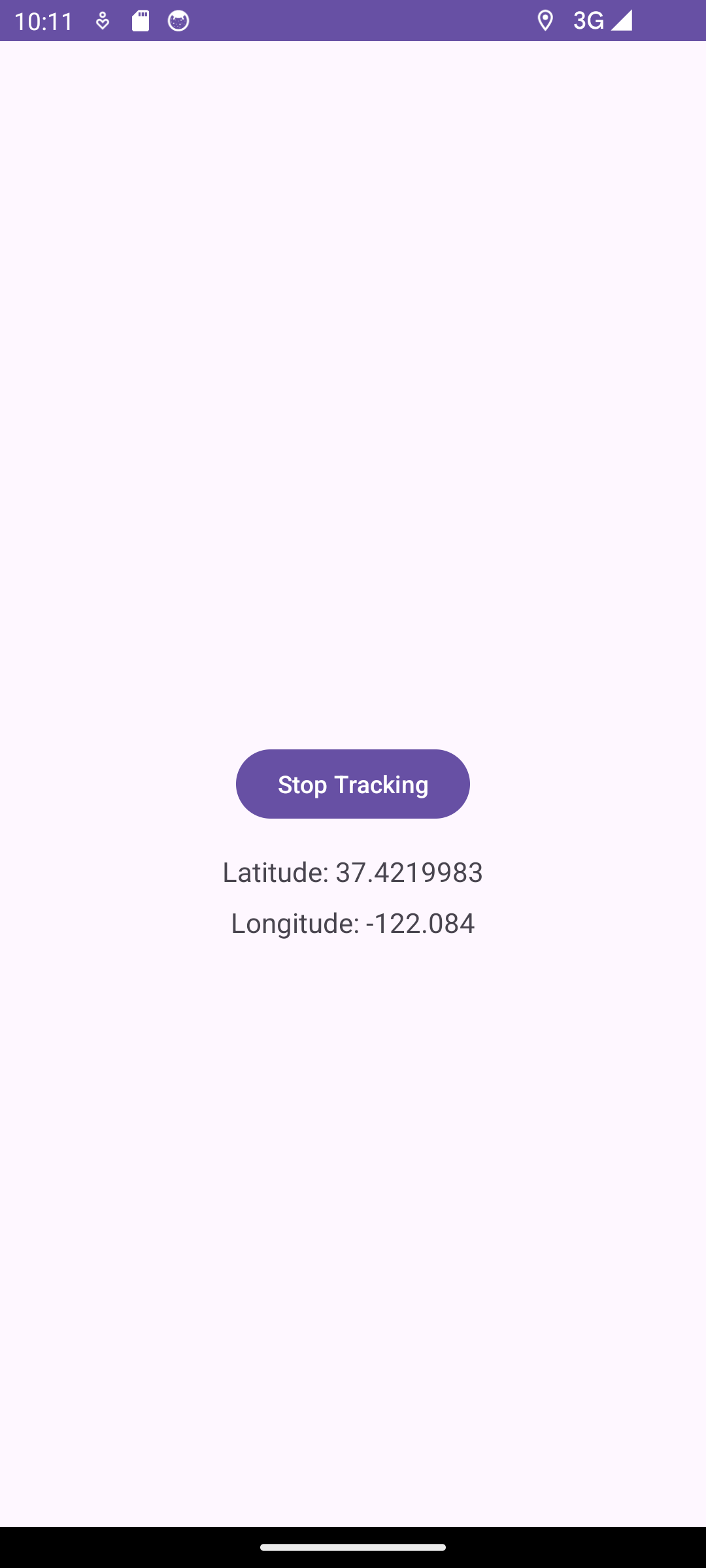
</intent-filter>

</activity>

</application>

</manifest>

**OUTPUT:**

****

**RESULT:**

The application that Google Maps on the screen is developed and tested using android studio