

III B.Tech. II-Sem (CSE)

T C
3+1* 3

(A0525156) ANDROID PROGRAMMING

UNIT-I

Introduction to Android, Types of Mobile Applications, Android Architecture(About DVM, Linux kernel, Java libraries & Native libraries, application frame work), Android Framework(Activity, Service, Broadcast Receiver, Content Provider), Android Studio Environment(how to Install, install in emulator, real device) Project Structure(R.Java, res folder, manifest.xml and .apk file), Android features, History, Layout UI groups(Leaner Layout, Relative Layout, Table Layout, Frame Layout, Grid Layout), Width and height properties(Match parent, Wrap content, Pixel, Density pixel and Scaled pixel).

UNIT-II

Activity, Activity lifecycle, Life cycle Methods, Intents, Intent Methods, layout file and its child elements and attributes, Basic UI Components(Text View, Button, Edit Text, Radio Button, Check Box), Advanced UI Components (AutoCompleteTextView, Spinner, ListView) Adapters (ArrayAdapter, Custom Adapter), Toast.

UNIT-III

WebView, WebView-HTML Communication, Fragment, Fragment Life Cycle. **Storage Methods:** shared preferences, SQLite Database (insert, read, update, delete). **Telephony:** send SMS, Call, Attaching File, and Send E-Mail.

UNIT-IV

Multimedia in Android: Media Player, Video View, Audio Recording, Video recording, Camera, Gallery. **Service:** Service, Service lifecycle methods.

UNIT-V

Built-in Services (Location service, Notification service, Sensor Service, WIFI Service, Bluetooth Service, Vibrator Service), Broadcast Receivers.

UNIT-VI

Content Provider, Dialog Boxes (Custom dialog, Alert dialog, date Picker, Time Picker, Progress dialog, dialog Fragment), Google Maps.

TEXT BOOKS:

1. Android Application Development (with Kitkat Support), Black Book by Pradeep Kothari
2. Android Programming: Pushing the Limits by Erik Hellman

REFERENCES:

1. Beginning Android 4 Application Development by Wei-Meng Lee
Android Application Development for Dummies by Michael Burton

ANDROID PROGRAMMING

RGM COLLEGE OF ENGINEERING AND TECHNOLOGY, NANDYAL-518501

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

LESSON PLAN

Name of the Faculty: **K. NARASIMHULU**

Academic Year: **2020-21**

Class & Semester : **III B. Tech II SEM-CSE-A&B&C**

Regulation : **R15**

Subject : **ANDROID PROGRAMMING**

Total Hours : **75**

| S.NO | UNIT | TOPIC | PERIODS |
|------|-----------|--|---------|
| 1 | I | Introduction to Android, Types of Mobile Applications | 2 |
| 2 | | Android Architecture(About DVM, Linux kernel, Java libraries & Native libraries, application frame work) | 2 |
| 3 | | Android Framework(Activity, Service, Broadcast Receiver, Content Provider) | 2 |
| 4 | | Android Studio Environment(how to Install, install in emulator, real device) | 2 |
| 5 | | Project Structure(R.Java, res folder, manifest.xml and .apk file) | 2 |
| 6 | | Android features, History | 2 |
| 7 | | LayoutUI groups(Leaner Layout, Relative Layout, Table Layout, Frame Layout, Grid Layout) | 2 |
| 8 | | Width and height properties (Match parent, Wrap content, Pixel, Density pixel and Scaled pixel). | 2 |
| 9 | II | Activity, Activity lifecycle, Life cycle Methods | 2 |
| 10 | | Intents, Intent Methods, layout file and its child elements and attributes. | 2 |
| 11 | | Basic UI Components(Text View, Button, Edit Text, Radio Button, Check Box) | 2 |
| 12 | | Advanced UI Components (AutoCompleteTextView, Spinner, ListView) | 6 |
| 13 | | Adapters (ArrayAdapter, Custom Adapter), Toast. | 1 |
| 14 | | Web View | 1 |

ANDROID PROGRAMMING

| | | | |
|----|------------|---|----------|
| 15 | III | Web View-HTML Communication | 2 |
| 16 | | Fragment, Fragment Life Cycle. | 2 |
| 17 | | Storage Methods: shared preferences | 2 |
| 18 | | SQLite Database | 2 |
| 19 | | Telephony: send SMS, Call, Attaching File, Send Mail | 2 |
| 20 | IV | Multimedia in Android: Media Player | 2 |
| 21 | | Video View | 2 |
| 22 | | Audio Recording | 2 |
| 23 | | Video Recording | 2 |
| 24 | | Camera | 2 |
| 25 | | Gallery | 2 |
| 26 | | Service: Service, Service lifecycle methods | 1 |
| 27 | V | Built-in Services : Location service | 2 |
| 28 | | Notification service, | 2 |
| 29 | | Sensor Service | 2 |
| 30 | | WIFI Service | 2 |
| 31 | | Bluetooth Service, Vibrator Service | 2 |
| 32 | | Broadcast Receivers | 2 |
| 33 | VI | Content Provider | 2 |
| 34 | | Dialog Boxes: Custom dialog | 1 |
| 35 | | Alert dialog | 1 |
| 36 | | date Picker | 1 |
| 37 | | Time Picker | 1 |
| 38 | | Progress dialog | 1 |
| 39 | | dialog Fragment | 1 |
| 40 | | Google Maps. | 2 |

Signature of Faculty

Signature of HOD

(A0525156) ANDROID PROGRAMMING**UNIT I****UNIT-I:**

Introduction to Android, Types of Mobile Applications, Android Architecture(About DVM, Linux kernel, Java libraries & Native libraries, application frame work), Android Framework(Activity, Service, Broadcast Receiver, Content Provider), Android Studio Environment(how to Install, install in emulator, real device) Project Structure(R.Java, res folder, manifest.xml and .apk file), Android features, History, Layout UI groups(Leaner Layout, Relative Layout, Table Layout, Frame Layout, Grid Layout), Width and height properties(Match parent, Wrap content, Pixel, Density pixel and Scaled pixel).

Introduction to Android

Android is mobile platform, which consists of three things. Such as operating system, Meddileware and KeyApplication.

Android is a platform for Mobile phone, Tv's, desktops, Wearable devices and Automobiles.

Android is a software package and Linux based operating system for mobile devices such as tablet computers and smartphones.

It is developed by Google and later the OHA (Open Handset Alliance). Java language is mainly used to write the android code even though other languages can be used.

The goal of android project is to create a successful real-world product that improves the mobile experience for end users.

There are many code names of android such as Lollipop, Kitkat, Jelly Bean, Ice cream Sandwich, Froyo, and Eclar.

Types of Mobile Applications

There are 3 **types of apps**:

1. Native apps

- **iOS** on Objective-C or Swift
- **Android** on Java
- **Windows Phone** on Net

2. Hybrid apps for all platforms all together with **Xamarin, React Native, Ionic, Angular Mobile Sencha Touch** etc.

3. Web apps as **responsive versions of website** to work on any mobile device.

Native apps:

Such apps are developed for a single mobile operating system exclusively, therefore they are “native” for a particular platform or device. App built for systems like iOS, Android, Windows phone, Symbian, Blackberry cannot be used on a platform other than their own. In other words, you won’t be able to use Android app on iPhone.

Main advantage of native apps is high performance and ensuring good user experience as developers use native device UI. Moreover, an access to wide range

of APIs that puts no limitation on app usage. Native applications are distinctly accessible from app stores of their kind and have the clear tendency to reach target customers.

Some icons to native apps are higher cost compared to other types of apps – due to the need of creating app duplicates for other platforms, separate support and maintenance for different types of apps resulting in bigger product price.

Hybrid apps:

They are built using multi-platform web technologies (for example HTML5, CSS and Java script). So-called hybrid apps are mainly website applications disguised in a native wrapper. Apps possess usual pros and cons of both native and web mobile applications.

Hybrid multi-platform apps are fast and relatively easy to develop – a clear advantage. Single code base for all platforms ensures low-cost maintenance and smooth updates. Widely used APIs, like gyroscope, accelerometer, and geolocation are available.

On the other hand, hybrid applications lack in performance, speed and overall optimization in comparison to native apps for instance. Also, there are certain design issues due to app inability to look in exactly same way on two or more platforms.

Web apps

These are software applications that behave in a fashion similar to native applications. Web apps use a browser to run and are usually written in HTML5, JavaScript or CSS. These apps redirect a user to URL and offer “install” option by simply creating a bookmark to their page.

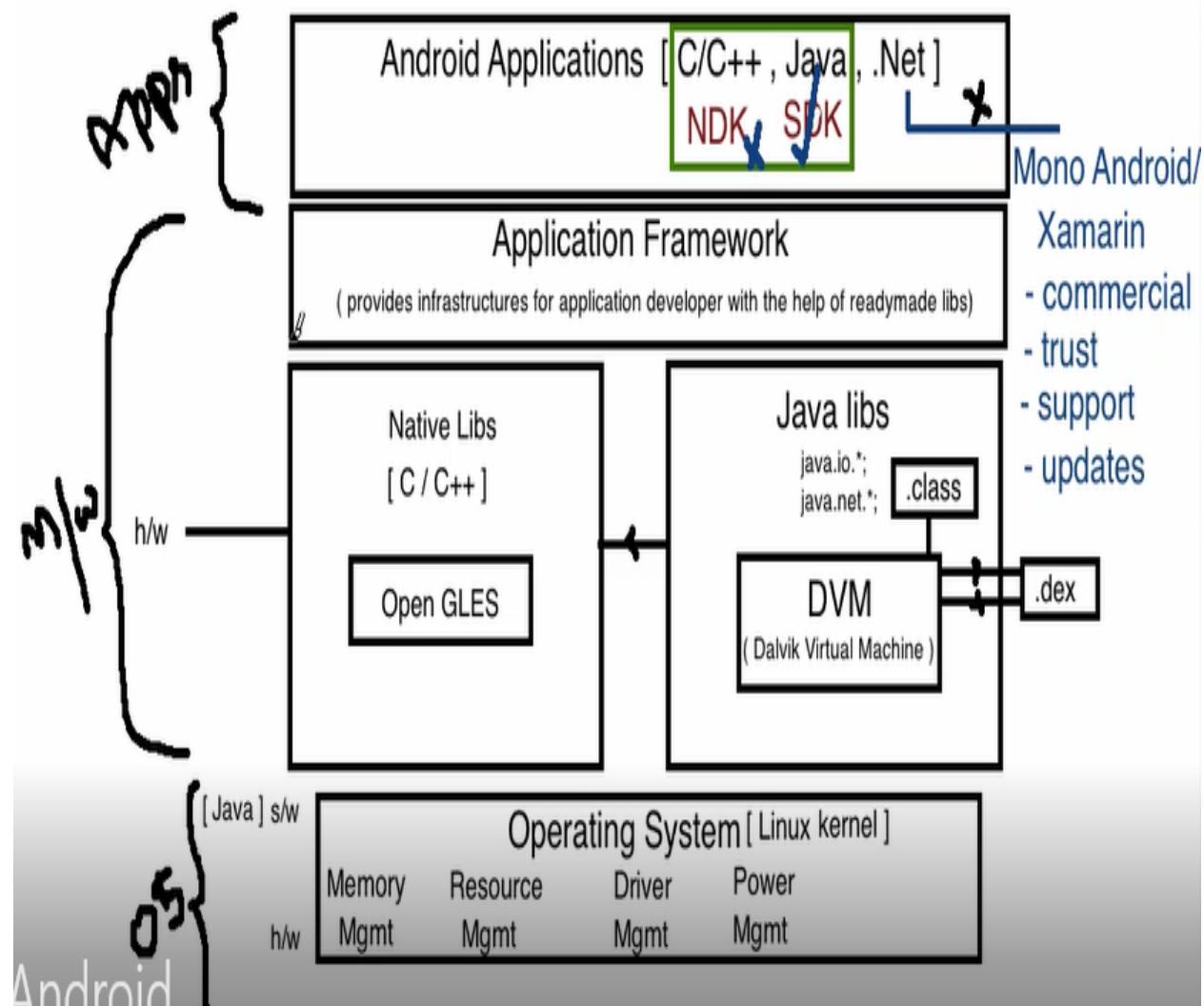
Web applications require minimum of device memory, as a rule. As all personal databases are saved on a server, users can get access from any device whenever there is internet connection. That is why the use of web apps with poor connection would result in bad user experience. The drawback is access to not that many APIs for developers, with exception of geolocation and few others.

Android Architecture

Android architecture or **Android software stack** is categorized into five parts:

1. Linux kernel(Operating System)
2. native libraries (middleware),
3. Android Runtime(middleware),
4. Application Framework(middleware),
5. Applications(Key Applications).

Let's see the android architecture first.



(OR)



1) Linux kernel

It is the heart of android architecture that exists at the root of android architecture. **Linux kernel** is responsible for device drivers, power management, memory management, device management and resource access.

2) Native Libraries

On the top of linux kernel, there are **Native libraries** such as WebKit, OpenGLES, FreeType, SQLite, Media, C runtime library (libc) etc.

The WebKit library is responsible for browser support SQLite is for database, FreeType for font support, Media for playing and recording audio and video formats.

3) Android Runtime

In android runtime, there are core libraries and DVM (Dalvik Virtual Machine) which is responsible to run android application. DVM is like JVM but it is optimized for mobile devices. It consumes less memory and provides fast performance.

4) Android Framework

On the top of Native libraries and android runtime, there is android framework. Android framework includes **Android API's** such as UI (User Interface), telephony, resources, locations, Content Providers (data) and package managers. It provides a lot of classes and interfaces for android application development.

5) Applications

On the top of android framework, there are applications. All applications such as home, contact, settings, games, browsers are using android framework that uses android runtime and libraries. Android runtime and native libraries are using linux kernal.

Android Core Building Blocks

An android **component** is simply a piece of code that has a well-defined life cycle e.g. Activity, Service, BroadCostReceiver and Content Provider.

The **core building blocks** or **fundamental components** of android are activities, views, intents, services, content providers, fragments and AndroidManifest.xml.



Activity

An activity is a class that represents a single screen. It is like a Frame in AWT.

View

A view is the UI element such as button, label, text field etc. Anything that you see is a **view**.

Intent

Intent is used to invoke components. It is mainly used to:

- Start the service
- Launch an activity
- Display a web page
- Display a list of contacts
- Broadcast a message
- Dial a phone call etc.

For example, you may write the following code to view the webpage.

```
Intent i=new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.rgmcet.edu.in"));
startActivity(intent);
```

Service

Service is a background processes that can run for a long time. There are two types of services local and remote. Local service is accessed from within the application whereas remote service is accessed remotely from other applications running on the same device.

Content Provider

Content Providers are used to share data between the applications.

Fragment

Fragments are like parts of activity. An activity can display one or more fragments on the screen at the same time.

AndroidManifest.xml

It contains information about activities, content providers, permissions etc. It is like the web.xml file in Java EE.

Android Virtual Device (AVD)

It is used to test the android application without the need for mobile or tablet etc. It can be created in different configurations to emulate different types of real devices.

Create your first instant app

How to build and run a very simple instant app using Android Studio. The app that you create has a simple structure similar to the structure described in Structure of a basic instant app.

Note: You must use Android Studio 3.0 to complete the procedures on this page. Also make sure that you have completed the setup instructions in Set up your development environment. This article demonstrates how to use the Android emulator to debug an instant app, so you must also complete the emulator set up steps under Set up your device or emulator.

To create a new instant app project in Android Studio 3.0, do the following:

1. Launch Android Studio and create a new project:
 - o If you have not opened a project yet, in the **Welcome to Android Studio** window, click **Start a new Android Studio project**.

- If you already have a project open, select **File > New Project**.
2. In the **Create Android Project** window, do the following:
- In the **Application name** box, enter "My First Instant App".
 - In the **Company domain** box, enter "example.com".
 - Leave the **Package name** as "com.example.myfirstinstantapp".
3. Click **Next**.
4. In the **Target Android Devices** window, do the following:
- Ensure that **Phone and Tablet** is selected.
 - In the **Minimum SDK** list, select **API 23: Android 6.0 (Marshmallow)**.
 - Under the **Minimum SDK** list, check **Include Android Instant app support**.

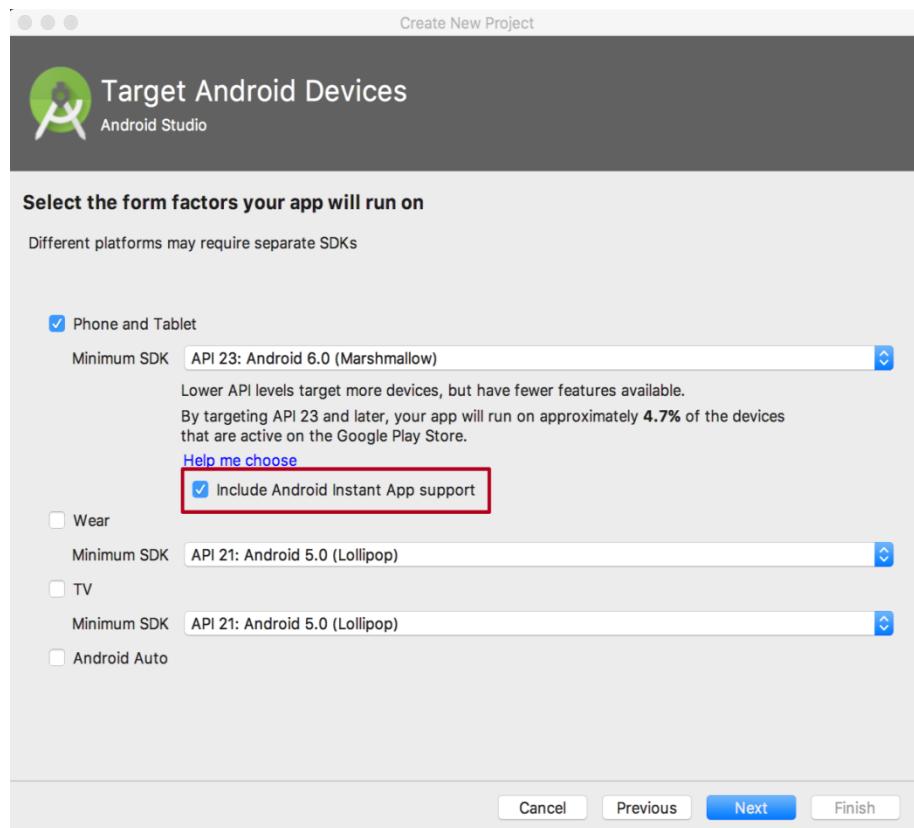


Figure : The **Target Android Devices** window.

5. Click **Next**.
6. In the **Customize Instant App Support** window, leave the default settings.
7. Click **Next**.

8. In the **Add an Activity to Phone and Tablet** window, select **Empty Activity**.
9. Click **Next**.
10. In the **Configure Activity** window, do the following:
 - o In the **Instant App URL Host** box, enter 'myfirstinstantapp.example.com'.
 - o In the **Instant App URL** route box, enter '/hello'.

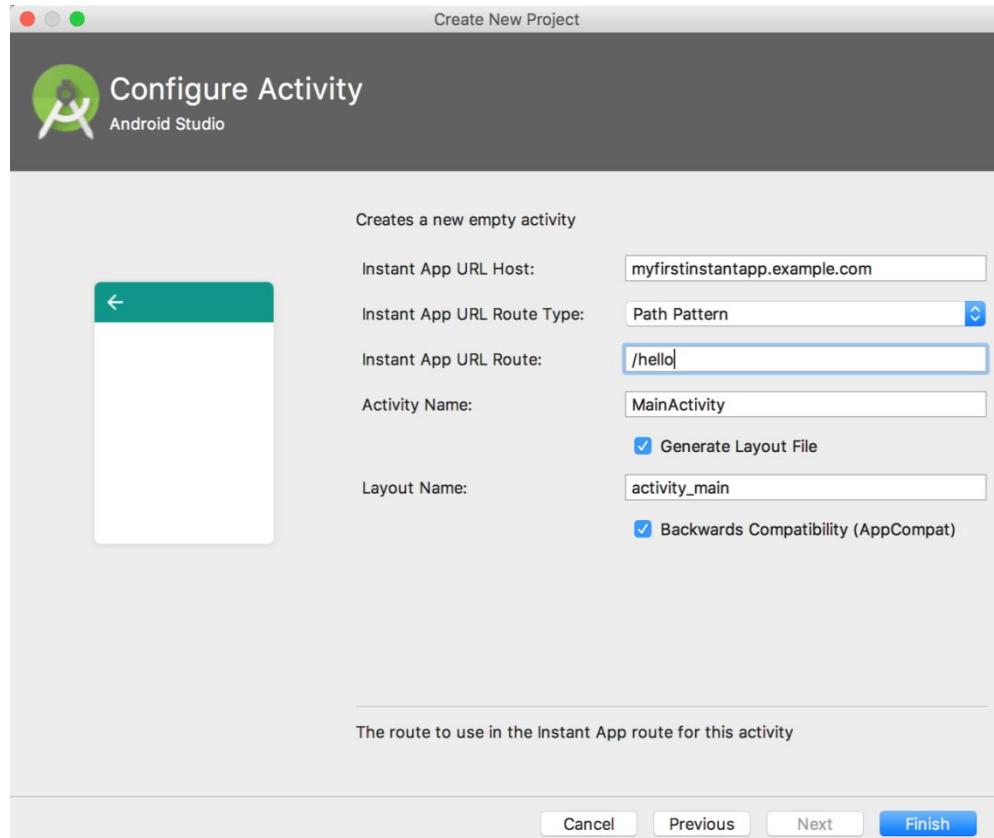


Figure: The **Configure Activity** window.

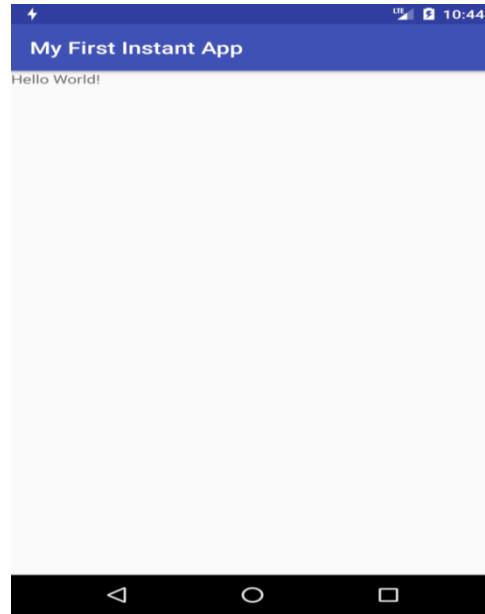
11. Click **Finish**.

After Android Studio has finished creating the project, you can run the instant app. Make sure that you have already created an emulator that can run instant apps, as described in Set up your device or emulator.

To run the project in Android Studio, do the following:

1. Click the the **instantapp** module in the **Project** window and then select **Run > Run 'instantapp'**.
2. In the **Select Deployment Target** window, select the emulator that you have set up for instant app development.

Android Studio builds and runs the app on the emulator as shown in figure



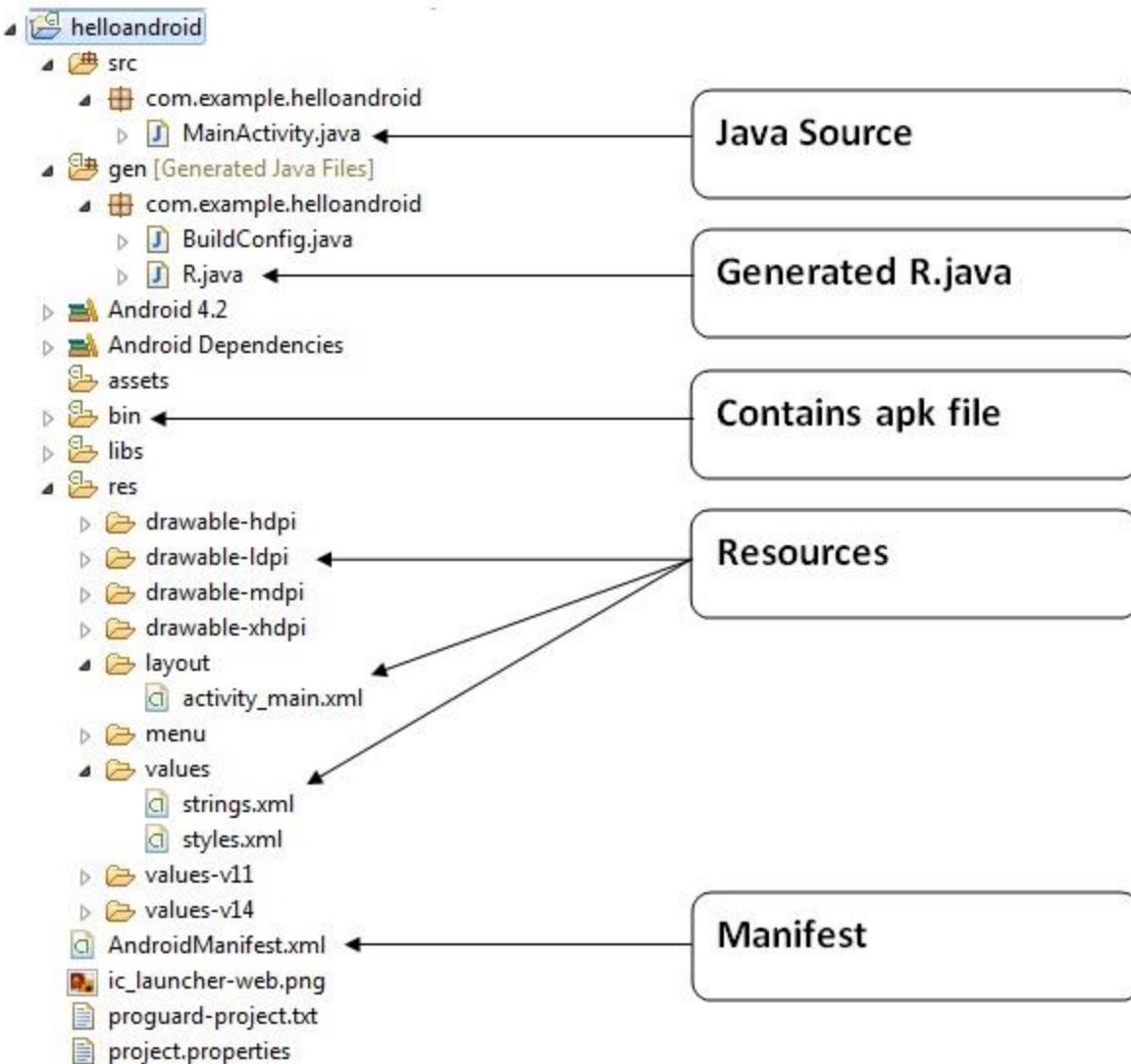
'My First Instant App' running.

Note: When you run your app locally on a device or emulator, the Android system displays the default placeholder icon for your app. Once you have deployed your Instant App APK to the Google Play Console, the appropriate icon shows up when the instant app launches.

Internal Details of Hello Android Example

Here, we are going to learn the internal details or working of hello android example.

Android application contains different components such as java source code, string resources, images, manifest file, apk file etc. Let's understand the project structure of android application.



Java Source Code

Let's see the java source file created by the Eclipse IDE:

File: MainActivity.java

1. **package** com.example.helloandroid;
2. **import** android.os.Bundle;
3. **import** android.app.Activity;
4. **import** android.view.Menu;
5. **import** android.widget.TextView;
6. **public class** MainActivity **extends** Activity { // (1)
7. @Override

```
8. protected void onCreate(Bundle savedInstanceState) { // (2)
9.     super.onCreate(savedInstanceState);
10.
11.    setContentView(R.layout.activity_main); // (3)
12. }
13. @Override
14. public boolean onCreateOptionsMenu(Menu menu) // (4)
15.     // Inflate the menu; this adds items to the action bar if it is present.
16.     getMenuInflater().inflate(R.menu.activity_main, menu);
17.     return true;
18. }
19. }
```

(1) Activity is a java class that creates and default window on the screen where we can place different components such as Button, EditText, TextView, Spinner etc. It is like the Frame of Java AWT. It provides life cycle methods for activity such as onCreate, onStop, OnResume etc.

(2) The **onCreate method** is called when Activity class is first created.

(3) The **setContentView(R.layout.activity_main)** gives information about our layout resource. Here, our layout resources are defined in activity_main.xml file.

File: activity_main.xml

```
1. <RelativeLayout>
2.     xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:tools="http://schemas.android.com/tools"
4.     android:layout_width="match_parent"
5.     android:layout_height="match_parent"
6.     tools:context=".MainActivity" >
7.
8.     <TextView>
9.         android:layout_width="wrap_content"
10.            android:layout_height="wrap_content"
11.            android:layout_centerHorizontal="true"
12.            android:layout_centerVertical="true"
13.            android:text="@string/hello_world" />
14.     </RelativeLayout>
```

As you can see, a textview is created by the framework automatically. But the message for this string is defined in the strings.xml file. The **@string/hello_world** provide information about the textview message. The value of the attribute hello_world is defined in the strings.xml file.

File: strings.xml

1. <?xml version="1.0" encoding="utf-8"?>
2. <resources>
3. <string name="app_name">helloandroid</string>
4. <string name="hello_world">Hello world!</string>
5. <string name="menu_settings">Settings</string>
6. </resources>

You can change the value of the hello_world attribute from this file.

Generated R.java file

It is the auto-generated file that contains IDs for all the resources of res directory. It is generated by aapt(Android Asset Packaging Tool). Whenever you create any component on activity_main, a corresponding ID is created in the R.java file which can be used in the Java Source file later.

R.java:

- R is termed as resource.
- R.java is an abstraction between the java file and different resources which are placed in 'res' folder.
- For every resource(file) in res folder it will create a static integer field in R.java, with the help of integer field we can access the resource into Activity(java file).
- Android Studio automatically will generate R.java.

APK File

An apk file is created by the framework automatically. If you want to run the android application on the mobile, transfer and install it.

Resources

It contains resource files including activity_main, strings, styles etc.

Manifest file

It contains information about package including components such as activities, services, content providers etc.

For more information about manifest file visit here: [AndroidManifest.xml file](#).

Android Features

- Android is Open Source.
- Application framework is providing infrastructures for application developer.
- DVM is optimized for mobile devices to work on low power, low memory and low RAM it is a customized JVM.
- SQLite Database is used to maintain structured data.
- Open GLES native library is used to display graphics.
- Android supports GPS and different media formats.
- Android Studio provides rich development environment.
- Android is a product from OHA [Open Handset Alliance] which is lead by Google.
- Android plays a key role in IOT [Internet of Things: Machine - Machine communication]

History of Android

The history and versions of android are interesting to know. The code names of android ranges from A to N currently, such as **Aestro, Blender, Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwitch, Jelly Bean, KitKat** and **Lollipop**. Let's understand the android history in a sequence.

- 1) Initially, **Andy Rubin** founded Android Incorporation in Palo Alto, California, United States in October, 2003.
- 2) In 17th August 2005, Google acquired android Incorporation. Since then, it is in the subsidiary of Google Incorporation.
- 3) The key employees of Android Incorporation are **Andy Rubin, Rich Miner, Chris White** and **Nick Sears**.
- 4) Originally intended for camera but shifted to smart phones later because of low market for camera only.
- 5) Android is the nick name of Andy Rubin given by co-workers because of his love to robots.
- 6) In 2007, Google announces the development of android OS.
- 7) In 2008, HTC launched the first android mobile.

Android Versions, Codename and API

Let's see the android versions, codenames and API Level provided by Google. The code names of android ranges from A to N currently, such as:

- | | |
|---------------------------------------|---------------------|
| • Aestro, | |
| • Blender, | |
| • Cupcake (April -27- 2009) | API-Leveal 3 |
| • Donut (Sept -15- 2009) | API-Leveal 4 |
| • Éclair (Oct -26- 2009) | API-Leveal 5,6,7 |
| • Froyo (May -20- 2010) | API-Leveal 8 |
| • Gingerbread (Dec -06- 2010) | API-Leveal 9,10 |
| • Honeycomb (Feb-22- 2011) | API-Leveal 11,12,13 |
| • Ice Cream Sandwich, (Oct -18- 2011) | API-Leveal 14,15 |
| • Jelly Bean (Jul -09- 2012) | API-Leveal 16,17,18 |
| • KitKat, (Oct -31- 2013) | API-Leveal 19,20 |
| • Lollipop (Nov-12- 2014) | API-Leveal 21,22 |
| • Marshmallow (Oct -05- 2015) | API-Leveal 23 |
| • Nougat (Aug-22-2016) | API-Level 24,25 |
| • Oreo(Aug-21-2017) | API-Level 26 |
| • Pie(Aug-6-2018) | API-Level 28 |
| • Android10(Sept-7-2019) | API-Level 29 |
| • Android11(Sept-8-2020) | API-Level 30 |

Dalvik Virtual Machine | DVM

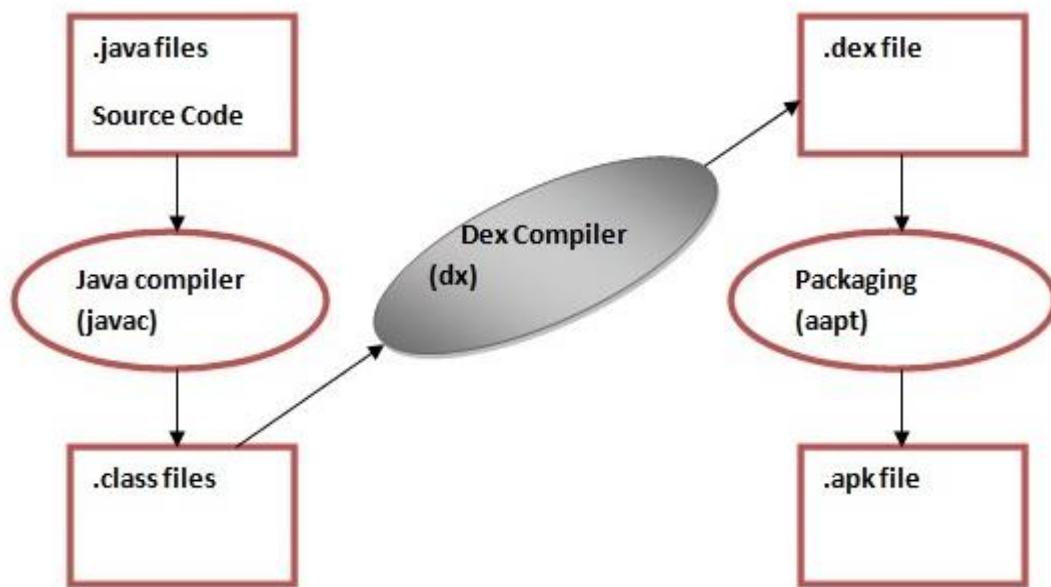
As we know the modern JVM is high performance and provides excellent memory management. But it needs to be optimized for low-powered handheld devices as well.

The **Dalvik Virtual Machine (DVM)** is an android virtual machine optimized for mobile devices. It optimizes the virtual machine for memory, battery life *and* performance.

Dalvik is a name of a town in Iceland. The Dalvik VM was written by Dan Bornstein.

The Dex compiler converts the class files into the .dex file that run on the Dalvik VM. Multiple class files are converted into one dex file.

Let's see the compiling and packaging process from the source file:



The **javac tool** compiles the java source file into the class file.

The **dx tool** takes all the class files of your application and generates a single .dex file. It is a platform-specific tool.

The **Android Assets Packaging Tool (aapt)** handles the packaging process.

Android Manifest.xml file in android

The **AndroidManifest.xml file** contains information of your package, including components of the application such as activities, services, broadcast receivers, content providers etc.

It performs some other tasks also:

It is **responsible to protect the application** to access any protected parts by providing the permissions.

- It also **declares the android api** that the application is going to use.
- It **lists the instrumentation classes**. The instrumentation class provides profiling and other informations. These informations are removed just before the application is published etc.

This is the required xml file for all the android application and located inside the root directory.

A simple AndroidManifest.xml file looks like this:

```
1. <manifest>
2.   xmlns:android="http://schemas.android.com/apk/res/android"
3.   package="com.NIT.hello"
4.   android:versionCode="1"
5.   android:versionName="1.0" >
6.   <uses-sdk>
7.     android:minSdkVersion="8"
8.     android:targetSdkVersion="15" />
9.   <application>
10.     android:icon="@drawable/ic_launcher"
11.     android:label="@string/app_name"
12.     android:theme="@style/AppTheme" >
13.     <activity>
14.       android:name=".MainActivity"
15.       android:label="@string/title_activity_main" >
16.     <intent-filter>
17.       <action android:name="android.intent.action.MAIN" />
18.       <category android:name="android.intent.category.LAUNCHER" />
19.     </intent-filter>
20.   </activity>
21. </application>
22. </manifest>
```

Elements of the AndroidManifest.xml file

The elements used in the above xml file are described below.

<manifest>

Manifest is the root element of the AndroidManifest.xml file. It has **package** attribute that describes the package name of the activity class.

<application>

application is the subelement of the manifest. It includes the namespace declaration. This element contains several sub elements that declares the application component such as activity etc.

The commonly used attributes are of this element are **icon**, **label**, **theme** etc.

Android:icon represents the icon for all the android application components.

android:label works as the default label for all the application components.

Android:theme represents a common theme for all the android activities.

<Activity>

Activity is the subelement of application and represents an activity that must be defined in the AndroidManifest.xml file. It has many attributes such as label, name, theme, launchMode etc.

android:label represents a label i.e. displayed on the screen.

android:name represents a name for the activity class. It is required attribute.

<Intent-filter>

Intent-filter is the sub-element of activity that describes the type of intent to which activity, service or broadcast receiver can respond to.

<action>

It adds an action for the intent-filter. The intent-filter must have at least one action element.

<category>

It adds a category name to an intent-filter.

Layout UI groups

UserInterface(UI) Groups:

-UI Group is used to specify how to arrange the UI components.

- **LinearLayout**
- **TableLayout**
- **GridLayout**
- **RelativeLayout**
- **FrameLayout**

-LinearLayout: LinearLayout is one of the UI group, which is used to present the data(item) in a vertical or Horizontal format one after another (Create UI elements step by step manner).

Syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width=" "
    android:layout_height=" "
    android:orientation="vertical| horizontal"
        // UI Components (here we can write UI components with in
LinearLayout. Example: TextView, EditText, Button...etc)
</LinearLayout >
```

-TableLayout: TableLayout is one of the UI group, which is used to present the data(item) in different size. Here all elements are placed in Tableformat.

Syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width=" "
    android:layout_height=" "
        // UI Components (here we can write UI components with in
LinearLayout. Example: TextView, EditText, Button...etc)
</TableLayout >
```

-GridLayout: GridLayout is one of the UI group, which is used to present the data(item) in columns size. Here all columns are same size follows.

Example: Mobile Apps

Syntax:

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

```
// UI Components (here we can write Ui components with in GridLayout.  
Example: TextView, EditText, Button...etc)  
</GridLayout >
```

-RelativeLayout: RelativeLayout is one of the UIgroup, which is used to present the data(item) in wherever you want. So, there is no specific rule in RelativeLayout. End of the Line also you can create UI elements.

Syntax:

<RelativeLayout

```
xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width=" "  
    android:layout_height=" "  
    // UI Components (here we can write Ui components with in GridLayout.  
    Example: TextView, EditText, Button...etc)  
</ RelativeLayout >
```

-FrameLayout: FrameLayout is one of the UIgroup, which is used to present the data(item) in Fixed position or other may be any position.

Syntax:

<FrameLayout

```
xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width=" "  
    android:layout_height=" "  
    // UI Components (here we can write Ui components with in GridLayout.  
    Example: TextView, EditText, Button...etc)  
</FrameLayout >
```

-- For every UI component and UI group we have to specify width and height, following are the possible parameters to specify width and height.

- **match_parent [≥ 2.2]** (match_parent attribute occupying the entire Screen width and height Position.)
- **fill_parent [< 2.2]**
- **wrap_content** (Wrap_content attribute occupying the screen upto the Content)
- **px [pixel]** (Pixel attribute can be used in the Size of UI Components have FixedSize)
- **dp[density pixel]** (Density Pixel attribute can be used based on the screen size
Increase or decrease the size of UI element)
- **sp[scaled pixel]** (Scaled pixel attribute can use only increasing the textSize)

UNIT-II

UNIT-II:

Activity, Activity lifecycle, Life cycle Methods, Intents, Intent Methods, layout file and its child elements and attributes, Basic UI Components(Text View, Button, Edit Text, Radio Button, Check Box), Advanced UI Components (AutoCompleteTextView, Spinner, ListView) Adapters (ArrayAdapter, Custom Adapter), Toast

Activity:

Q: What is an Activity?

Ans: Activity is a single screen application with UI components, user can interact the device through activity.

-Activity is a Java code that supports a screen or UI. In other words, building block of the user interface is the activity.

-Activity class is a pre-defined class in Android and every application which has UI must inherit it to create window.

Steps to create an Activity:

- 1) Create a class as a child of android.app.Activity [why?]

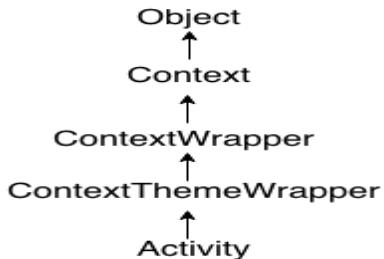
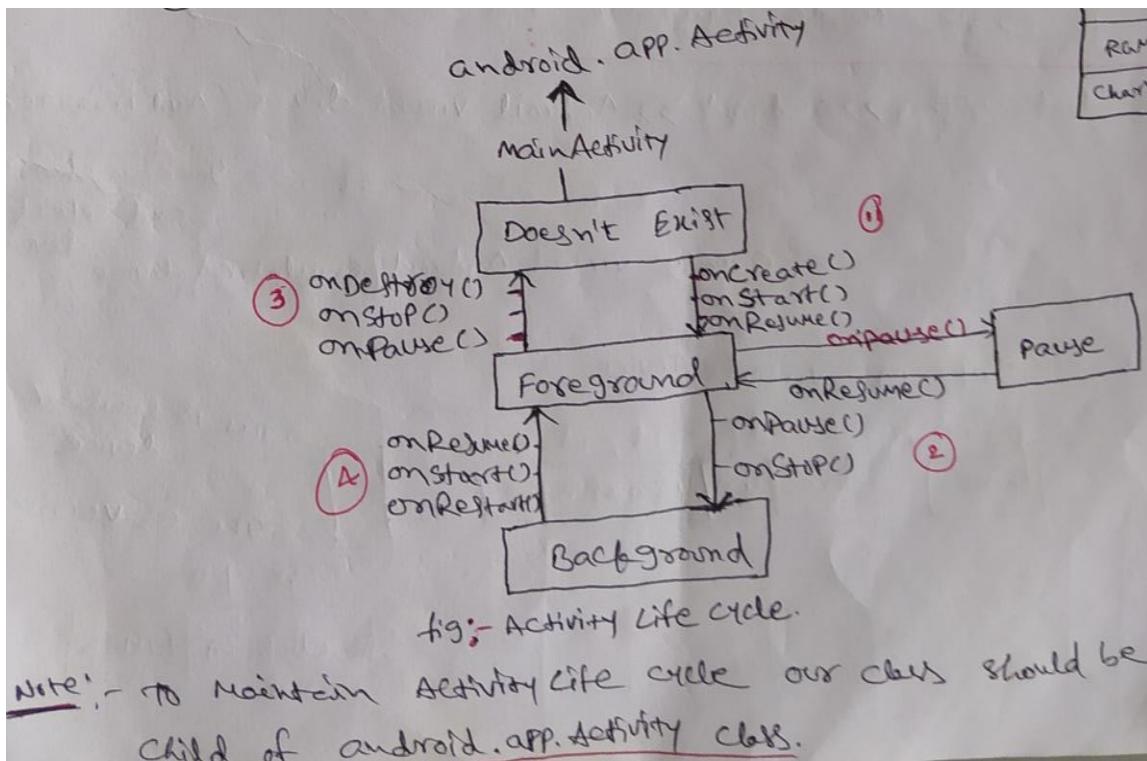
Activity Life Cycle:

Activity is having 4 states.

- 1) **Activity doesn't exist State**
- 2) **Foreground State**
- 3) **Pause State**
- 4) **Background State**

-Following are the major methods in Activity class.

- 1) **onCreate()**
- 2) **onStart()**
- 3) **onResume()**
- 4) **onPause()**
- 5) **onStop()**
- 6) **onRestart()**
- 7) **onDestroy()**

Android Activity Lifecycle:

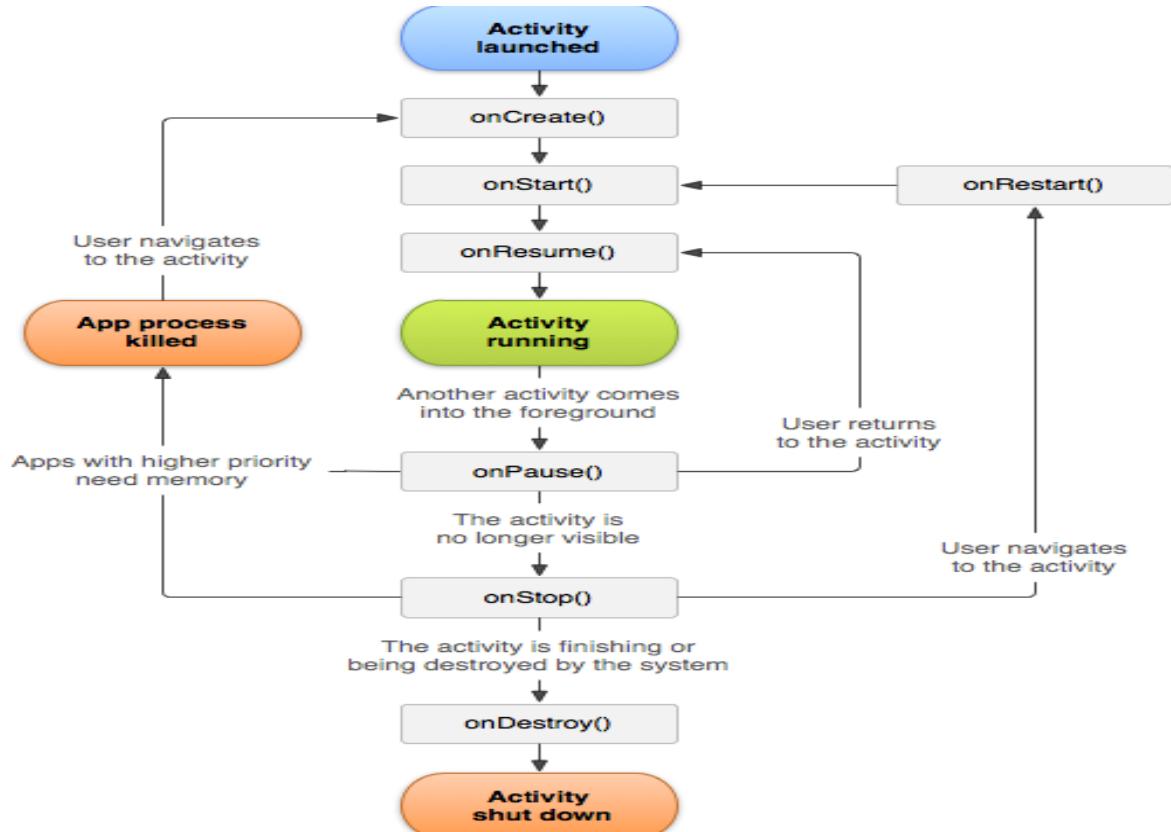
Android Activity Lifecycle is controlled by 7 methods of **android.app.Activity** class. The android Activity is the subclass of `ContextThemeWrapper` class. An activity is the single screen in android. It is like window or frame of Java.

By the help of activity, you can place all your UI components or widgets in a single screen. The 7 lifecycle method of Activity describes how activity will behave at different states.

Android Activity Lifecycle methods:

Let's see the 7 lifecycle methods of android activity.

| Method | Description |
|------------------|--|
| onCreate | called when activity is first created. |
| onStart | called when activity is becoming visible to the user. |
| onResume | called when activity will start interacting with the user. |
| onPause | called when activity is not visible to the user. |
| onStop | called when activity is no longer visible to the user. |
| onRestart | called after your activity is stopped, prior to start. |
| onDestroy | called before the activity is destroyed. |



To maintain the Activity Life cycle, our Activity class should be subtype of android.app.Activity.

Same like main() method in C/C++/Java in Android Activity onCreate() method will be invoke first so provide the implementation/override for onCreate() method.

- Bundle class is used to get the state of an Activity.
- We designed the UI in XML, we have to set the XML to java , use the following method in java (inside onCreate() method) to set the xml file.

setContentView(R.layout.file_name);

activity_main.xml:

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Welcome to RGMCET"
        android:textSize="40sp"
        android:textColor="#F000" />
</LinearLayout>
```

MainActivity.java

```
package cubexsoft.helloworld;
import android.os.Bundle;
import android.support.annotation.Nullable;
public class MainActivity extends android.app.Activity
{
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

OUTPUT:



- Button is having a click event, we can configure the click event in 2 ways.
 - XML
 - Java

XML :

- use the following to configure the click event using XML.
`android:onClick="method_name"`.
 - if we click the button it will invoke the specified method in java, if the method is not available it will throw MethodNotFoundException.

eg : xml:

android:onClick="getText"

java:

```
public void getText(View v)
{
    .......;
}
```

- to get the UI component into java we have to configure an id for the UI component, use the following attribute to set the id.

android:id="@+id/id_name"

- use the following method in java to get the UI component from XML.

findViewById(R.id.id_name);

Configure the click event using Java:

- to configure the click event using java, get the UI component from XML to java.

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

- to configure the following listener in java to configure the click event.

```
b.setOnClickListener(new View.OnClickListener()
{
    public void onClick(View v) {
        // .....
    }
});
```

activity_main.xml

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Enter Text:"
        android:textSize="40sp"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/et1"/>

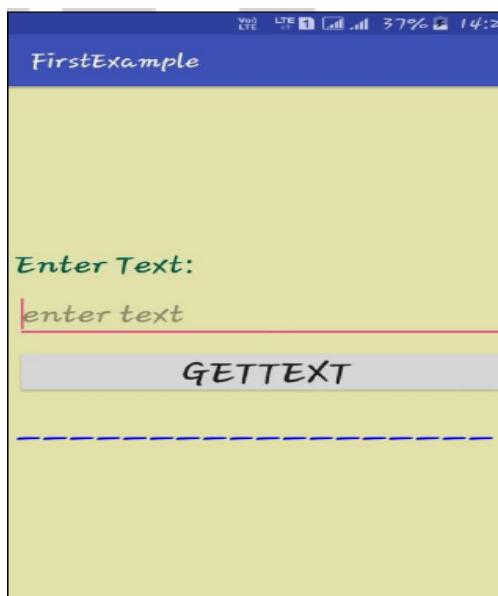
    <Button
        android:id="@+id/b1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="GetText"
        android:textSize="40sp"/>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="_____"
        android:textSize="40sp"
        android:id="@+id/tv1"/>
</LinearLayout>
```

MainActivity.java

```
package com.example.hi.welcomeApp;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
public class MainActivity extends android.app.Activity
{
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button b=(Button)findViewById(R.id.b1);
        b.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                EditText et1=(EditText)findViewById(R.id.et1);
                TextView tv1=(TextView)findViewById(R.id.tv1);
                tv1.setText(et1.getText());
            }
        });
    }
}
```

OUTPUT:



Android Intent

Android Intent is the message that is passed between components such as activities, content providers, broadcast receivers, services etc.

- **Intents are used to provide a communication between Activity - Activity, Activity -Service, Activity - Broadcast Receiver.**

-It is generally used with startActivity() method to invoke activity, broadcast receivers etc.

-The **dictionary meaning** of intent is intention or purpose. So, it can be described as the intention to do action.

-The LabeledIntent is the subclass of android.content.Intent class.

-Android intents are mainly used to:

- o Start the service
- o Launch an activity
- o Display a web page
- o Display a list of contacts
- o Broadcast a message
- o Dial a phone call etc.

- **Intents are used to provide a communication between Activity - Activity, Activity - Service, Activity - Broadcast Receiver.**

- **With respect to Activity there are 2 types of intents.**

- **Implicit Intents**
- **Explicit Intents**

Implicit Intents:

- Implicit Intents is used to call builtin Activities.
e.g. : camera , browser , settings, messages..

syntax:

```
Intent i=new Intent();
i.setAction(Intent.ACTION_NAME);
startActivity(i);
```

Explicit Intents:

- Explicit Intents is used to call user-defined Activities.

Syntax:

```
Intent i=new Intent(context, Activity_name.class);
startActivity(i);
```

-by using explicit intents we can invoke 3rd party applications(Eg:- WhatsApp, Gmail etc..) activities from our application.

Syntax:

```
Intent i=getPackageManager().getLaunchIntentForPackage("package_name");
startActivity(i);
```

-following are the major methods in Intent class.

**-setAction()
-setData()
-setComponent()**

**-setType()
-putExtra()
-getExtra()**

- every user defined Activity has to be configure in "**Manifest.xml**" with the following tag inside <application> tag.

```
<activity android:name="package_name.class_name"/>
```

- Which activity you want to display on the initial screen in your application configure the following tag to that Activity.

```
<activity android:name=".MainActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<activity android:name=".welcomeActivity"/>
```

activityMain.xml File:

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

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Number"
        android:id="@+id/et1"
        android:inputType="phone"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="DIAL"
        android:onClick="dial"
        android:layout_gravity="center"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="DIAL1"
        android:onClick="dial1"
        android:layout_gravity="center"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="NEXT"
        android:onClick="next"
        android:layout_gravity="center"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="WHATSAPP"
        android:onClick="whatsapp"
        android:layout_gravity="center"/>

</LinearLayout>
```

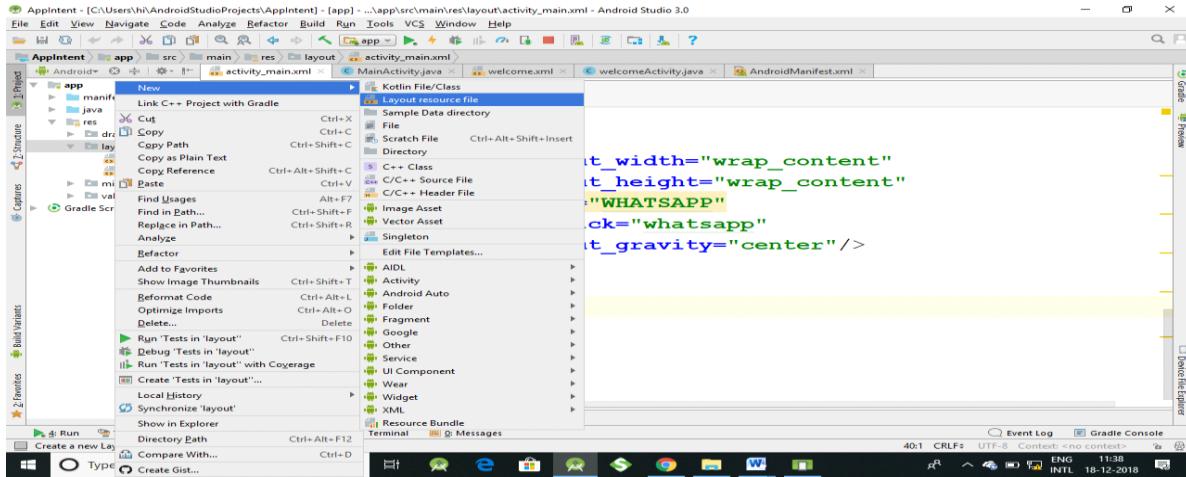
MainActivity.Java File:

```
package com.example.hi.appintent;
import android.content.ComponentName;
import android.content.Intent;
import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void dial(View v){
        Intent i=new Intent();
        i.setAction(Intent.ACTION_DIAL);
        EditText et1=(EditText)findViewById(R.id.et1);
        i.setData(Uri.parse("tel:"+et1.getText().toString()));
        startActivity(i);
    }
    public void dial1(View v){
        Intent i=new Intent();
        i.setAction(Intent.ACTION_GET_CONTENT);
        i.setType("Image/*");
        startActivity(i);
    }
    public void next(View v){
        Intent i=new Intent();
        i.setComponent(new ComponentName(this,welcomeActivity.class));
        startActivity(i);
    }
    public void whatsapp(View v){
        Intent i=getPackageManager().getLaunchIntentForPackage("com.whatsapp");
        startActivity(i);
    }
}
```

Creating a New Layout:

Goto **res folder >> layoutfolder**. Now right click layoutfolder >> **new >> layout resources file** and provide **name(welcome)**



Welcome.xml :

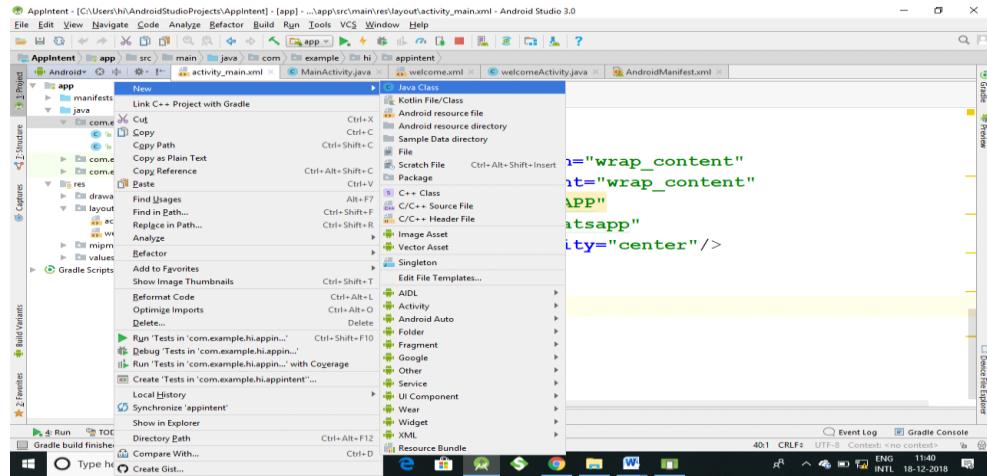
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="WELCOME TO CSE LAB2"
        android:textSize="30sp"
        android:textColor="#FF0000"/>

</LinearLayout>
```

Creating a New Java File:

-Goto **Java folder >> package folder >> Now right click packagefolder >> new >> java class** and provide **name(welcomeActivity)**.



WelcomeActivity.java

```
package com.example.hi.appintent;
import android.app.Activity;
import android.os.Bundle;
import android.support.annotation.Nullable;
```

```
public class welcomeActivity extends Activity {
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.welcome);
    }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hi.appintent">

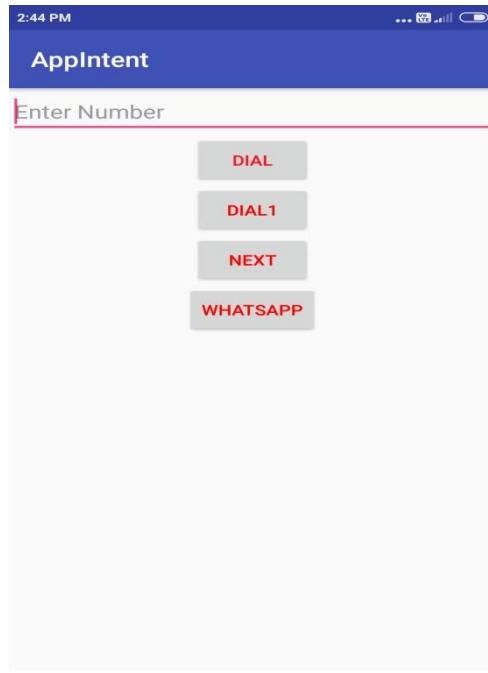
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
```

```
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

OUTPUT:



Basic User Interface Elements:

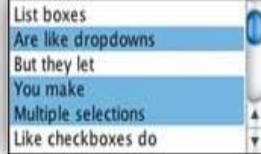
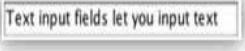
When designing your interface, try to be consistent and predictable in your choice of interface elements. Whether they are aware of it or not, users have become familiar with elements acting in a certain way, so choosing to adopt those elements when appropriate will help with task completion, efficiency, and satisfaction.

Interface elements include but are not limited to:

- **Input Controls:** checkboxes, radio buttons, dropdown lists, list boxes, buttons, toggles, text fields, date field.

Input Controls

| Element | Description | Examples |
|-----------------------|--|---|
| Checkboxes | Checkboxes allow the user to select one or more options from a set. It is usually best to present checkboxes in a vertical list. More than one column is acceptable as well if the list is long enough that it might require scrolling or if comparison of terms might be necessary. | A screenshot of a user interface showing two checkboxes. The first checkbox is labeled "NonFederal (99)" and the second is labeled "Federal (57)". Both checkboxes are currently unchecked. |
| Radio buttons | Radio buttons are used to allow users to select one item at a time. | A screenshot of a user interface showing two radio buttons. One radio button is labeled "Yes" and the other is labeled "No". The "Yes" button is currently selected. |
| Dropdown lists | Dropdown lists allow users to select one item at a time, similarly to radio buttons, but are more compact allowing you to save space. Consider adding text to the field, such as 'Select one' to help the user recognize the necessary action. | A screenshot of a user interface showing a search bar with the placeholder text "Find your state or...". To the right of the search bar is a small "Go" button. |

| Element | Description | Examples |
|------------------------|---|---|
| List boxes | List boxes, like checkboxes, allow users to select a multiple items at a time, but are more compact and can support a longer list of options if needed. |  |
| Buttons | A button indicates an action upon touch and is typically labeled using text, an icon, or both. |  |
| Dropdown Button | The dropdown button consists of a button that when clicked displays a drop-down list of mutually exclusive items. |  |
| Text fields | Text fields allow users to enter text. It can allow either a single line or multiple lines of text. |  |
| <u>EditText</u> | EditText is a predefined subclass of TextView that includes rich editing capabilities. | |

Advanced UI Components

AutoCompleteTextView :

- ACTV is one of the UI component in Android which is used to provide an auto completion support to the user it is a subtype of EditText component.

xml:

```
<AutoCompleteTextView  
    android:id="@+id/actv"  
    ..... />
```

Java:

```
AutoCompleteTextView actv=(AutoCompleteTextView)findViewById(R.id.actv);
```

-To provide auto completion support we have to configure the values, in android there are 2 ways to configure the values

- XML
- Java

XML:

```
res >> values >> strings.xml  
<string-array name="array_name">  
    <item>value1</item>  
    <item>value2</item>  
    <item>value3</item>  
    <item>value4</item>  
    .....  
</string-array>
```

-Use the following code in java to get these configured values.

```
String [] values=getResources().getStringArray(R.array.array_name);
```

Java:

```
String[] values=new String[]{"value1","value2","value3".....};
```

(or)

```
ArrayList<String> list=new ArrayList<String>();  
    list.add(value1);  
    list.add(value2);  
    list.add(value3);  
    .....
```

-To present the values we have to create an adapter, in android there are 3 types of adapters

- **Array Adapter**
- **Custom Adapter**
- **Cursor Adapter**

ArrayAdapter: By Using ArrayAdapter we can present string type of data on the screen.

Syntax:

```
ArrayAdapter<String> adapter=new ArrayAdapter<String>(context, xml_file, values);  
actv.setAdapter(adapter);  
actv.setThreshold(int);
```

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical">  
  
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Enter Country"  
    android:textSize="40sp"  
    android:textColor="#f00" />  
  
<AutoCompleteTextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/actv"/>  
  
</LinearLayout>
```

Strings.xml:

```
<resources>
    <string name="app_name">ACTCApp</string>
    <string-array name="country">
        <item>India</item>
        <item>Indonasia</item>
        <item>Ierland</item>
        <item>Srilanka</item>
        <item>SouthAfrica</item>
        <item>Pakistan</item>
        <item>Bangladesh</item>
        <item>Brezil</item>
    </string-array>
</resources>
```

MainActivity.java:

```
package com.example.hi.actcapp;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;

public class MainActivity extends AppCompatActivity {

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

        AutoCompleteTextView
            actv=(AutoCompleteTextView)findViewById(R.id.actv);

        String[] values=getResources().getStringArray(R.array.country);

        //String[] values=new String[]{"India", "Srilanka", "Arebia"};

        ArrayAdapter adapter=new
            ArrayAdapter(MainActivity.this, android.R.layout.
                simple_list_item_single_choice, values);
```

```
        actv.setAdapter(adapter);
        actv.setThreshold(1);
    }
}
```

OUTPUT:



Spinner:

- Spinner is one of the UI component in android, which is used to present the list of configured values in a drop-down menu.

Syntax:

XML:

```
<Spinner
    android:id="@+id/sp1"
    ..... />
```

Java:

```
Spinner sp1=(Spinner)findViewById(R.id.sp1);
```

-Same like ACTV for presenting the values we have to configure the values in XML/Java.

- if the values are configured in XML, use the following attribute to set the values.
android:entries="@array/array_name"

- if the values are configured in Java we have to create an adapter to set the values.

sp.setAdapter(adapter_obj);

- Spinner is having item selected event, to get this event configure the following listener.

- **sp.setOnItemSelectedListener(new OnItemSelectedListener(){
 onItemSelected(){};
 doNothingSelected() {};
});**

Toast:

- Toast is one of the notification method in Android which is used to display the text on the screen for few seconds.

syntax:

Toast.makeText(context, message, duration).show();

Activity_main.xml:

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Select Country"
        android:textSize="40sp" />

    <Spinner
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/sp1"
        android:entries="@array/countries">
    </Spinner>
```

```
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Select Gender"  
    android:textSize="40sp" />  
  
<Spinner  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/sp2">  
</Spinner>  
  
</LinearLayout>
```

Strings.xml

```
<resources>  
    <string name="app_name">SPINNERapp</string>  
    <string-array name="countries">  
        <item>-----Select-----</item>  
        <item>India</item>  
        <item>Ireland</item>  
        <item>Australia</item>  
        <item>China</item>  
        <item>Srilanka</item>  
        <item>Bangladesh</item>  
    </string-array>  
</resources>
```

MainActivity.java

```
package com.example.hi.spinnerapp;  
  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.AdapterViewAnimator;  
import android.widget.ArrayAdapter;  
import android.widget.Spinner;  
import android.widget.Toast;  
import java.util.ArrayList;
```

```
public class MainActivity extends AppCompatActivity {

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

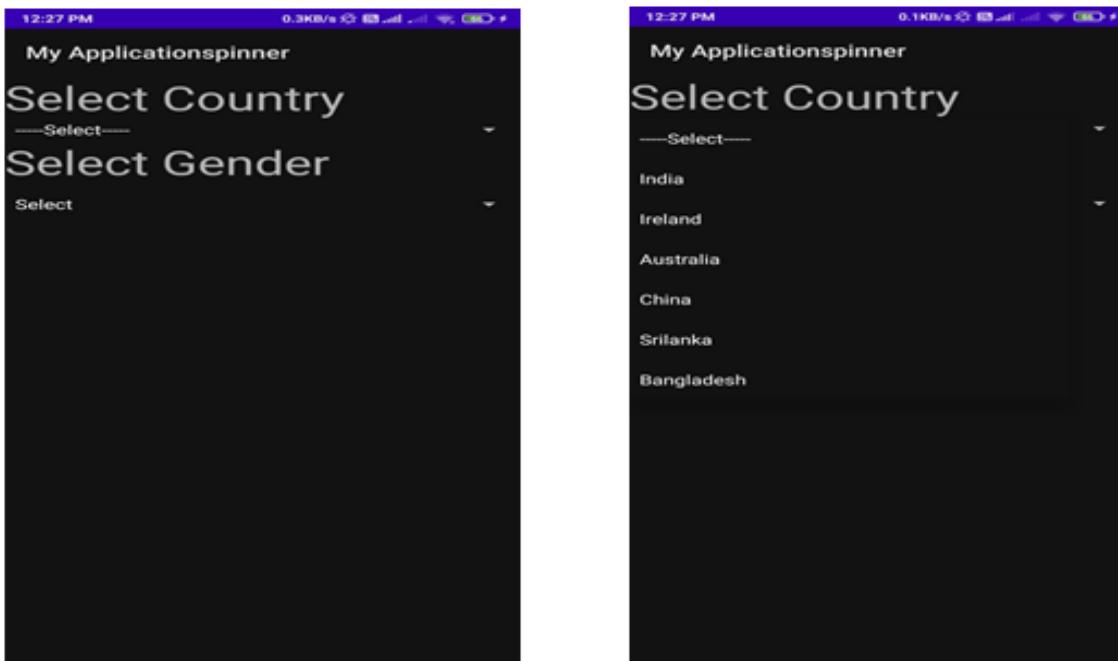
        final Spinner sp1 = (Spinner) findViewById(R.id.sp1);
        sp1.setOnItemSelectedListener(new
            AdapterView.OnItemSelectedListener() {
                @Override
                public void onItemSelected(AdapterView<?> parent, View view, int
                    position, long id) {
                    if (position > 0) {
                        Toast.makeText(MainActivity.this, sp1.getSelectedItem().toString(),
                            Toast.LENGTH_LONG).show();
                    }
                }

                @Override
                public void onNothingSelected(AdapterView<?> parent) {
                    }
                });
    }

    Spinner sp2 = (Spinner) findViewById(R.id.sp2);
    ArrayList<String> List = new ArrayList<>();
    List.add("Select");
    List.add("Male");
    List.add("Female");
    List.add("TransGender");
    ArrayAdapter<String> adapter = new
        ArrayAdapter<String>(MainActivity.this,
            R.layout.support_simple_spinner_dropdown_item, List);
    sp2.setAdapter(adapter);
}

}
```

OUTPUT:



-ListView :

- ListView is one of the UI component which is used present the list of configured values.

syntax:

Xml:

```
<ListView  
    android:id="@+id/lview1"  
...../>
```

Java:

```
ListView lview=(ListView)findViewById(R.id.lview1);
```

-same like ACTV & Spinner for presenting the values configure the values in XML / Java, if the values are configured in XML use the following attribute to set the values.

android:entries="@array/array_name"

- if the values are configured in Java, create an adapter to set the values.

ui.setAdapter(adapter);

Internal storage path:

```
String path="/storage/sdcard0/"  
      (OR)  
String path="/storage/emulated/0/"
```

External storage path:

```
String path="/storage/sdcard1/"  
      (OR)  
String path="/storage/extSdCard/"
```

```
ListView lview=(ListView)findViewById(R.id.lview);  
String path="/storage/emulated/0/";  
File f=new File(path);  
String[] files=f.list();  
ArrayAdapter<String> adapter=new  
    ArrayAdapter<String>(MainActivity.this,  
    R.layout.support_simple_spinner_dropdown_item, files);  
lview.setAdapter(adapter);
```

- if we are accessing the device storage information we have to add the following permission in Manifest.xml.

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

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical" >  
  
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="First practise of android"  
    android:textSize="40sp" />  
  
<ListView  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:id="@+id/lview">  
</ListView>  
  
</LinearLayout>
```

MainActivity.java:

```
package com.example.hi.listviewapp;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import java.io.File;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ListView lview=(ListView)findViewById(R.id.lview);
        String path="/storage/emulated/0/";
        File f=new File(path);
        String[] files=f.list();
        ArrayAdapter<String> adapter=new
            ArrayAdapter<String>(MainActivity.this,
            R.layout.support_simple_spinner_dropdown_item, files);
        lview.setAdapter(adapter);
    }
}
```

AndroidManifest.xml:

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

OUTPUT:



CustomAdapter:

- by using ArrayAdapter we can present only String type of data if you want to present your own UI on individual item use CustomAdapter.

Steps to create CustomAdapter :

- create a class with subtype of android.widget.BaseAdapter.
- it is an abstract class, having following abstract methods.
 - **getCount()**
 - **getItem()**
 - **getItemId()**
 - **getView()**

LayoutInflater:

- LayoutInflater class is used to convert the Xml UI into View object.

LayoutInflater inflater=LayoutInflater.from(context);

View v=inflater.inflate(R.layout.xml_file, view_group object);

- from Activity use the following code to set the custom adapter to the UI component.

ui.setAdapter(new CustomAdapterName());

activity_main.xml:

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

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

    </ListView>
</LinearLayout>
```

Indiview.xml:

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

    <ImageView
        android:layout_width="0dp"
        android:layout_weight="0.2"
        android:layout_height="100dp"
        android:id="@+id/lview1"
        android:src="@drawable/ic_launcher_background" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_weight="0.6"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tv1"
            android:text="File Name"
```

```
    android:textSize="20sp"
    android:textStyle="bold"
    android:textColor="#FF0000" />

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/tv2"
    android:text="File Size"
    android:textSize="20sp"
    android:textStyle="bold"
    android:textColor="#0000FF" />

</LinearLayout>

<Button
    android:layout_width="0dp"
    android:layout_weight="0.2"
    android:layout_height="wrap_content"
    android:text="Del"
    android:id="@+id/b1"/>

</LinearLayout>
```

MainActivity.java:

```
package com.example.hi.listview_customadapter;
import android.Manifest;
import android.content.pm.PackageManager;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;
import java.io.File;

public class MainActivity extends AppCompatActivity {
    ListView lview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        lview=(ListView)findViewById(R.id.lview);
```

```

int status= ContextCompat.checkSelfPermission
    (this, Manifest.permission.WRITE_EXTERNAL_STORAGE);
if(status== PackageManager.PERMISSION_GRANTED){
    readFiles();
} else{
    ActivityCompat.requestPermissions(this, new
        String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, 111);
}

}
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if(grantResults[0]==PackageManager.PERMISSION_GRANTED);
    readFiles();
}

public void readFiles( )
{
    String path=
        "/storage/emulated/0/WhatsApp/Media/WhatsApp Images/";
    File f=new File(path);
    File[] files=f.listFiles();
    lview.setAdapter(new MyAdapter(this, files));
}

}

```

MyAdapater.java:

```

package com.example.hi.listview_customadapter;
import android.net.Uri;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import java.io.File;
/***
 * Created by hi on 25-05-2021.
 */

```

```
public class MyAdapter extends BaseAdapter {
    MainActivity mActivity;
    File[] files;
    MyAdapter(MainActivity mActivity, File[] files){
        this.mActivity=mActivity;
        this.files=files;
    }
    @Override
    public int getCount() {

        return files.length;
    }

    @Override
    public Object getItem(int i) {
        return null;
    }

    @Override
    public long getItemId(int i) {
        return 0;
    }

    @Override
    public View getView(final int i, View view, ViewGroup viewGroup) {
        LayoutInflator inflater=LayoutInflater.from(mActivity);
        View v=inflater.inflate(R.layout.indiview,null);
        ImageView iview=v.findViewById(R.id.lview1);
        TextView tv1=v.findViewById(R.id.tv1);
        TextView tv2=v.findViewById(R.id.tv2);
        Button b1=v.findViewById(R.id.b1);
        Uri u=Uri.parse(files[i].getAbsolutePath());
        iview.setImageURI(u);
        tv1.setText(files[i].getName());
        tv2.setText(String.valueOf(files[i].length()));
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                files[i].delete();
            }
        });
    }
}
```

```
String path="/storage/emulated/0/WhatsApp/Media/WhatsApp Images/";
    File f=new File(path);
    files=f.listFiles();
    MyAdapter.this.notifyDataSetChanged();
}
});
return v;
}
}
```

AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.hi.listview_customadapter">
<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
</manifest>
```

OUTPUT:



Gallery Application By Using CustomAdapater:

-Gallery is one of the UI component in Android which is used to present the list of items in a horizontal format.

syntax:

Xml:

```
<Gallery  
    android:id="@+id/gall"  
..... />
```

Java:

```
Gallery gal=(Gallery)findViewById(R.id.gal);
```

activity_main.xml:

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

    <Gallery
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/gal" >
    </Gallery>
</LinearLayout>
```

indiview.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_margin="10dp">

    <VideoView
        android:layout_width="120dp"
        android:layout_height="120dp"
        android:id="@+id/vview1"
        android:padding="10dp"/>

    <CheckBox
        android:layout_width="120dp"
        android:layout_height="wrap_content"
        android:textSize="20sp"
        android:text="FileName"
        android:id="@+id/cb1" />

</LinearLayout>
```

MyAdapter.java:

```
package cubexsoft.gallerytest;
import android.net.Uri;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.CheckBox;
import android.widget.CompoundButton;
import android.widget.VideoView;
import java.io.File;

public class MyAdapter extends BaseAdapter {
    MainActivity activity;
    File[] files;

    MyAdapter(MainActivity activity, File[] files)
    {
        this.activity=activity;
        this.files=files;
    }

    @Override
    public int getCount()
    {
        return files.length;
    }

    @Override
    public Object getItem(int i)
    {
        return null;
    }

    @Override
    public long getItemId(int i)
    {
        return 0;
    }

    @Override
    public View getView(int i, View view, ViewGroup viewGroup) {
        LayoutInflater inflater=LayoutInflater.from(activity);
        View v=inflater.inflate(R.layout.indiview,null);
        final VideoView vview=v.findViewById(R.id.vview1);
        CheckBox cb1=v.findViewById(R.id.cb1);
        Uri u=Uri.fromFile(files[i]);
    }
}
```

```
vview.setVideoURI(u);
cb1.setText(files[i].getName());
cb1.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(CompoundButton compoundButton,
boolean b) {
        if(b){
            vview.start();
        }else{
            vview.stopPlayback();
        }
    }
});
return v;
}
```

MainActivity.java:

```
package com.example.hi.gallerytest;
import android.Manifest;
import android.content.pm.PackageManager;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Gallery;
import android.widget.GridView;
import java.io.File;

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

        int status= ContextCompat.checkSelfPermission(this,
            Manifest.permission.WRITE_EXTERNAL_STORAGE);
```

```

if(status== PackageManager.PERMISSION_GRANTED){
    readFiles( );
}else{
    ActivityCompat.requestPermissions(this,
        new String[]{Manifest.permission.READ_EXTERNAL_STORAGE},
        111);
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions,
    @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if(grantResults[0]==PackageManager.PERMISSION_GRANTED){
        readFiles( );
    }
}

public void readFiles( )
{
    String path=
        "/storage/emulated/0/WhatsApp/Media/WhatsApp Video/";
    File f=new File(path);
    File[] files=f.listFiles();
    gal.setAdapter(new MyAdapter(this,files));
}
}

```

AndroidManifest.xml:

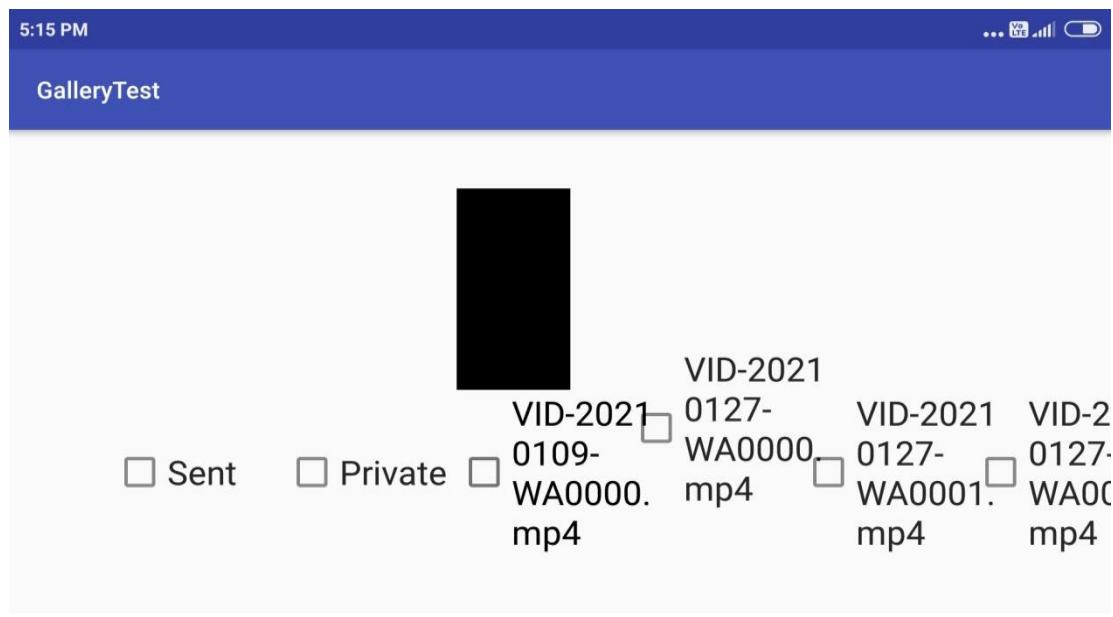
```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hi.gallerytest">
    <uses-permission android:name=
        "android.permission.READ_EXTERNAL_STORAGE"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">

```

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:



UNIT-III

UNIT-III:

WebView, WebView-HTML Communication, Fragment, Fragment Life Cycle.

Storage Methods: shared preferences, SQLite Database (insert, read, update, delete). **Telephony:** send SMS, Call, Attaching File, and Send E-Mail.

WebView:

WebView is one of the UI component in Android, which is used to display webpages in Android application.

Syntax:

xml:

```
<WebView  
    android:id="@+id/wview1"  
...../>
```

java:

```
WebView wview=(WebView)findViewById(R.id.wview1);
```

-By using web view we can perform the following operations:

- call browser
- integrate the browser in our application.
- display static .html files.
- we can provide a communication between HTML UI and Android Activity.
- wview.loadUrl("url_address") method is used to display specific webpage on WebView.

Permission:

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

- set the following method to web view component to display webpages on web view component.

```
wview.setWebViewClient(new WebViewClient());
```

- Following are the major methods in WebViewClient class:

- onPageStarted();
- shouldOverrideUrlLoading();
- onPageFinished();

- by default web view will not enable Javascript and zoom controls, to enable java script and zoom controls set the following methods to web view component.

```
wview.getSettings().setJavaScriptEnabled(true);  
wview.getSettings().setBuiltInZoomControls(true);
```

- display .html files:

Goto App folder >> new >> folder

- To display html files place the .html file in assets folder, use the following code to display .html files.

```
wview.loadUrl("file:///android_asset/file_name.html");
```

- Communication between HTML UI & Android Activity:

- By using JavaScript Interface we can provide the communication between HTML UI & Android Activity.

```
- wview.addJavaScriptInterface(class_object,"interface_name");
```

- by using the second parameter (interface_name) we can communicate with the specified class methods from JavaScript.

- Which method you are calling from Java Script, add the following annotation on top the method.

@JavaScriptInterface

activity_main.xml:

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.1"
        android:orientation="horizontal">

        <EditText
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.8"
            android:id="@+id/et1"
            android:hint="Enter URL:"/>
    
```

```
<ImageView  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.2"  
    android:src="@drawable/magnifier"  
    android:id="@+id/srch"  
    android:onClick="load"  
    android:layout_margin="5dp" />  
</LinearLayout>  
  
<WebView  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="0.8"  
    android:id="@+id/wview">  
</WebView>  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="0.1"  
    android:orientation="horizontal">  
  
<ImageView  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.25"  
    android:src="@drawable/fb"  
    android:id="@+id/fb"  
    android:onClick="load" />  
  
<ImageView  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.25"  
    android:src="@drawable/google"  
    android:id="@+id/google"  
    android:onClick="load" />  
<ImageView  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.25"  
    android:src="@drawable/youtube"  
    android:id="@+id/youtube"  
    android:onClick="load" />
```

```
<ImageView  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.25"  
    android:src="@drawable/html"  
    android:id="@+id/html"  
    android:onClick="load" />  
  
</LinearLayout>  
  
</LinearLayout>
```

MainActivity.java:

```
package com.example.hi.webviewapp;  
import android.graphics.Bitmap;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.webkit.JavascriptInterface;  
import android.webkit.WebChromeClient;  
import android.webkit.WebView;  
import android.webkit.WebViewClient;  
import android.widget.EditText;  
import android.widget.Toast;  
public class MainActivity extends AppCompatActivity {  
    WebView wview;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        wview=(WebView)findViewById(R.id.wview);  
        wview.setWebViewClient(new WebViewClient(){  
            @Override  
            public void onPageStarted(WebView view, String url, Bitmap favicon) {  
                super.onPageStarted(view, url, favicon);  
                Toast.makeText(getApplicationContext(),"PAGE LOADING STARTED....",  
                    Toast.LENGTH_LONG).show();  
            }  
        });  
    }  
}
```

```
@Override
public boolean shouldOverrideUrlLoading(WebView view, String url) {
    Toast.makeText(getApplicationContext(),url,Toast.LENGTH_LONG).show();
    return super.shouldOverrideUrlLoading(view, url);
}

@Override
public void onPageFinished(WebView view, String url) {
    super.onPageFinished(view, url);
    Toast.makeText(getApplicationContext(),"PAGE LOADING FINISHED",
        Toast.LENGTH_LONG).show();
}

wview.getSettings().setJavaScriptEnabled(true);
wview.getSettings().setBuiltInZoomControls(true);
wview.addJavascriptInterface(this,"myinterface");
}

@JavascriptInterface
public void displayMsg(String name,String pass){
    Toast.makeText(this, name+ "\n" +pass, Toast.LENGTH_LONG).show();
}

public void load(View v)
{
    switch(v.getId())
    {
        case R.id.srch:
            EditText et1=(EditText)findViewById(R.id.et1);
            wview.loadUrl(et1.getText().toString());
            break;

        case R.id.fb:
            wview.loadUrl("http://www.facebook.com");
            break;

        case R.id.google:
            wview.loadUrl("http://www.google.com");
            break;

        case R.id.youtube:
            wview.loadUrl("http://www.youtube.com");
            break;
    }
}
```

```
        case R.id.html:  
            wview.loadUrl("file:///android_asset/login.html");  
            break;  
    }  
}  
}
```

login.html:

```
<html>  
<head>  
    <script language="JavaScript">  
  
        function login(){  
  
            var name=document.getElementById("name").value;  
            var pass=document.getElementById("pass").value;  
  
            myinterface.displayMsg(name,pass);  
        }  
  
    </script>  
</head>
```

login.html:

```
<html>  
<head>  
    <script language="JavaScript">  
  
        function login(){  
            var name=document.getElementById("name").value;  
            var pass=document.getElementById("pass").value;  
            myinterface.displayMsg(name,pass);  
        }  
  
    </script>  
</head>  
    <body bgcolor="#054" text="white">  
        <center>  
            <table border="1">  
                <tr>  
                    <td colspan="2" align="center">  
                        <h3>Login Form</h3>  
                    </td>  
                </tr>  
                <tr>
```

```
<td>Enter Uname:</td>
<td> <input type="text" id="name"/>
</td>
</tr>
<tr>
<td>Enter Pass</td>
<td>
<input type="password" id="pass"/>
</td>
</tr>
<tr>
<td colspan="2" align="center">
<input type="button" value="Login" onclick="login()"/>
</td>
</tr>
</table>
</center>
</body>
</html>
```

AndroidManifest.xml:

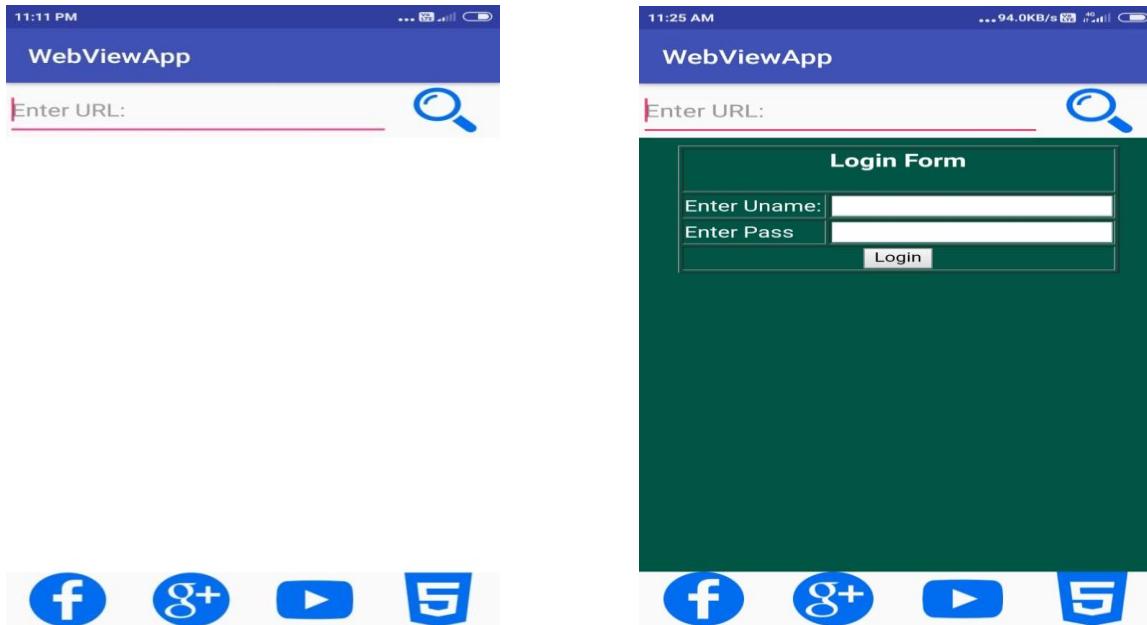
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hi.webviewapp">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

OUTPUT:



Fragments:

- Fragment is one of the UI component in Android (or) fragment is a subtype of Activity.

Syntax:

```
<FrameLayout  
    android:id="@+id/fram1"  
..... />
```

- In a single Activity we can create number of fragments, each fragment is having its own lifecycle and its own UI.

Steps to create a Fragment:

- create a class with subtype of android.app.Fragment.
- Same like Activity onCreate() method in fragments onCreateView() method will be invoke first so provide the implementation for onCreateView() method.

```
public View onCreateView(LayoutInflater inflater,  
                        ViewGroup vgroup, Bundle b){  
    View v=inflater.inflate(R.layout.xml_file, vgroup, false);  
    return v;  
}
```

- we will manage fragments from Activity, use the following to code manage fragments.

```
FragmentManager fManager=getFragmentManager();
FragmentTransaction tx=fManager.beginTransaction();
tx.add/replace/remove(frag_id, frag_class_obj);
tx.commit();
```

- Fragment concept is introduced from Android 3.0 so the minimum version to use fragments is Android 3.0.
- If the UI component is available on fragment XML, we can't configure the events using XML.
- use the View object in onCreateView() method to get the UI component from fragment XML.
- Android fragment lifecycle is affected by activity lifecycle because fragments are included in activity.
- Each fragment has its own life cycle method that is affected by activity life cycle because fragments are embedded in activity.
- The **FragmentManager** class is responsible to make interaction between fragment objects.

Android Fragment Lifecycle:

The lifecycle of android fragment is like the activity lifecycle. There are 12 lifecycle methods for fragment.

| No. | Method | Description |
|-----|--------------------|---|
| 1) | onAttach(Activity) | It is called only once when it is attached with activity. |
| 2) | onCreate(Bundle) | It is used to initialize the fragment. |

| | | |
|-----|---|--|
| 3) | onCreateView(LayoutInflater, ViewGroup, Bundle) | Creates and returns view hierarchy. |
| 4) | onActivityCreated(Bundle) | It is invoked after the completion of onCreate() method. |
| 5) | onViewStateRestored(Bundle) | It provides information to the fragment that all the saved state of fragment view hierarchy has been restored. |
| 6) | onStart() | Makes the fragment visible. |
| 7) | onResume() | Makes the fragment interactive. |
| 8) | onPause() | Is called when fragment is no longer interactive. |
| 9) | onStop() | Is called when fragment is no longer visible. |
| 10) | onDestroyView() | Allows the fragment to clean up resources. |
| 11) | onDestroy() | Allows the fragment to do final clean up of fragment state. |
| 12) | onDetach() | It is called immediately prior to the fragment no longer being associated with its activity. |

activity_main.xml:

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.1"
        android:text="RGMCET"
        android:textSize="30sp"
        android:gravity="center"
        android:textStyle="bold"
        android:textColor="#FFFFFF"
        android:background="#054" />

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.75"
        android:id="@+id/frag1" >
    </FrameLayout>

    <HorizontalScrollView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.15"
        android:orientation="horizontal" >

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="horizontal"
            android:background="#054" >
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:textColor="#FFFFFF"
    android:text="HOME"
    android:onClick="home"
    android:layout_margin="5dp"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:textColor="#FFFFFF"
    android:text="COURSES"
    android:onClick="courses"
    android:layout_margin="5dp"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:textColor="#FFFFFF"
    android:text="PLACEMENTS"
    android:onClick="placements"
    android:layout_margin="5dp"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:textColor="#FFFFFF"
    android:text="FACULTIES"
    android:onClick="faculties"
    android:layout_margin="5dp"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:textColor="#FFFFFF"
    android:text="RESULTS"
    android:onClick="results"
    android:layout_margin="5dp"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
```

```
        android:textColor="#FFFFFF"
        android:text="EVENTS"
        android:onClick="events"
        android:layout_margin="5dp"/>

    </LinearLayout>

</HorizontalScrollView>

</LinearLayout>
```

Home.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#098">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="WELCOME TO RGMCET"/>

</LinearLayout>
```

HomeFragment.java

```
package com.example.hi.fragmentapp;
import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/**
 * Created by hi on 27-05-2021.
 */
```

```
public class HomeFragment extends Fragment {  
    @Nullable  
    @Override  
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup  
    container, Bundle savedInstanceState) {  
        View v=inflater.inflate(R.layout.home, container, false);  
        return v;  
    }  
}
```

courses.xml:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:orientation="vertical"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:background="#FF0000">  
  
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:textSize="30sp"  
    android:text="courses"  
    android:gravity="center"/>  
  
<Spinner  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/sp1"  
    android:entries="@array/courses"  
    android:layout_gravity="center">  
</Spinner>  
  
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="GET INFO"
```

```
    android:textSize="20sp"
    android:id="@+id/b1"
    android:textStyle="bold"
    android:layout_gravity="center" />

</LinearLayout>
```

Strings.xml:

```
<resources>
    <string name="app_name">FragmentApp</string>
    <string-array name="courses">
        <item>-----Select-----</item>
        <item>CSE</item>
        <item>ECE</item>
        <item>EEE</item>
        <item>MCA</item>
        <item>IT</item>
        <item>CIVIL</item>
        <item>MECH</item>
        <item>MBA</item>
    </string-array>
</resources>
```

courseFragment:

```
package com.example.hi.fragmentapp;
import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.Spinner;
import android.widget.Toast;

/**
 * Created by hi on 27-05-2021.
 */
```

```
public class CoursesFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup
    container, Bundle savedInstanceState) {
        View v=inflater.inflate(R.layout.courses,container,false);
        final Spinner sp1=(Spinner)v.findViewById(R.id.sp1);
        Button b1=(Button)v.findViewById(R.id.b1);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Toast.makeText(getActivity(),sp1.getSelectedItem().toString(),Toast.LENGTH_LONG)
                .show();
            }
        });
        return v;
    }
}
```

placements.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#941">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="placements"
        android:textSize="30sp"
        android:gravity="center" />

</LinearLayout>
```

placementFragment:

```
package com.example.hi.fragmentapp;
import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/*
 * Created by hi on 27-05-2021.
 */

public class PlacementFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {
        View v=inflater.inflate(R.layout.placements, container, false);
        return v;
    }
}
```

faculties.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#f00">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="faculties"
        android:gravity="center" />

</LinearLayout>
```

FacultiesFragment:

```
package com.example.hi.fragmentapp;
import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/*
 * Created by hi on 27-05-2021.
 */

public class FacultyFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {
        View v=inflater.inflate(R.layout.faculties, container, false);
        return v;
    }
}
```

events.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#054">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="events"
        android:gravity="center" />
</LinearLayout>
```

Eventsfragment.java

```
package com.example.hi.fragmentapp;
import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/*
 * Created by hi on 27-05-2021.
 */

public class EventsFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup
            container, Bundle savedInstanceState) {
        View v=inflater.inflate(R.layout.events, container, false);
        return v;
    }
}
```

results.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#309">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="results"
        android:gravity="center"/>

</LinearLayout>
```

resultsFragment.java

```
package com.example.hi.fragmentapp;
import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/*
 * Created by hi on 27-05-2021.
 */
public class ResultsFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {

        View v=inflater.inflate(R.layout.results,container,false);
        return v;
    }
}
```

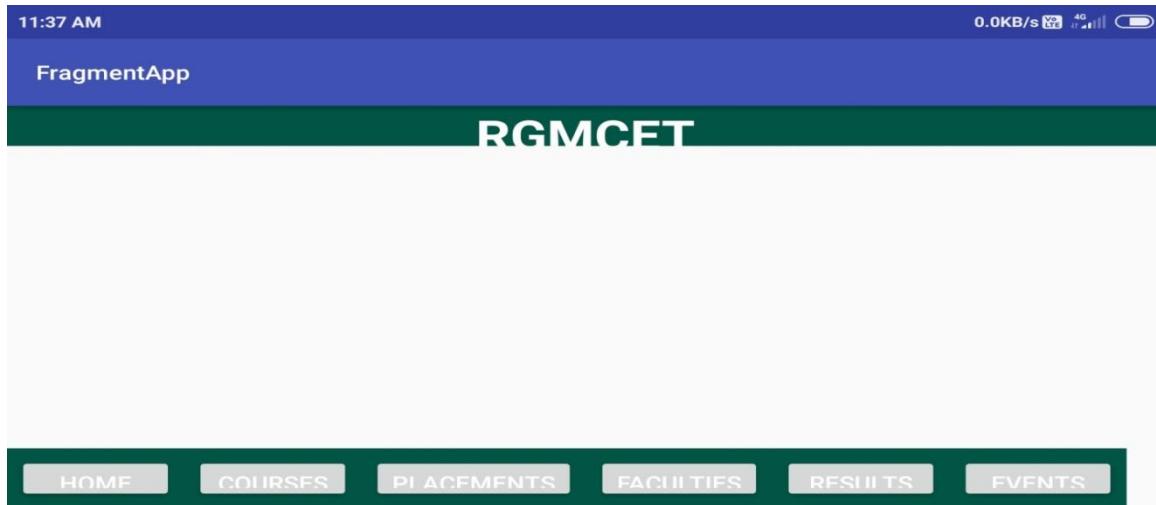
mainActivity.java

```
package com.example.hi.fragmentapp;
import android.app.FragmentManager;
import android.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {
    FragmentManager fManager;
    FragmentTransaction tx;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        fManager=getFragmentManager();
        /*tx=fManager.beginTransaction();
        tx.add(R.id.frag1, new HomeFragment());
        tx.commit();*/
    }
}
```

```
public void home(View v){  
    tx=fManager.beginTransaction();  
    tx.replace(R.id.frag1, new HomeFragment());  
    tx.commit();  
}  
  
public void courses(View v){  
    tx=fManager.beginTransaction();  
    tx.add(R.id.frag1, new CoursesFragment());  
    tx.commit();  
}  
  
public void faculties(View v){  
    tx=fManager.beginTransaction();  
    tx.add(R.id.frag1, new FacultyFragment());  
    tx.commit();  
}  
public void results(View v){  
    tx=fManager.beginTransaction();  
    tx.add(R.id.frag1, new ResultsFragment());  
    tx.commit();  
}  
public void placements(View c){  
    tx=fManager.beginTransaction();  
    tx.add(R.id.frag1, new PlacementFragment());  
    tx.commit();  
}  
  
public void events(View v)  
{  
    tx=fManager.beginTransaction();  
    tx.add(R.id.frag1, new EventsFragment());  
    tx.commit();  
}
```

OUTPUT:



Storage Methods:

- Shared preferences**
- SQLite DataBase**
- Files[XML/JSON]**

Shared Preferences:

- SPF is used to maintain the data in a **KEY, Value pairs**.
- SharedPreferences interface is used to manage SharedPreferences.
- **getSharedPreferences()** method is used to get the implementation object for Shared preferences.

```
SharedPreferences spf= getSharedPreferences("spf_name",  
Context.MODE_PRIVATE);
```

- if the SharedPreferences is not available with the specified name then it will create a new SharedPreferences.

-if the SharedPreferences is already available with the specified name then it will get the data from existing SharedPreferences.

-Following are the possible parameters to specify MODE.

Context.MODE_PRIVATE
Context.MODE_WORLD_READABLE
Context.MODE_WORLD_WRITABLE

- By using above object(spf) only we can read the data from SPF, to write the data into SPF additionally we have to create an object for SharedPreferences.Editor.

```
SharedPreferences.Editor spe=spf.edit();
spe.putXXX("key", value);      XXX-datatype
spe.commit();
```

-spf.getXXX("key","default value") method is used to read the data from SPF, if the data is available with the specified key it will return the actual value otherwise it will return the default value.

-internally SPF will maintain the data in XML file, we can explore the XML file using ADM[Android Device Monitor].

ADM >> data >> data >> pkg_name >> shared_prfs >> spf_name.xml

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

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/flayout">
    </FrameLayout>
</LinearLayout>
```

Login.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="5dp"
    android:layout_marginBottom="5dp"
```

```
    android:layout_marginRight="5dp"
    android:layout_marginLeft="5dp" >

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter username"
    android:id="@+id/l_uname" />

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter password"
    android:id="@+id/l_pwd"
    android:inputType="textPassword"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Login"
    android:textColor="#FFFFFF"
    android:background="#054"
    android:id="@+id/l_login"
    android:layout_gravity="center" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Register"
    android:textColor="#FFFFFF"
    android:background="#054"
    android:id="@+id/l_register"
    android:layout_gravity="center"
    android:layout_marginTop="5dp" />

</LinearLayout>
```

LoginFragment:

```
package com.android.developer.spftest;
import android.app.FragmentManager;
import android.app.FragmentTransaction;
import android.content.Context;
import android.content.SharedPreferences;
import android.os.Bundle;
```

```
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class LoginFragment extends android.app.Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {
        View v=inflater.inflate(R.layout.login,container,false);
        final EditText luname=(EditText)v.findViewById(R.id.l_uname);
        final EditText lupass=(EditText)v.findViewById(R.id.l_pwd);
        Button btn_login=(Button)v.findViewById(R.id.l_login);
        Button btn_register=(Button)v.findViewById(R.id.l_register);
        btn_login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                SharedPreferences spf= getActivity().getSharedPreferences("myspf",
                        Context.MODE_PRIVATE);
                String username=spf.getString("uname",null);
                String userpwd=spf.getString("upass",null);

                if(luname.getText().toString().equals(username) &&
                        lupass.getText().toString().equals(userpwd)) {
                    FragmentManager fManager=getFragmentManager();
                    FragmentTransaction tx=fManager.beginTransaction();
                    tx.replace(R.id.flayout,new WelcomeFragment());
                    tx.commit();

                }
                else {
                    Toast.makeText(getActivity(),"Invalid credentials", Toast.LENGTH_LONG).show();
                }
            }
        });

        btn_register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                FragmentManager fManager=getFragmentManager();
                FragmentTransaction tx=fManager.beginTransaction();
```

```
        tx.replace(R.id.flayout,new RegisterFragment());
        tx.addToBackStack("true");
        tx.commit();
    }
});
return v;
//return super.onCreateView(inflater, container, savedInstanceState);
}
}
```

register.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="5dp"
    android:layout_marginBottom="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp">

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter username"
        android:id="@+id/r_uname"
        android:textColorHint="#054"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter password"
        android:inputType="textPassword"
        android:id="@+id/r_upass"
        android:textColorHint="#054"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter email address"
        android:inputType="textEmailAddress"
        android:id="@+id/r_email"
        android:textColorHint="#054"/>
```

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:hint="Enter mobile number"  
    android:inputType="phone"  
    android:id="@+id/r_mobno"  
    android:textColorHint="#054"/>  
  
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Register"  
    android:id="@+id/btn_r_register"  
    android:layout_gravity="center"  
    android:background="#054"  
    android:textColor="#FFFFFF"/>  
</LinearLayout>
```

registerFragment:

```
package com.android.developer.spftest;  
import android.app.FragmentManager;  
import android.app.FragmentTransaction;  
import android.content.Context;  
import android.content.SharedPreferences;  
import android.os.Bundle;  
import android.support.annotation.Nullable;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.EditText;  
  
public class RegisterFragment extends android.app.Fragment {  
    @Nullable  
    @Override  
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup  
    container, Bundle savedInstanceState) {  
        View v=inflater.inflate(R.layout.register,container,false);  
        final EditText r_uname=(EditText)v.findViewById(R.id.r_uname);  
        final EditText r_upass=(EditText)v.findViewById(R.id.r_upass);  
        final EditText r_email=(EditText)v.findViewById(R.id.r_email);  
        final EditText r_mobno=(EditText)v.findViewById(R.id.r_mobno);  
        Button btn_register=(Button)v.findViewById(R.id.btn_r_register);  
        btn_register.setOnClickListener(new View.OnClickListener() {
```

```
    @Override
    public void onClick(View v) {
        SharedPreferences spf= getActivity().getSharedPreferences("myspf",
                Context.MODE_PRIVATE);
        SharedPreferences.Editor spe=spf.edit();
        spe.putString("uname",r_uname.getText().toString());
        spe.putString("upass",r_upass.getText().toString());
        spe.putString("email",r_email.getText().toString());
        spe.putLong("mno", Long.parseLong(r_mobno.getText().toString()));
        spe.commit();
        FragmentManager fManager= getFragmentManager();
        FragmentTransaction tx=fManager.beginTransaction();
        tx.replace(R.id.flayout,new LoginFragment());
        tx.commit();
    }
}
return v;
//return super.onCreateView(inflater, container, savedInstanceState);
}
```

Welcome.xml:

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="WELCOME TO"
        android:textSize="40sp"
        android:textColor="#FF00"
        android:layout_gravity="left" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="RGMCSE DEPT"
        android:gravity="right"
        android:textSize="40sp"
        android:textColor="#708090"/>
</LinearLayout>
```

WelcomeFragment:

```
package com.android.developer.spftest;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class WelcomeFragment extends android.app.Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {
        View v=inflater.inflate(R.layout.welcome,container,false);
        return v;
        //return super.onCreateView(inflater, container, savedInstanceState);
    }
}
```

MainActivity.java:

```
package com.example.hi.spftest;
import android.app.FragmentManager;
import android.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

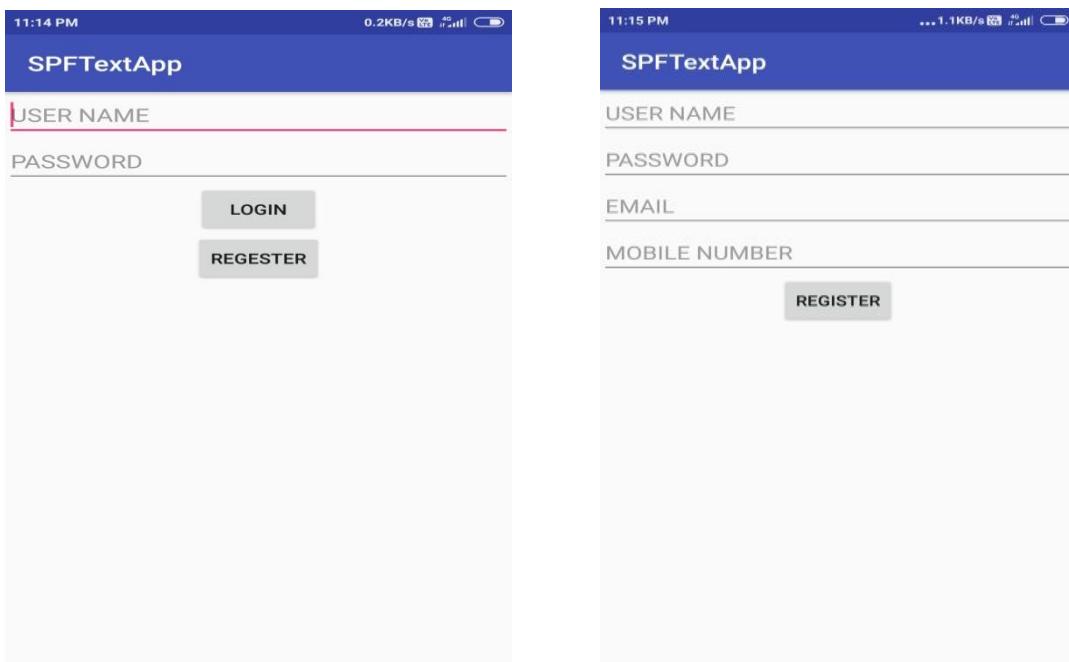
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        FragmentManager fManager=getFragmentManager();
        FragmentTransaction tx=fManager.beginTransaction();
        tx.add(R.id.flayout,new LoginFragment());
        tx.commit();
    }
}
```

AndroidManifest.xml:

```

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

```

OUTPUT:

- By using SPF we can maintain huge amount of data, but for every Value we have to specify a unique key it's difficult to assign and remember the keys that's why SPF is preferred to maintain limited data (e.g. Pattern lock key, Alarm time, login credentials ...), if you want to maintain the huge amount of data Android is preferred to use SQLiteDatabase.

SQLite Database:

- SQLite Database is used to maintain structured data in Android.
- SQLiteDatabase class is used to manage SQLite database.
- openOrCreateDatabase(db_name, mode, cursor_factory) method is used to get the object for SQLiteDatabase.

SQLiteDatabase dBase=openOrCreateDatabase(db_name, mode, cursor_factory);

- SQLiteDB is providing built-in methods to perform CRUD Operations.

**-insert()
-query() // read
-update()
-delete()**

- Apart from builtin methods we can execute nativeSQL Statements by using following methods.

**- execSQL(sql_query) // I U D C
- rawQuery(sql_query) // READ
- We can explore the SQLite the DB file using ADM.**

ADM >> file_explorer >> data >> data >> pkg_name >>database >> file_name.db

- To open the SQLite db file we use SQLite browser.

Caused by: android.database.sqlite.SQLiteException: table employee already exists (code 1);,

While compiling: create table employee (id number, name varchar (50),
desig varchar(50), dept varchar(50))

Read Operation:

```
public void read(View v){  
/*query(String table, String[] columns, String selection, String[] selectionArgs, String  
groupBy, String having, String orderBy)*/  
Cursor c=dBase.query("employee", null, null, null, null, null, null);  
    while(c.moveToNext()) {  
        int id=c.getInt(0);
```

```
String name=c.getString(1);
String desig=c.getString(2);
String dept=c.getString(3);
Toast.makeText(MainActivity.this, id+"\n"+name+"\n"+desig+"\n" +dept,
                Toast.LENGTH_LONG).show();
        }
    }
```

Update Operation:

```
public void update(View v){
/*String table, ContentValues values, String whereClause, String[] whereArgs*/
ContentValues cv=new ContentValues();
cv.put("name",et2.getText().toString());
cv.put("desig",et3.getText().toString());
int count=dBase.update("employee", cv,"id=? and dept=?",
        new String[]{et1.getText().toString(),et4.getText().toString()});
if(count>0)
Toast.makeText(MainActivity.this,"Record is updated..",
                Toast.LENGTH_LONG).show();
et1.setText(""); et2.setText(""); et3.setText(""); et4.setText("");
else{
Toast.makeText(MainActivity.this,"Failed to
                    update.. ", Toast.LENGTH_LONG).show(); }
}
```

Delete Operation:

```
int count=dBase.delete("employee","id=?", new String[]{et1.getText().toString()});
```

activity_main.xml file:

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

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter ID"
        android:id="@+id/et1"/>
```

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:hint="ENTER NAME"  
    android:id="@+id/et2"/>  
  
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:hint="DESIG"  
    android:id="@+id/et3" />  
  
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:hint="DEPT"  
    android:id="@+id/et4"/>  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="horizontal">  
  
    <Button  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:text="INSERT"  
        android:onClick="insert"  
        android:layout_weight="0.5"/>  
  
    <Button  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:text="READ"  
        android:onClick="read"  
        android:layout_weight="0.5"/>  
  
</LinearLayout>  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="horizontal">
```

```
<Button  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:text="UPDATE"  
    android:onClick="update"  
    android:layout_weight="0.5"/>  
  
<Button  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:text="DELETE"  
    android:onClick="delete"  
    android:layout_weight="0.5"/>  
</LinearLayout>  
  
<ListView  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:id="@+id/lview">  
  
</ListView>  
  
</LinearLayout>
```

mainActivity.java:

```
package com.example.hi.sqlitedbapp;  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.os.AsyncTask;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.ArrayAdapter;  
import android.widget.EditText;  
import android.widget.ListView;  
import android.widget.Toast;  
import org.w3c.dom.Text;  
import java.util.ArrayList;  
import static android.content.Context.MODE_PRIVATE;
```

```

public class MainActivity extends AppCompatActivity {
    EditText et1, et2, et3, et4;
    SQLiteDataBase dBase;
    ListView lview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        et1 = (EditText) findViewById(R.id.et1);
        et2 = (EditText) findViewById(R.id.et2);
        et3 = (EditText) findViewById(R.id.et3);
        et4 = (EditText) findViewById(R.id.et4);

        dBase = openOrCreateDatabase("empdb", Context.MODE_PRIVATE, null);
        dBase.execSQL("create table if not exists tbemp (empid number,
        empname varchar(100),empdesig varchar(100),empdept varchar(100))");
        lview = (ListView) findViewById(R.id.lview);
    }
    public void insert(View v) {
        ContentValues cv = new ContentValues();
        // dBase.insert("employee",null,Integer.parseInt(et1.getText().toString()));

        cv.put("empid", et1.getText().toString());
        cv.put("empname", et2.getText().toString());
        cv.put("empdesig", et3.getText().toString());
        cv.put("empdept", et4.getText().toString());    long status =
        dBase.insert("tbemp", null, cv);
        if (status != -1) {
            Toast.makeText(MainActivity.this, "Data Inserted", Toast.LENGTH_LONG).show();
            et1.setText("");
            et2.setText("");
            et3.setText("");
            et4.setText("");
            read(v);
        } else {
            Toast.makeText(MainActivity.this, "Fails To Inserted", Toast.LENGTH_LONG).show();
        }
    }
    public void read(View v) {
        Cursor c = dBase.query("tbemp", null, null, null, null, null, null);
        ArrayList<String> list = new ArrayList<>();
        while (c.moveToNext()) {
            String msg = c.getInt(0) + " | " + c.getString(1) + " | " + c.getString(2) + " | " + c.getString(3);
            list.add(msg);
        }
    }
}

```

```

ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
        android.R.layout.simple_list_item_single_choice, list);

    lview.setAdapter(adapter);
}

public void update(View v) {
    ContentValues cv = new ContentValues();
    cv.put("empname", et2.getText().toString());
    cv.put("empdesig", et3.getText().toString());
    cv.put("empdept", et4.getText().toString());

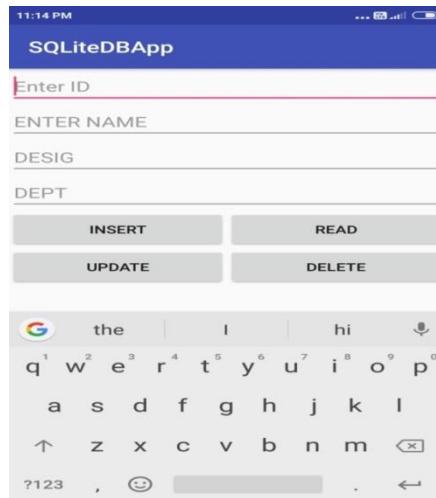
    int status = dBase.update("tbemp", cv, "empid=?", new String[]{et1.getText().toString()});
    if(status > 0) {
        Toast.makeText(MainActivity.this, "Success", Toast.LENGTH_LONG).show();
    } else {
        Toast.makeText(MainActivity.this, "FAIL", Toast.LENGTH_LONG).show();
    }
}

public void delete(View v){
    int status = dBase.delete("tbemp", "empid=?", new String[]{et1.getText().toString()});
    if(status > 0){

        Toast.makeText(MainActivity.this,"SUCCESS",Toast.LENGTH_LONG).show();
    }else{
        Toast.makeText(MainActivity.this, "UNSUCCESSFUL",Toast.LENGTH_LONG).show();
    }
}

```

OUTPUT:



Android Telephony:

- SMS
- Calls
- Emails
 - Builtin Activity
 - Java Mail API

SMS: - android.telephony.SmsManager class is used to send text messages in Android.

```
SmsManager sManager=SmsManager.getDefault();  
  
sManager.sendTextMessage(rec_num, send_num, message,  
sent_intent, delevery_intent);
```

permission :

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

Call:

-for making calls android is providing builtin activity use implicit intents to call builtin Activity.

```
Intent i=new Intent();  
i.setAction(Intent.ACTION_CALL);  
i.setData(Uri.parse("tel:"+number));  
startActivity(i);
```

permission :

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

E-mail (builtin Activity) :

- for sending an email android is providing a builtin activity use implicit intents to call builtin Activity.

```
Intent i=new Intent();
```

```
i.setAction(Intent.ACTION_SEND);
i.putExtra(Intent.EXTRA_EMAIL, new
String[]{m1,m2,m3..});
i.putExtra(Intent.EXTRA_SUBJECT,"subject_here");
i.putExtra(Intent.EXTRA_TEXT,"text_here");
i.putExtra(Intent.EXTRA_STREAM, uri_object);
i.setType("message/rfc822"); // enable MIME .
startActivity(i.createChooser(intent_obj,"message here"));
```

permissions :

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

Attachment functionality:

- Android is providing a builtin Activity to show the list of files, use implicit intents to call builtin Activity.

```
Intent i=new Intent();
i.setAction(Intent.ACTION_GET_CONTENT);
i.setType("*/*");
startActivityForResult(intent_obj, request_code(int));
```

- In the above method if we specify the second parameter (request_code) >=0 after the next Activity is completed (after selecting a file) it will invoke onActivityResult() method in the current activity class.

```
public void onActivityResult(int req_code, int res_code, Intent i){
    Uri u=i.getData();
}
```

By using builtin Activity we can't send an email in the background, user has to select any one of the email client for sending an email, if you want to send an email in the background without any user interaction use JavaMail API.

First add these permissions in Manifest File:

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

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

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

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

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

Go to Activity_main.XML file:

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

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Mobile Num"
        android:id="@+id/et1" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Message"
        android:id="@+id/et2" />

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

        <Button
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:text="SEND SMS"
            android:onClick="sendSms" />

        <Button
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:text="CALL"
```

```
        android:onClick="call" />
    </LinearLayout>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/et3"
        android:hint="Enter Mail id"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/et4"
        android:hint="Enter Subject" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/et5"
        android:hint="Enter Message"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="ATTACH"
        android:onClick="attach"
        android:layout_gravity="right"/>

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

        <Button
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:text="SEND MAIL"
            android:onClick="send_mail"/>

        <Button
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:text="JAVA MAIL"
            android:onClick="java_mail"/>

```

```
</LinearLayout>  
</LinearLayout>
```

Now create two Activity XML files.

- 1) Goto java folder and rightclick on folder → new → Activity → select Empty Activity and provide name

Activity_delivery.xml file:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical">  
  
    <TextView  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:text="Message Delivered"  
        android:textSize="30sp"/>  
  
</LinearLayout>
```

DeliveryActivity.java:

```
package com.example.hi.telephonetextapp;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
  
public class DeliveryActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_delivery);  
    }  
}
```

- 2) Goto java folder and rightclick on folder → new → Activity → select Empty Activity and provide name

Activity_send.xml file:

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Message Send"
        android:textSize="30sp"/>

</LinearLayout>
```

SendActivity.java:

```
package com.example.hi.telephonetextapp;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class sendActivity extends AppCompatActivity {

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

MainActivity.Java

```
package com.example.hi.telephonetextapp;
import android.app.PendingIntent;
import android.content.Intent;
import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.view.View;
import android.widget.EditText;
```

```

public class MainActivity extends AppCompatActivity {
    EditText et1,et2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        et1=(EditText)findViewById(R.id.et1);
        et2=(EditText)findViewById(R.id.et2);
    }

    public void sendSms(View v)
    {
        SmsManager sManager=SmsManager.getDefault();
        Intent send_intent=new Intent(this,sendActivity.class);
        Intent del_intent=new Intent(this,DelivaryActivity.class);
        PendingIntent
            P_Send_Intent=PendingIntent.getActivity(this,0,send_intent,0);
        PendingIntent P_Del_Intent=PendingIntent.getActivity(this,0,del_intent,0);
        sManager.sendTextMessage(et1.getText().toString(),null,et2.getText().toString(),
            P_Send_Intent, P_Del_Intent);
    }

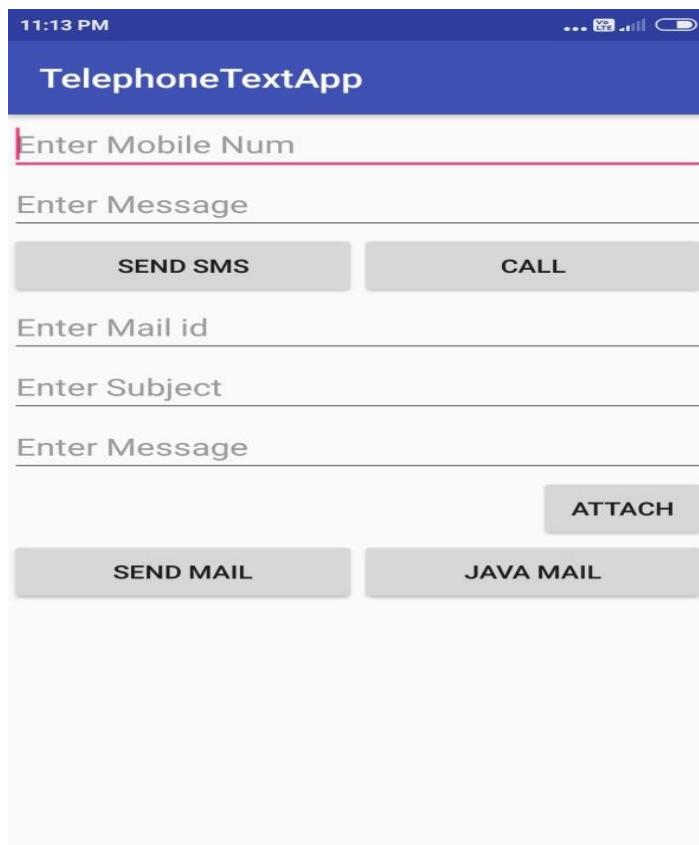
    public void call(View v)
    {
        Intent i=new Intent();
        i.setAction(Intent.ACTION_CALL);
        i.setData(Uri.parse("tel:"+et1.getText().toString()));
        startActivity(i);
    }
    public void attach(View v){
        Intent i=new Intent();
        i.setAction(Intent.ACTION_GET_CONTENT);
        i.setType("*/*");
        startActivityForResult(i,123);

    }
    Uri u;
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if(resultCode==RESULT_OK){
            u=data.getData();
        }
    }
}

```

```
public void send_mail(View v){  
    EditText et3=(EditText)findViewById(R.id.et3);  
    EditText et4=(EditText)findViewById(R.id.et4);  
    EditText et5=(EditText)findViewById(R.id.et5);  
    Intent i=new Intent();  
    i.setAction(Intent.ACTION_SEND);  
    i.putExtra(Intent.EXTRA_EMAIL,new String[]{et3.getText().toString()});  
    i.putExtra(Intent.EXTRA_SUBJECT,et4.getText().toString());  
    i.putExtra(Intent.EXTRA_TEXT, et5.getText().toString());  
    i.putExtra(Intent.EXTRA_STREAM, u);  
    i.setType("message/rfc822");  
    startActivity(i.createChooser(i, "Select any Email Client"));  
}  
}
```

OUTPUT:



UNIT-IV

UNIT-IV:

Multimedia in Android: Media Player, Video View, Audio Recording, Video recording, Camera, Gallery. **Service:** Service, Service lifecycle methods.

Android Media :

- MediaPlayer
- VideoView
- AudioRecording
- VideoRecording
- Camera & Gallery

Media Player:

- android.media.MediaPlayer class is used to play audio files in Android.
- we can get the audio file from 3 different sources.
 - apk
 - Storage [internal / external]
 - network

apk: [res >> raw >> audio_file.mp3 ..]

```
MediaPlayer mPlayer = MediaPlayer.create(context, R.raw.audio_file);
```

Storage: | network :

- 1) MediaPlayer mPlayer=new MediaPlayer();
mPlayer.setFileUri(Context, Uri_Object);
- 2) MediaPlayer mPlayer=new MediaPlayer();
mPlayer.setDataSource(Context, Uri_Object);
- 3) MediaPlayer mPlayer=new MediaPlayer();
mPlayer.prepare();

network:

- 1) MediaPlayer mPlayer=new MediaPlayer();
mPlayer.setDataSource ("network_url");
- 2) MediaPlayer mPlayer=new MediaPlayer();
mPlayer.setDataSource(audio_file_url);
- 2) MediaPlayer mPlayer=new MediaPlayer();
mPlayer.prepare();

following are the major methods in MediaPlayer class:

- start()**
- seekTo(int)**
- setDataSource()**
- pause()**
- getCurrentPosition()**
- prepare()**
- stop()**
- getDuration()**
- reset()**

Handler:

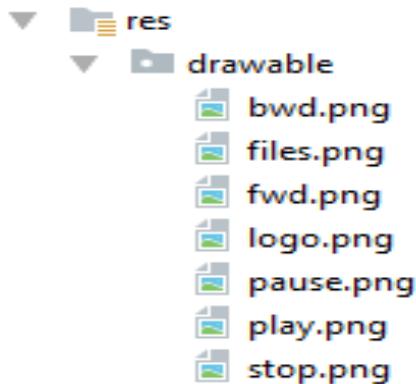
- Android.os.Handler class is used to execute a process continuously for the specified time interval.
- Internally Handler is a separate thread, we can communicate with activity Thread.

```
Inside init(){  
    android.os.Handler handler=new Handler();  
    handler.postDelayed(new Runnable(){  
        public void run() {  
            // logic to execute;  
            Handler.postDelayed(this, duration(milli-seconds));  
            Sbar.setProgress(mp.getCurrentPosition()); //cycle  
        }  
    }, delay_milli_seconds);
```

Setting Image:

```
MediaMetadataRetriever mr=new MediaMetadataRetriever();  
Uri u= Uri.parse("android.resource://com.example.hi.mediaplayer/raw/dj");  
byte[] im_data=mr.getEmbeddedPicture();  
Bitmap bmp= BitmapFactory.decodeByteArray(im_data, 0, im_data.length);  
iview.setImageBitmap(bmp);
```

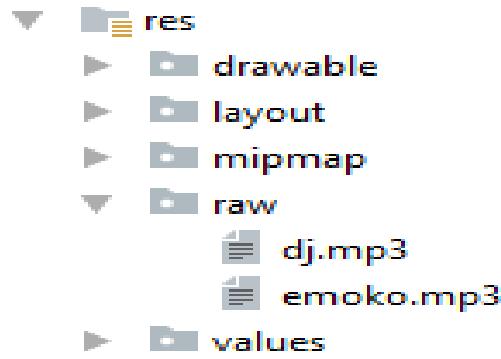
--First download the all media player images like **logo image, backword image, play image, forward image, pause image, stop image and files image**. These images download with **png** format.



-Next you create one folder, the folder name is "raw"

-Go to res folder >> new >> **Android resource directory** >> now open the dialogue box >> choose resource type to raw folder give directory name to "raw" then click on **Ok Button**.

-After **creating raw** folder you can copy any one mp3 file in your device and place with in raw folder.



activity_main.xml

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

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="0dp"
```

```
    android:layout_weight="0.6"
    android:src="@drawable/logo"
    android:id="@+id/iview1"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="0.1"
    android:orientation="horizontal">

    <TextView
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:text="Cur Pos:"
        android:textSize="15sp"
        android:gravity="left"
        android:id="@+id/cp"
        android:textStyle="bold"
        android:layout_weight="0.5" />

    <TextView
        android:textStyle="bold"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:text="Tot Dur:"
        android:textSize="15sp"
        android:gravity="right"
        android:id="@+id/td"
        android:layout_weight="0.5" />
</LinearLayout>

<SeekBar
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="0.1"
    android:id="@+id/s1" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="0.2"
    android:orientation="horizontal">

    <ImageView
        android:layout_width="0dp"
        android:layout_height="match_parent"
```

```
    android:layout_weight="0.16"
    android:src="@drawable/bwd"
    android:id="@+id/bwd"
    android:onClick="media"
    android:padding="5dp" />

<ImageView
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="0.16"
    android:src="@drawable/play"
    android:id="@+id/play"
    android:onClick="media"
    android:padding="5dp" />

<ImageView
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="0.16"
    android:src="@drawable/pause"
    android:id="@+id/pause"
    android:onClick="media"
    android:padding="5dp" />

<ImageView
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="0.16"
    android:src="@drawable/stop"
    android:id="@+id/stop"
    android:onClick="media"
    android:padding="5dp" />

<ImageView
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="0.16"
    android:src="@drawable/fwd"
    android:id="@+id/fwd"
    android:onClick="media"
    android:padding="5dp" />

<ImageView
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="0.16"
```

```
    android:src="@drawable/files"
    android:id="@+id/files"
    android:onClick="media"
    android:padding="5dp" />

</LinearLayout>

</LinearLayout>
```

MainActivity.java:

```
package com.example.hi.mediaplayer;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.media.MediaMetadata;
import android.media.MediaMetadataRetriever;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.SeekBar;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    SeekBar s1;
    TextView cp,td;
    MediaPlayer mPlayer;
    Uri u;
    ImageView iview;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        s1=(SeekBar)findViewById(R.id.s1);
        cp=(TextView)findViewById(R.id.cp);
        td=(TextView)findViewById(R.id.td);
        iview=(ImageView)findViewById(R.id.iview1);
    }
}
```

```

void init(){
    if(mPlayer==null) {
        if(u==null) {
            mPlayer = MediaPlayer.create(MainActivity.this,R.raw.dj);
        MediaMetadataRetriever mr=new MediaMetadataRetriever();
        mr.setDataSource(MainActivity.this,Uri.parse("android.resource://com.example.hi.mediaplayer/raw/dj"));
        byte[] im_data=mr.getEmbeddedPicture();
        Bitmap bmp= BitmapFactory.decodeByteArray(im_data, 0, im_data.length);
        iview.setImageBitmap(bmp);
    }else{
        try{
            mPlayer = new MediaPlayer();
            mPlayer.setDataSource(MainActivity.this, u);
            MediaMetadataRetriever mr=new MediaMetadataRetriever();
            mr.setDataSource(MainActivity.this,u);
            byte[] im_data=mr.getEmbeddedPicture();
            Bitmap bmp= BitmapFactory.decodeByteArray(im_data,0,im_data.length);
            iview.setImageBitmap(bmp);
            mPlayer.prepare();
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
    td.setText("Tot Dur :" + mPlayer.getDuration());
    s1.setMax(mPlayer.getDuration());
    s1.setProgress(0);
    s1.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
            mPlayer.seekTo(i);
            cp.setText("Cur pos:" + i);
        }
        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {
        }
        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {
    });
}

final Handler handler=new Handler();
handler.postDelayed(new Runnable() {
    @Override
    public void run() {

```

```

    s1.setProgress(mPlayer.getCurrentPosition());
    cp.setText("Cur Pos:"+mPlayer.getCurrentPosition());
    handler.postDelayed(this,5000);
}
},5000);
}
}

public void media(View v){
switch (v.getId()){
case R.id.bwd:
mPlayer.seekTo(mPlayer.getCurrentPosition()-mPlayer.getDuration()/10);
cp.setText("Cur Pos:"+mPlayer.getCurrentPosition());
break;

case R.id.play:
init();
mPlayer.start();
break;

case R.id.pause:
mPlayer.pause();
break;

case R.id.stop:
mPlayer.stop();
mPlayer=null;
init();
break;

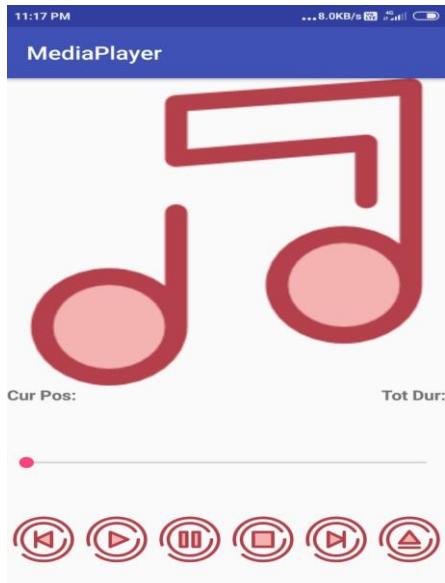
case R.id.fwd:
mPlayer.seekTo(mPlayer.getCurrentPosition()+mPlayer.getDuration()/10);
cp.setText("Cur Pos:"+mPlayer.getCurrentPosition());
break;

case R.id.files:
Intent i=new Intent();
i.setAction(Intent.ACTION_GET_CONTENT);
i.setType("audio/*");
startActivityForResult(i,123);
break;
}
}
}

```

```
@Override  
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    super.onActivityResult(requestCode, resultCode, data);  
  
    if(resultCode==RESULT_OK && mPlayer!=null) {  
        mPlayer.stop();  
        mPlayer = null;  
        u=data.getData();  
        init();  
        mPlayer.start();  
    }  
}  
}
```

OUTPUT:



VideoView:

VideoView is one of the UI component in Android which is used to play video files.

xml:

```
<VideoView  
    android:id="@+id/vview1"  
...../>
```

java:

```
VideoView vview=(VideoView)findViewById(R.id.vview1);
vview.setVideoPath(file_url | network_url);
vview.start();
```

For providing pause, forward, backward and seekbar functionality set the following object to VideoView.

```
vview.setMediaController(new MediaController(this));
```

Activity main.XML File:

-Initially we select Landscap(this is available on xml design page with orientation editor icon).

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

    <VideoView
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.8"
        android:id="@+id/Vview"/>

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.2"
        android:orientation="vertical">
        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="LIST"
            android:onClick="list"/>

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
    android:text="START"
    android:onClick="start"/>

</LinearLayout>
</LinearLayout>
```

MainActivity.Java:

```
package com.example.hi.videoviewapp;
import android.content.Intent;
import android.net.Uri;
import android.provider.MediaStore;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.MediaController;
import android.widget.VideoView;

public class MainActivity extends AppCompatActivity {
    VideoView Vview;
    Uri u;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Vview=(VideoView)findViewById(R.id.Vview);
        Vview.setMediaController(new MediaController(this));
    }
    public void list(View v){
        Intent i=new Intent();
        i.setAction(Intent.ACTION_GET_CONTENT);
        i.setType("video/*");
        startActivityForResult(i,111);
    }
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if(requestCode==RESULT_OK);
```

```
{  
    u=data.getData();  
    // Vview.setVideoURI(u);  
    //Vview.start();  
  
}  
}  
  
public void start(View v){  
    Vview.setVideoURI(u);  
    Vview.start();  
}  
}
```

Manifestfile.xml

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

OUTPUT:



Audio Recording:

- android.media.MediaRecorder class is used to record audio and video in Android.

MediaRecorder recorder=new MediaRecorder();

- To record audio set the following properties to media recorder:

- 1) **recorder.set AudioSource(MediaRecorder.AudioSource.MIC);**
- 2) **recorder.setOutputFormat(MediaRecorder.OutputFormat.AMR_NB);**
- 3) **recorder.set AudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);**
- 4) **recorder.setOutputFile("file_loc/file_name.amr");
recorder.prepare();
recorder.start();**

Permissions:

**RECORD_AUDIO
WRITE_EXTERNAL_STORAGE**

Note:

- Android is not providing any api support to pause record an audio, to provide pause functionality maintain the temp files and combine all these temp files into a single file when user selects stop.

Activity_main.XML File:

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

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.7"
        android:src="@drawable/ic_keyboard_voice_black_24dp" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.1"
        android:id="@+id/status"
        android:text="Recording status"
        android:textSize="30sp"
        android:gravity="center"
        android:textStyle="bold"
        android:textColor="#0000FF" />

    <ImageView
        android:id="@+id/r_ivew"
        android:layout_width="60dp"
        android:layout_height="0dp"
        android:layout_weight="0.2"
        android:layout_gravity="center"
        android:src="@drawable/ic_radio_button_checked_black_24dp"
        android:onClick="record"/>

</LinearLayout>
```

MainActivity.java File:

```

import android.media.MediaRecorder;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    MediaRecorder recorder;
    ImageView iview;
    TextView tv1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        iview=(ImageView)findViewById(R.id.r_ivew);
        tv1=(TextView)findViewById(R.id.status);
    }

    public void init(){
        recorder =new MediaRecorder();
        recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
        recorder.setOutputFormat(MediaRecorder.OutputFormat.AMR_NB);
        recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
        recorder.setOutputFile("/storage/emulated/0/cse"+
                System.currentTimeMillis() + ".amr");
        try {
            recorder.prepare();
        }catch (Exception e){
            e.printStackTrace();
        }
    }

    public void record(View v){
        if(recorder==null){
            init();
        }

        iview.setImageResource(R.drawable.ic_radio_button_checked_black_24dp);
        tv1.setText("Recording is started");
        try{
            recorder.start();
        }
        catch (Exception e) {
    
```

```
        e.printStackTrace();
    }
}else{
    iview.setImageResource(R.drawable.ic_brightness_1_black_24dp);
    tv1.setText("Record is stoped..");
    recorder.stop();
    recorder=null;
}

}
```

Next go to APP Folder click on right -→new→select vectorAsset

Now create three icons and paste drawable folder.

```
1 <vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24.0"
    android:viewportHeight="24.0">
<path
    android:fillColor="#CC0000"
    android:pathData="M12,12m-10,0a10,10 0,1 1,20 0a10,10 0,1 1,-20 0"/>
</vector>
```

```
2 <vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24.0"
    android:viewportHeight="24.0">
<path
    android:fillColor="#0000FF"
    android:pathData="M12,15c1.66,0 2.99,-1.34 2.99,-3L15,6c0,-1.66 -
1.34,-3 -3,-3S9,4.34 9,6v6c0,1.66 1.34,3 3,3zM17.3,12c0,3 -2.54,5.1 -
5.3,5.1S6.7,15 6.7,12L5,12c0,3.42 2.72,6.23 6,6.72L11,22h2v-3.28c3.28,-
0.48 6,-3.3 6,-6.72h-1.7z"/>
</vector>
```

```
3 <vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24.0"
    android:viewportHeight="24.0">
```

```
<path
    android:fillColor="#38f820"
    android:pathData="M12,7c-2.76,0 -5,2.24 -5,5s2.24,5 5,5 5,-2.24 5,-5 -
2.24,-5 -5,-5zM12,2C6.48,2 2,6.48 2,12s4.48,10 10,10 10,-4.48 10,-
10s17.52,2 12,2zM12,20c-4.42,0 -8,-3.58 -8,-8s3.58,-8 8,-8 8,3.58 8,8 -
3.58,8 -8,8z"/>
</vector>
```

Next add two permissions in Manifest FILE:

- 1) <uses-permission android:name="android.permission.RECORD_AUDIO"/>
- 2) <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

OUTPUT:



Video Recording:

- android.media.MediaRecorder class is used to record audio and video in Android.

MediaRecorder recorder=new MediaRecorder();

- To record video set the following properties to media recorder.

- 1) **recorder.set AudioSource(MediaRecorder.AudioSource.MIC);**
- 2) **recorder.set VideoSource(MediaRecorder.VideoSource.CAMERA);**
- 3) **CamcorderProfile
profile=CamcorderProfile.get(CamcorderProfile.Quality_High);**
- 4) **recorder.setProfile(profile);**
- 5) **recorder.setOutputfile("file_loc/file_name.mp4");**

-- By using the above properties we can record video but while we are recording video we have to show the video preview to the user to show the video preview we use an advanced UI component called **SurfaceView**.

XML:

```
<SurfaceView  
    android:id="@+id/sview1"  
    .....>
```

JAVA:

```
SurfaceView sview=(SurfaceView)findViewById(R.id.sview1);
```

- Directly we can't perform any operations on SurfaceView, to manage SurfaceView we have to create an object for SurfaceHolder.

```
SurfaceHolder sHolder=sview.getHolder();  
recorder.setPreviewDisplay(sHolder.getSurface());  
recorder.prepare()  
recorder.start();
```

permissions:

**-CAMERA
-RECORD_AUDIO
-WRITE_EXTERNAL_STORAGE**

Activity_main.xml file:

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

    <SurfaceView
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.8"
        android:id="@+id/sview1"/>

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.2"
        android:orientation="vertical">

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="START"
            android:onClick="start"/>

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="STOP"
            android:onClick="stop"/>

    </LinearLayout>

</LinearLayout>
```

MainActivity.java file:

```
package com.example.hi.videorecordingapp;
import android.media.CamcorderProfile;
import android.media.MediaRecorder;
import android.provider.Settings;
import android.support.v7.app.AppCompatActivity;
```

```
import android.os.Bundle;
import android.view.Surface;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.View;

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

    public void init() {
        recorder = new MediaRecorder();

        recorder.setAudioSource(MediaRecorder.AudioSource.MIC);

        recorder.setVideoSource(MediaRecorder.VideoSource.CAMERA);

        CamcorderProfile profile =
            CamcorderProfile.get(CamcorderProfile.Quality.HIGH);

        recorder.setProfile(profile);

        recorder.setOutputFile("/storage/emulated/0/cse"+System.currentTimeMillis()
                +".mp4");
    }

    SurfaceView sview=(SurfaceView)findViewById(R.id.sview1);

    SurfaceHolder sholder=sview.getHolder();

    recorder.setPreviewDisplay(sholder.getSurface());

    try {
        recorder.prepare();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void start(View v)
{
    init();
    recorder.start();
}
```

```
public void stop(View v)
{
    recorder.stop();
}
}
```

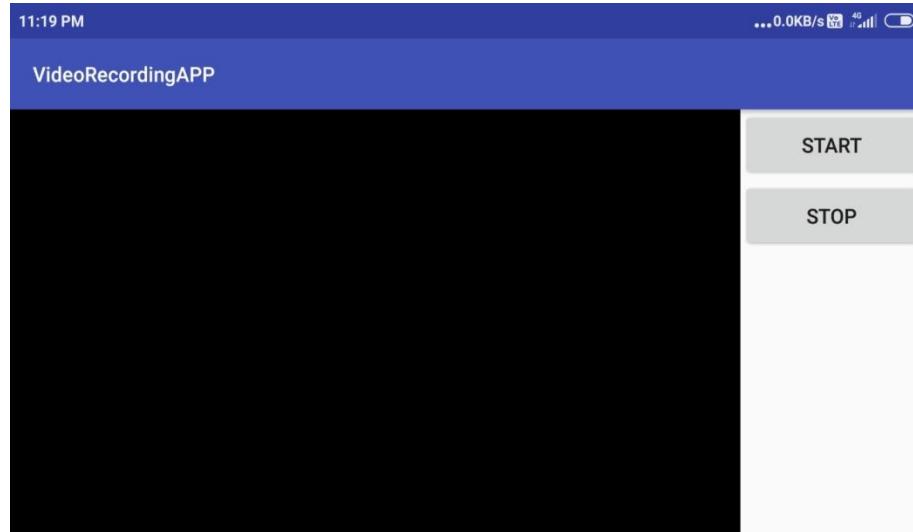
AndroidManifest.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hi.videorecordingapp">

    <uses-permission android:name="android.permission.CAMERA"/>
    <uses-permission
        android:name="android.permission.RECORD_AUDIO"/>
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity"
            android:screenOrientation="landscape">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

OUTPUT:



Camera & Gallery:

Camera:

- Camera is one of the built-in Activities in Android use implicit intents to call built-in Activity.

```
Intent i=new Intent ("android.media.action.IMAGE_CAPTURE");
startActivityForResult(i, request_code(123));
```

Gallery:

- Gallery is one of the built-in Activities in Android use implicit intents to call built-in Activity.

```
Intent i=new Intent();
i.setAction(Intent.ACTION_GET_CONTENT);
i.setType("image/*");
startActivityForResult(i, request_code(124));
```

-- In the above method (SAFR()) if we specify the second parameter (request_code >=0) after the next activity is completed it will invoke onActivityResult() method in the current activity class.

```
public void onActivityResult(int request_code, int result_code)
{
    if(request_code==123){
        // camera captured image
    }else if(request_code==124)
    {
        // image from gallery
    }
}
```

activity_main.xml file

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

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.9"
        android:id="@+id/ivew1"
        android:src="@mipmap/ic_launcher_round"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.1"
        android:orientation="horizontal">

        <Button
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.5"
            android:text="CAMERA"
            android:onClick="camera"/>

        <Button
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.5"
            android:text="GALLERY"
            android:onClick="gallery"/>
    
```

```
</LinearLayout>  
</LinearLayout>
```

MainActivity.java file:

```
package com.example.hi.camera_galleryapp;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.net.Uri;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.ImageView;  
  
public class MainActivity extends AppCompatActivity {  
  
    ImageView iview;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        iview=(ImageView)findViewById(R.id.iview1);  
    }  
  
    public void camera(View v){  
        Intent i=new Intent("android.media.action.IMAGE_CAPTURE");  
        startActivityForResult(i, 111);  
    }  
  
    public void gallery(View v){  
        Intent i=new Intent();  
        i.setAction(Intent.ACTION_GET_CONTENT);  
        i.setType("image/*");  
        startActivityForResult(i, 123);  
    }  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        super.onActivityResult(requestCode, resultCode, data);  
        if(requestCode==111 && resultCode==RESULT_OK){
```

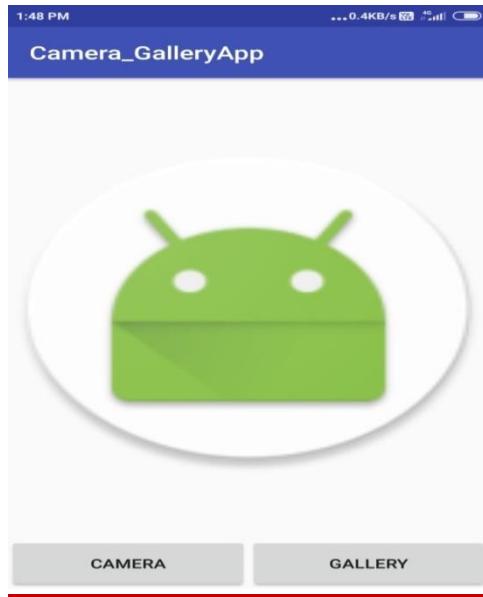
```
Object ob= data.getExtras().get("data");
Bitmap bmp=(Bitmap)ob;
iview.setImageBitmap(bmp);
}else if(requestCode==123 && resultCode==RESULT_OK){

    Uri u=data.getData();
    iview.setImageURI(u);
}
}
```

AndroidManifest.xml file:

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

OUTPUT:



Service:

- Service is one of the Component in Android.
- Service is used to run a long background process without any user Interaction.
- Examples for service is MediaPlayer, FMPlayer, Alarm, etc..
- We can create a new service by extending android.app.Service class (or) we can access the built-in services.

Steps to create a new Service:

1. Create a class as child of android.app.Service.
2. It is an abstract class having an abstract method called onBind() so provide the implementation for onBind() method.
3. Following are the major methods in Service class.
`onCreate();
onStartCommand();
onDestroy();`
4. When we start a service if the service is not available it will Invoke onCreate() and onStartCommand() methods.
5. If the service is already available it will invoke only onStartCommand() method.

6. When we stop service it will invoke onDestroy() method.
7. Service doesn't contain any UI we will manage service from Activity using the following methods.
 - i) Intent i=new Intent(context, service_name.class);
 - ii) startService(i);
 - iii) stopService(i);
8. Every service class we have to configure in the manifest.xml, with the following tag inside <application> tag.
<service android:name="pkg_name.class_name"/>
9. By using Service class we can't communicate with activity UI components. To communicate with activity from service use IntentService.

UNIT-V

UNIT-V:

Built-in Services (Location service, Notification service, Sensor Service, WIFI Service, Bluetooth Service, Vibrator Service), Broadcast Receivers.

Builtin Services:

getSystemService(Context.SERVICE_NAME) method is used to get the builtin service into Activity.

- Location Service**
- Notification Service**
- Sensor Service**
- WiFi Service**
- Bluetooth Service**
- Vibrator Service**

Location Service:

- getSystemService(Context.LOCATION_SERVICE) method is used to get the builtin location service into Activity, application framework is providing a class called LocationManager to manage location service.

```
LocationManager lManager=(LocationManager)
getSystemService(Context.LOCATION_SERVICE);
```

- In Android we can get the location in 2 ways.
 - GPS_Provider
 - Network_Provider

-lManager.getLastKnownLocation(LocationManager.PROVIDER_NAME)

- above method is used to get the last updated location.

-To get the current updated location we have configure the following listener to get the updated location.

```
- lManager.requestLocationUpdates(provider_name,
min_time(ms), min_dist(meters), new LocationListener(){

    onLocationChanged(Location l)
    {
        double lati=l.getLatitude();
        double longi=l.getLongitude();
    }
    onStatusChanged(String s, int i, Bundle bundle){ };
    onProviderEnabled(){};
    onProviderDisabled() { };
});
```

Permissions:

ACCESS_FINE_LOCATION
ACCESS_COARSE_LOCATION

Activity Main.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    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="com.example.hi.locationserviceapp.MainActivity">

    <TextView
        android:id="@+id/tv1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView"
        android:textStyle="bold"
        android:textSize="30sp"
        android:gravity="center"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginLeft="113dp"
        android:layout_marginStart="113dp"
        android:layout_marginTop="40dp" />

    <TextView
        android:id="@+id/tv2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView"
        android:gravity="center"
        android:textSize="30sp"
        android:textStyle="bold"
        android:layout_marginTop="143dp"
        android:layout_alignParentTop="true"
        android:layout_alignLeft="@+id/tv1"
        android:layout_alignStart="@+id/tv1" />

</RelativeLayout>
```

MainActivity.java:

```
package com.example.hi.locationserviceapp;
import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    TextView tv1, tv2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tv1 = (TextView) findViewById(R.id.tv1);
        tv2 = (TextView) findViewById(R.id.tv2);

        LocationManager lManager = (LocationManager)
                getSystemService(Context.LOCATION_SERVICE);
        lManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
        lManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,
                1000, 1, new LocationListener() {

            @Override
            public void onLocationChanged(Location location) {
                double lati=location.getLatitude();
                double longi=location.getLongitude();
                tv1.setText(String.valueOf(lati));
                tv2.setText(String.valueOf(longi));
            }

            @Override
            public void onStatusChanged(String s, int i, Bundle bundle) {
            }

            @Override
            public void onProviderEnabled(String s) {
            }

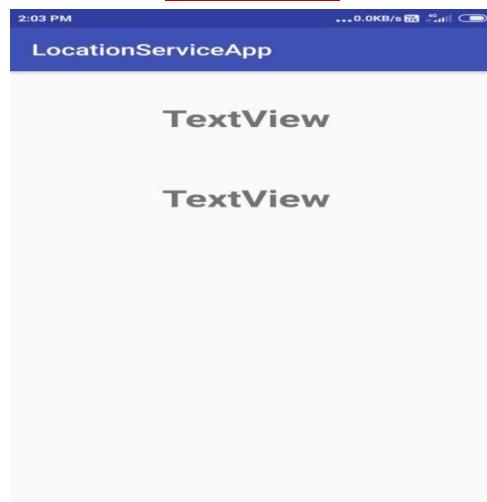
            @Override
            public void onProviderDisabled(String s) {
            }
        });
    }
}
```

```
        }
    });
}
}
```

androidManifest.xml file:

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

OUTPUT:



Notification Service:

- Every android device is having a notification bar/status bar we can display notifications on Notification bar using Notification Service.
- `getSystemService(Context.NOTIFICATION_SERVICE)` method is used to get the builtin notification service into Activity, application framework is providing a class called `NotificationManager` to manage notification service.

```
NotificationManager nManager=(NotificationManager)
    getSystemService(Context.NOTIFICATION_SERVICE);

NotificationCompact.Builder builder=new NotificationCompact.Builder(this);

builder.setTicker("SAMPLE NOTIFICATION");
builder.setSmallIcon(R.drawable.ic_beach_access_black_24dp);
builder.setContentTitle("SAMPLE NOTIFICATION");
builder.setContentText("sample notification for RGM CET");

Bitmap bmp= BitmapFactory.decodeResource(getResources(),
    R.drawable.ic_beach_access_black_24dp);
builder.setLargeIcon(bmp);
Intent i=new Intent(this,MainActivity.class);
PendingIntent pIntent=PendingIntent.getActivity(this,0,i,0);
builder.setContentIntent(pIntent);
builder.setAutoCancel(true);
//nManager.notify(1,builder.build());
nManager.notify((int)System.currentTimeMillis(),builder.build());
```

ActivityMain.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    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="com.example.hi.notificationserviceapp.MainActivity">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:text="NOTIFY"
    android:onClick="notify"/>

</RelativeLayout>
```

ManiActivity.java:

```
package com.example.hi.notificationserviceapp;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.app.NotificationCompat;
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 notify(View v){
        NotificationManager nManager=(NotificationManager)
            getSystemService(Context.NOTIFICATION_SERVICE);

        NotificationCompat.Builder builder=new NotificationCompat.Builder(this);

        builder.setTicker("SAMPLE NOTIFICATION");
        builder.setSmallIcon(R.drawable.ic_beach_access_black_24dp);
        builder.setContentTitle("SAMPLE NOTIFICATION");
        builder.setContentText("sample notification for RGM CET");

        Bitmap bmp=
        BitmapFactory.decodeResource(getResources(),R.drawable.ic_beach_access_black_24dp);

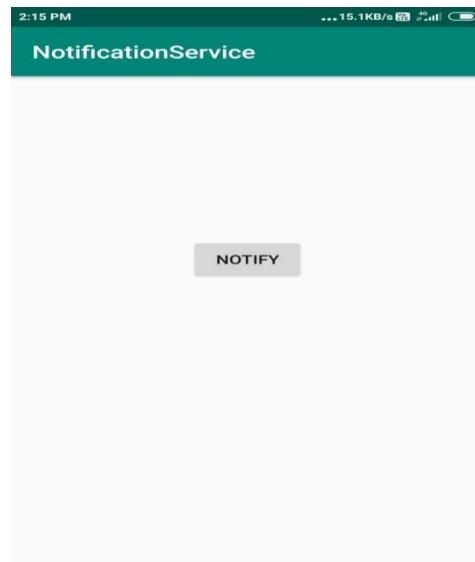
        builder.setLargeIcon(bmp);
```

```
Intent i=new Intent(this,MainActivity.class);
PendingIntent pIntent=PendingIntent.getActivity(this,0,i,0);
builder.setContentIntent(pIntent);
builder.setAutoCancel(true);
//nManager.notify(1,builder.build());
nManager.notify((int)System.currentTimeMillis(),builder.build());
```

}

}

OUTPUT:



SensorService:

- Based on the device capability Android supports different types of sensors like
 - ACCESSORY**
 - PROXIMITY**
 - ORIENTATION**
 - GYROSCOPE**
 - GRAVITY**

-`getSystemService(Context.SENSOR_SERVICE)` method is used to get the built-in sensor service into Activity, application framework is providing a class called `SensorManager` to manage SensorService.

**SensorManager sManager=(SensorManager)
getSystemService(Context.SENSOR_SERVICE);**

- To get the sensor events configure the following listener.

```
sManager.registerListener(new SensorListener(){  
    public void onSensorChanged(float[] values){  
        }  
    public void onAccuracyChanged(int accuracy){  
        }  
    }, SENSOR_TYPE);
```

Activity_Main.xml file:

```
<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout  
    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="com.example.hi.sensorserviceapp.MainActivity">  
  
<TextView  
    android:id="@+id/textView"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentTop="true"  
    android:layout_centerHorizontal="true"  
    android:layout_marginTop="117dp"  
    android:text="TextView"  
    android:textSize="25sp"/>  
  
<TextView  
    android:id="@+id/textView2"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_centerHorizontal="true"  
    android:layout_centerVertical="true"  
    android:text="TextView"  
    android:textSize="25sp"/>  
</RelativeLayout>
```

MainActivity.java file:

```

package com.example.hi.sensorserviceapp;
import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorListener;
import android.hardware.SensorManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    TextView tv1, tv2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tv1 = (TextView) findViewById(R.id.textView);
        tv2 = (TextView) findViewById(R.id.textView2);

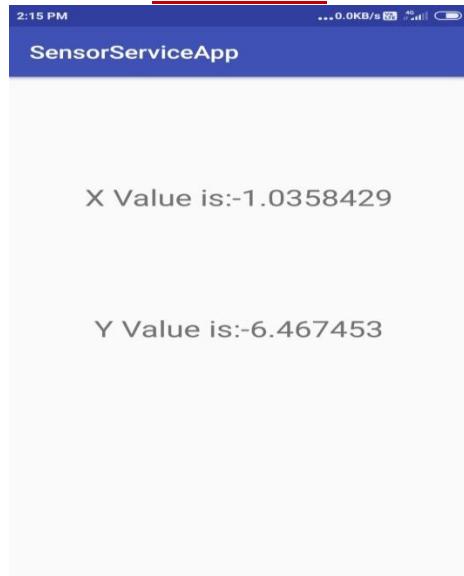
        SensorManager sManager=(SensorManager)
            getSystemService(Context.SENSOR_SERVICE);

        sManager.registerListener(new SensorListener() {
            @Override
            public void onSensorChanged(int i, float[] floats) {
                tv1.setText("X Value is:"+String.valueOf(floats[0]));
                tv2.setText("Y Value is:"+String.valueOf(floats[1]));
            }
            @Override
            public void onAccuracyChanged(int i, int i1) {

            }
        },SensorManager.SENSOR_ACCELEROMETER);
    }
}

```

OUTPUT:



Wifi Service:

- `getSystemService(Context.WIFI_SERVICE)` method is used to get the builtin wifi service into Activity, application framework is providing a class called WifiManager to manage wifi service.

```
WifiManager wManager=(WifiManager) getSystemService(Context.WIFI_SERVICE);
```

- `wManager.getWifiState()` method is used to **get the wifi state**, return type of this method is int (0-disabled, 1- disabling , 2- enabling , 3- enabled).

- `wManager.setWifiEnabled(Boolean)`method is used to **change the wifi state**.

- `wManager.getScanResults()` method is used to **get the list of available wifi devices** return type of this method is `List<ScanResult>`.

```
List<ScanResult> results=wManager.getScanResults();
for(ScanResult result:results){
    result.SSID ;           // name
    result.frequency;      // signal strength
}
```

- `wManager.getConfiguredNetworks()` method is used to **get the list of connected wifi devices**.

```
List<WifiConfiguration> list=wManager.getConfiguredNetworks();  
for(WifiConfiguration config:list)  
{  
    config.SSID; // name  
    config.status; // connected / not-connected  
}
```

permissions:

ACCESS_WIFI_STATE
CHANGE_WIFI_STATE
ACCESS_COARSE_LOCATION

Activity Main.xml file:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical"  
    tools:context="com.example.hi.wifiapp.MainActivity">  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="horizontal">  
  
<TextView  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:layout_weight="0.5"  
    android:text="WIFI"  
    android:gravity="center"  
    android:textSize="25sp"/>  
  
<Switch  
    android:id="@+id/sw_wifi"  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:layout_weight="0.5"/>  
  
</LinearLayout>
```

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

    <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="0.5"
        android:id="@+id	btn_get_wifi_device"
        android:text="GET_WIFI_DEVICE"
        android:onClick="getWifiDevice"/>

    <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="0.5"
        android:id="@+id	btn_get_paired_wifi"
        android:text="PAIRED_WIFI_DEVICE"
        android:onClick="getPairedWifiDevice"/>
</LinearLayout>

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

</LinearLayout>
```

MainActivity.java file:

```
package com.example.hi.wifiapp;
import android.content.Context;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiConfiguration;
import android.net.wifi.WifiManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.CompoundButton;
import android.widget.ListView;
import android.widget.Switch;
```

```

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

public class MainActivity extends AppCompatActivity {
    Switch sw1;
    ListView lview;
    WifiManager wManager;

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

        sw1=(Switch)findViewById(R.id.sw_wifi);
        lview=(ListView)findViewById(R.id.lview);
        wManager=(WifiManager)getApplicationContext().getSystemService(Context.
                                                WIFI_SERVICE);
        int wifi_status=wManager.getWifiState();
        if(wifi_status== 0 || wifi_status== 1){
            sw1.setChecked(false);
        }
        else if(wifi_status== 2 || wifi_status== 3){
            sw1.setChecked(true);
        }

        sw1.setOnCheckedChangeListener(new
                                    CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton buttonView,
                                         boolean isChecked) {
                if(isChecked){
                    wManager.setWifiEnabled(true);
                }
                else {
                    wManager.setWifiEnabled(false);
                }
            }
        });
    }

    public void getWifiDevice(View v){
        ArrayList<String> list=new ArrayList<>();
        ArrayAdapter adapter=new
        ArrayAdapter(this,android.R.layout.simple_list_item_single_choice, list);
        lview.setAdapter(adapter);
        List<ScanResult>sResult=wManager.getScanResults();
    }
}

```

```
for (ScanResult item :sResult){
    list.add(item.SSID +"\t\t"+ item.frequency);
    adapter.notifyDataSetChanged();
}
}

public void getPairedWifiDevice(View v){
    ArrayList<String>list=new ArrayList<>();
    ArrayAdapter adapter=new ArrayAdapter(this,
        android.R.layout.simple_list_item_single_choice, list);
    lview.setAdapter(adapter);
    List<WifiConfiguration>wifiConfigurationList=wManager.getConfiguredNetworks();
    for (WifiConfiguration item:wifiConfigurationList){
        list.add(item.SSID +"\t\t"+item.status);
        adapter.notifyDataSetChanged();
    }
}
}
```

String.xml file:

```
<resources>
    <string name="app_name">WifiApp</string>
    <string name="wifi_name">Wifi</string>
    <string name="get_wifi_device"></string>
    <string name="paired_wifi_device"></string>
</resources>
```

androidManifest.xml file:

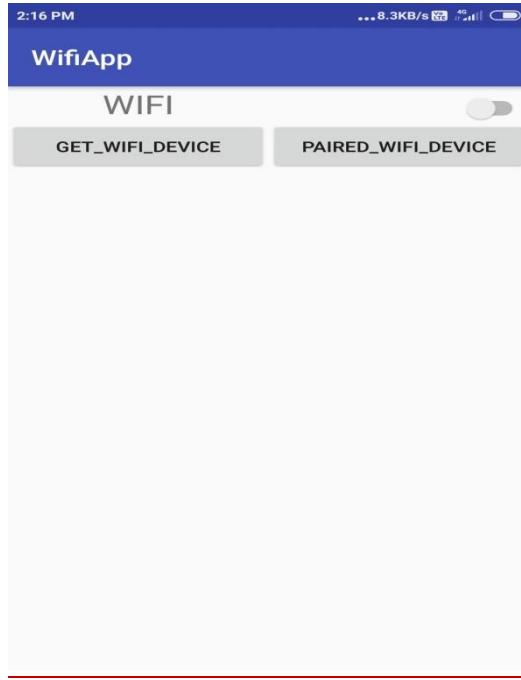
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hi.wifiapp">

    <uses-permission
        android:name="android.permission.ACCESS_WIFI_STATE"/>
    <uses-permission
        android:name="android.permission.CHANGE_WIFI_STATE"/>
    <uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
```

```
android:label="@string/app_name"
android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER">
    </intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:



Bluetooth:

In android there are different ways to manage BluetoothService one of the best approaches is manage BT service using BluetoothAdapter.

```
BluetoothAdapter bAdapter=BluetoothAdapter.getDefaultAdapter();
```

- bAdapter.isEnabled() method is used to get the BT state, return type of this method is boolean.

- bAdapter.enable() / bAdapter.disable() methods is used to change the BT state.
- bAdapter.startDiscovery() method is used to start searching for BT devices.
- to get the list of available BT devices we have to configure a Broadcast receiver.

```
IntentFilter filter=new IntentFilter();
filter.addAction(BluetoothDevice.EXTRA_FOUND);
registerReceiver(object_of_BR, filter);
```

- If any new BT device is found it will invoke onReceive() method in BR class
class MyReceiver extends BroadcastReceiver
 - {
 - onReceive(Context c, Intent data){
 - BluetoothDevice device=
 - data.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
 - device.getName();
 - device.getAddress();
 - }
 - }

permissions:

-BLUETOOTH
-BLUETOOTH_ADMIN
-ACCESS_COARSE_LOCATION

Vibrate:

-getSystemService(Context.VIBRATOR_SERVICE) method is used to get the builtin Vibrator service into Activity. Application framework is providing a class called Vibrator to manage Vibrator service.

```
Vibrator vib=(Vibrator) getSystemService(Context.VIBRATOR_SERVICE);
vib.vibrate(milli_seconds);
```

permissions:

VIBRATE

activity_Main.xml file:

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

    <Switch
        android:id="@+id/s1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="BLUETOOTH"
        android:textSize="25sp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="GETBTDEVICES"
        android:onClick="getBTDevices"
        android:layout_gravity="center"
        android:textSize="25sp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="VIBRATE"
        android:onClick="vibrate"
        android:layout_gravity="center"
        android:textSize="25sp" />

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

    </ListView>

</LinearLayout>
```

MainActivity.java file:

```

package com.example.hi.btservicesapp;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Vibrator;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.CompoundButton;
import android.widget.ListView;
import android.widget.Switch;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    Switch s1;
    ListView lview;
    BluetoothAdapter bAdapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        s1=(Switch)findViewById(R.id.s1);
        lview=(ListView)findViewById(R.id.lview);
        bAdapter=BluetoothAdapter.getDefaultAdapter();
        s1.setChecked(bAdapter.isEnabled());
        s1.setOnCheckedChangeListener(new
            CompoundButton.OnCheckedChangeListener() {
                @Override
                public void onCheckedChanged(CompoundButton
                    compoundButton, boolean b) {
                    if(b) {
                        bAdapter.enable();
                    }else{
                        bAdapter.disable();
                    }
                }
            });
    }
}

```

```
        }
    });

public void getBTDevices(View v){
    final ArrayList<String>list=new ArrayList<>();
    final ArrayAdapter<String>adapter=new ArrayAdapter<String>(this,
        android.R.layout.simple_spinner_dropdown_item,list);
    lview.setAdapter(adapter);
    bAdapter.startDiscovery();
    IntentFilter filter=new IntentFilter();
    filter.addAction(BluetoothDevice.ACTION_FOUND);
    registerReceiver(new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            BluetoothDevice
                device=intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
                list.add(device.getName()+"\n"+device.getAddress());
                adapter.notifyDataSetChanged();
            }
        },filter);
    }
}

public void vibrate(View v){
    Vibrator vib=(Vibrator)getSystemService(Context.VIBRATOR_SERVICE);
    vib.vibrate(10000);
}
```

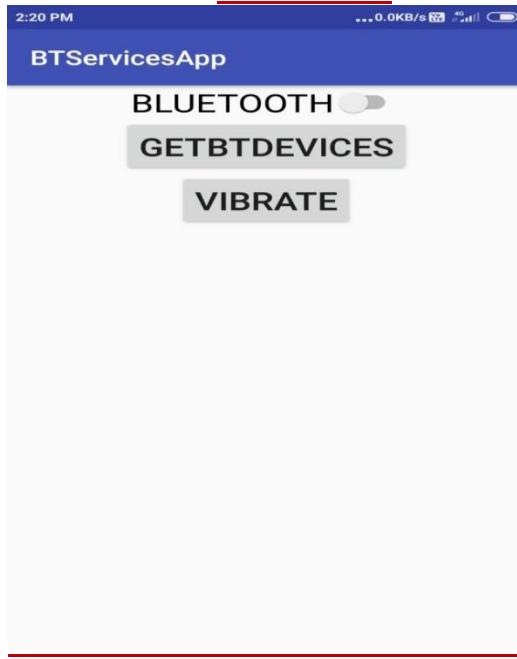
androidManifest.xml file

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hi.btservicesapp">

    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission
        android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.VIBRATE"/>

    <application
        android:allowBackup="true"
```

```
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
</manifest>
```

OUTPUT:**Broadcast Receiver:**

- Broadcast Receiver is registered to get the system announcements.
 - e.g: Charger connected/disconnected,
Screen on/off,
Making/receiving call,
Headset plugin,
Battery low.... etc..

Steps to work with Broadcast Receiver:

- 1) Create a class with subtype of android.content.BroadcastReceiver.
- 2) It is an abstract class having an abstract method called onReceive() , so provide the implementation for onReceive() method.
- 3) From Activity class for which events you want to get the broadcast announcements configure the events using IntentFilter.
(group of Intents is called as IntentFilter).

```
IntentFilter filter=new IntentFilter();
filter.addAction(Intent.ACTION_NAME);
.....
registerReceiver(obj_of_br, filter);
```

- if any one of the configured event is happened it will invoke onReceive() method in broadcast receiver.

activity_main.xml:

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

    <TextView
        android:id="@+id/tv1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TEXTVIEW"
        android:textSize="40sp"
        android:textColor="#FF0000"
        tools:layout_editor_absoluteY="-2dp"
        tools:layout_editor_absoluteX="32dp" />

</android.support.constraint.ConstraintLayout>
```

MyReceiver.java:

```
package com.example.hi.broadcastreceiverapp;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.widget.TextView;

public class MyReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        MainActivity mActivity = (MainActivity) context;
        TextView tv = (TextView) mActivity.findViewById(R.id.tv1);

        if (intent.getAction().equals(Intent.ACTION_POWER_CONNECTED)) {
            tv.setText("POWER CONNECTED");

        } else if (intent.getAction().equals(Intent.ACTION_POWER_DISCONNECTED)) {
            tv.setText("POWER DISCONNECTED");

        } else if (intent.getAction().equals(Intent.ACTION_SCREEN_ON)) {
            tv.setText("SCREEN ON");

        } else if (intent.getAction().equals(Intent.ACTION_SCREEN_OFF)) {
            tv.setText("SCREEN OFF");

        } else if (intent.getAction().equals(Intent.ACTION_AIRPLANE_MODE_CHANGED)) {
            tv.setText("AIRPLANE MODE CHANGED");

        } else if (intent.getAction().equals(Intent.ACTION_HEADSET_PLUG)) {
            tv.setText("HEADSET PLUGIN");
        }
    }
}
```

MainActivity.java:

```
package com.example.hi.broadcastreceiverapp;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
```

```
TextView tv;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    tv=(TextView)findViewById(R.id.tv1);

    IntentFilter filter=new IntentFilter();

    filter.addAction(Intent.ACTION_HEADSET_PLUG);
    filter.addAction(Intent.ACTION_POWER_CONNECTED);
    ;
    filter.addAction(Intent.ACTION_POWER_DISCONNECTED);
    filter.addAction(Intent.ACTION_SCREEN_ON);
    filter.addAction(Intent.ACTION_SCREEN_OFF);
    filter.addAction(Intent.ACTION_AIRPLANE_MODE_CHANGED);

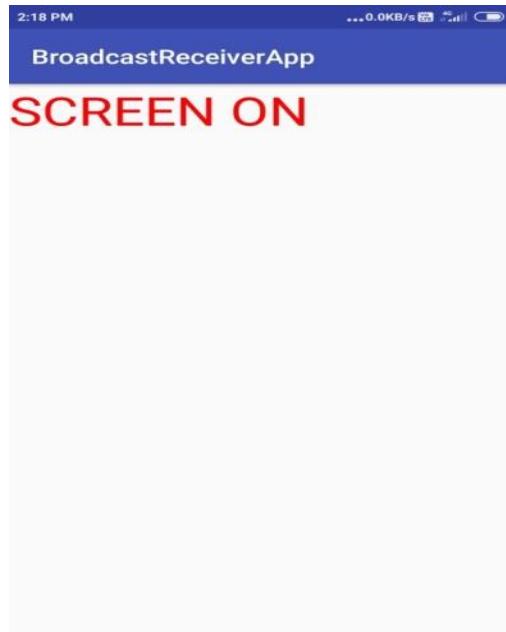
    registerReceiver(new MyReceiver(), filter);
    // registerReceiver(new MyReceiver,filter);
}
}
```

androidManifest.xml file:

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

```
</activity>
</application>
</manifest>
```

OUTPUT:



UNIT VI

UNIT-VI:

Content Provider, Dialog Boxes (Custom dialog, Alert dialog, date Picker, Time Picker, Progress dialog, dialog Fragment), Google Maps.

Content Provider:

- ContentProvider is used to share the data between multiple applications [in Android one of the security feature is we can't access the other applications data into our application but if the application is providing content provider then we can access the data into our application, in android following builtin applications are providing content provider e.g. : contacts, callog, settings, media, calendar ..]

Steps to work with Content Provider:

1) ContentResolver class is used to get the builtin content providers into Activity, getContentResolver() method is used to get the object of ContentResolver.

ContentResolver resolver=getContentResolver();

2) resolver.query(CP_URI,.....) method is used to get the data from content provider, return type of this method is Cursor.

Cursor c=resolver.query(CP_URI,.....);

3) If the data is available in a cursor for presenting the data Android is providing an adapter called SimpleCursorAdapter.

```
SimpleCursorAdapter adapter=new SimpleCursorAdapter
(context, xml_file, cursor_obj, from, to);
ui_comp.setAdapter(adapter);
```

- **from:** String[]{DB_COLUMN1 , DB_COLUMN2}
- **to:** int[]{R.id.ui_comp1, R.id.ui_comp2}

permissions:

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

activity_Main.xml file:

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

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

Indiview.xml file:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical">  
  
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="NAME"  
    android:id="@+id/tv1"  
    android:textSize="25sp"  
    android:textStyle="bold"  
    android:textColor="#FF0000"  
    android:gravity="center"/>  
  
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="NUMBER"  
    android:id="@+id/tv2"  
    android:textSize="25sp"  
    android:textStyle="bold"  
    android:textColor="#F0FF"  
    android:gravity="center"/>  
  
</LinearLayout>
```

MainActivity.java file:

```

package com.example.hi.contentproviderapp;
import android.content.ContentResolver;
import android.database.Cursor;
import android.provider.ContactsContract;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;

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

        ContentResolver resolver=getContentResolver();
        int[] to=new int[]{R.id.tv1,R.id.tv2};
        //Cursor
        c=resolver.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,null,null,null,null);
        //you can test first above statement and second below statement
        Cursor c=resolver.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI, null, null,
                               null, ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME);

        String[] from=new String[]{ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME,
                               ContactsContract.CommonDataKinds.Phone.NUMBER};

        SimpleCursorAdapter adapter=new SimpleCursorAdapter(this, R.layout.indiview, c, from, to);

        lview.setAdapter(adapter);
    }
}

```

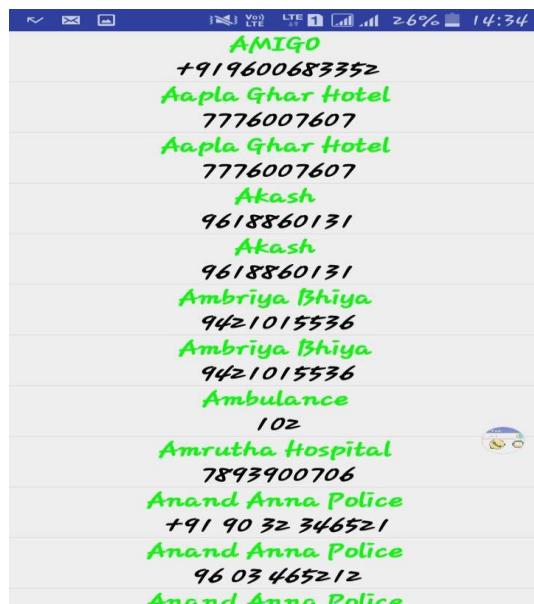
androidManifest.xml file:

```

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

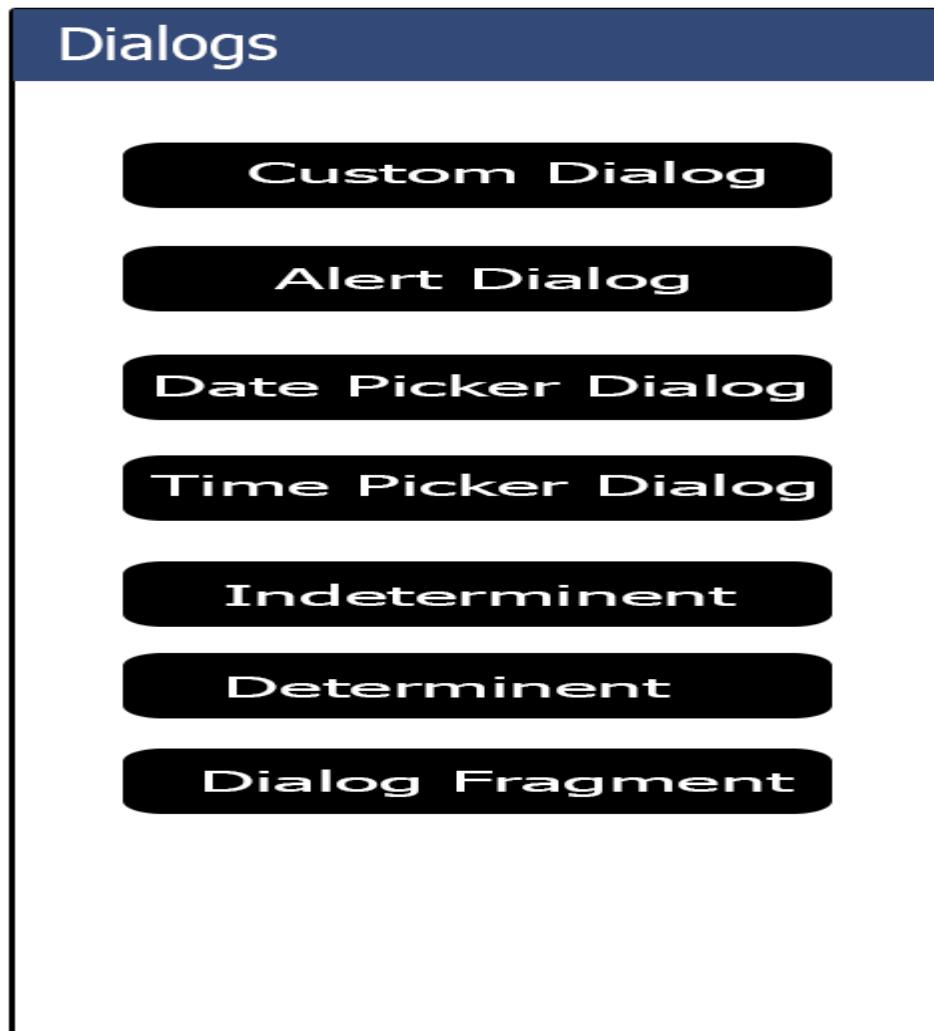
</manifest>

```

OUTPUT:

Dialogs:

- Custom Dialog
- Alert Dialog
- DatePicker
- TimePicker
- Progress Dialog [Indeterminent / determininent]



- **android.app.Dialog class is the parent for all the dialogs in Android.**

Custom Dialog: - By using custom dialog we can create our own UI and we can display the UI on dialog.

Syntax:

```
Dialog d=new Dialog(activity_object);
d.setContentView(R.layout.xml_file);
d.show();
```

- If the UI component is available on Dialog XML, we can't configure the events (e.g. : onClick) using XML, we can configure the events only by using Java.
- To get the UI component from dialog XML, use dialog object before findViewById() method.

```
Button b=(Button)d.findViewById(R.id.id_name);
```

AlertDialog:

- android.app.AlertDialog is a child of android.app.Dialog.
- AlertDialog is one of the builtin dialog, with few builtin functionalities like title, icon, message and max we can create 3 possible buttons [positive, negative, neutral].

AlertDialog.Builder builder=new AlertDialog.Builder(this);

```
builder.setTitle("title_here");
builder.setIcon(R.drawable.image_name);
builder.setMessage("message_here");
builder.setPositiveButton("text", listener);
builder.setNegativeButton("text", listener);
builder.setNeutralButton("text", listener);
builder.show();
```

-if we click any one of the above buttons it will invoke the following listener.

```
DialogInterface.OnClickListener listener=new DialogInterface.OnClickListener(){
    public void onClick(DialogInterface di, int what)
    {
        // what will tell the button type.
    }
};
```

DatePickerDialog:

- DatePickerDialog is a child of android.app.Dialog.
- DatePickerDialog is used to get date as input from the user.

```
DatePickerDialog dpd=new DatePickerDialog(context, listener, year, month, day);
dpd.show();
```

-If we change the date it will invoke the following listener.

```
DatePickerDialog.OnDateSetListener listener=new
                               DatePickerDialog.OnDateSetListener() {
    public void onDateChanged(int year, int month, int day)
    {
        // logic when date changed..
    }
}
```

TimePickerDialog:

- TimePickerDialog is a child of android.app.Dialog.
- TimePickerDialog is used to get time as input from the user.

```
TimePickerDialog tpd=new TimePickerDialog(context, listener, hour, minute,
                                         24hrs_format(Boolean));
tpd.show();
```

-If we change the time it will invoke the following listener.

```
TimePickerDialog.OnTimeSetListener listener=new
                               TimePickerDialog.OnTimeSetListener() {
    public void onTimeChanged(int hour,int minute)
    {
        // logic when time changed..
    }
}
```

ProgressDialog:

There are 2 types of Progress dialogs.

- Indeterminent**
- Determinant**

- if you can't calculate the time, how much time it will take to complete a specific task use Indeterminent progress dialog, if you can calculate the time use determinant progress dialog.

```
ProgressDialog pDialog=new ProgressDialog(context);
pDialog.setTitle("title_here");
pDialog.setMessage("message_here");
pDialog.setIcon(R.drawable.image_name);
pDialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Indeterminent
pDialog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL); //Determinent
pDialog.show();
```

activity_Main.xml file:

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

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="CUSTOM DIALOG"
        android:onClick="CustomDailog"
        android:layout_gravity="center"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="ALERT DIALOG"
        android:onClick="AlertDialog"
        android:layout_gravity="center"/>

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

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Date Picker Dailog"
            android:onClick="datepickerdailog"/>

        <EditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
    android:id="@+id/datepicker"
    android:editable="false"/>
</LinearLayout>

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

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Time Picker Dailog"
        android:onClick="timepickerdialog"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/timepicker"
        android:editable="false"/>
</LinearLayout>

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Indeterminant Progress Dailog"
    android:onClick="indeterminent"
    android:layout_gravity="center"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Determinant Progress dailog"
        android:onClick="determinent"
        android:layout_gravity="center"/>
</LinearLayout>
```

customDialog.xml:

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

```
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Are You Sure To Exit"
        android:textSize="25sp"/>

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

        <Button
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.2"
            android:id="@+id	btn_yes"
            android:text="Yes"/>

        <TextView
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.6"/>

        <Button
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.2"
            android:text="No"
            android:id="@+id	btn_no"/>
    </LinearLayout>

</LinearLayout>
```

MainActivity.java

```
package com.example.hi.dialogapp;
import android.app.DatePickerDialog;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.app.TimePickerDialog;
import android.content.Context;
import android.icu.util.Calendar;
```

```
import android.os.Vibrator;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TimePicker;
import java.sql.Time;

public class MainActivity extends AppCompatActivity {

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

    public void CustomDialog(View view){
        final Dialog d=new Dialog(MainActivity.this);
        d.setTitle("Message");
        d.setContentView(R.layout.customdialog);
        d.show();

        Button yes_btn=(Button)d.findViewById(R.id.btn_yes);
        Button no_btn=(Button)d.findViewById(R.id.btn_no);

        yes_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                d.dismiss();
                System.exit(0);
            }
        });

        no_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                d.dismiss();
            }
        });
    }

    public void AlertDialog(View view){
        AlertDialog.Builder builder=new AlertDialog.Builder(this);
    }
}
```

```

builder.setTitle("Message");
builder.setMessage("Are You Sure To Exit");
builder.setIcon(R.mipmap.ic_launcher_round);
AlertDialog.OnClickListener listener= new AlertDialog.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int which) {
        if(which== dialogInterface.BUTTON_POSITIVE){
            dialogInterface.dismiss();
            finish();
        }else if(which== dialogInterface.BUTTON_NEGATIVE) {
            dialogInterface.dismiss();
        }
    }
};

builder.setPositiveButton("YES",listener);
builder.setNegativeButton("NO",listener);
builder.show();

}

public void datepickerdialog(View view) {
DatePickerDialog.OnDateSetListener listener= new
    DatePickerDialog.OnDateSetListener() {
        @Override
        public void onDateSet(DatePicker datePicker, int year,
                             int monthofyear, int dayofmonth) {

            EditText et=(EditText)findViewById(R.id.datepicker);
            et.setText(dayofmonth + "-" + monthofyear + "-" + year);
        }
    };
}

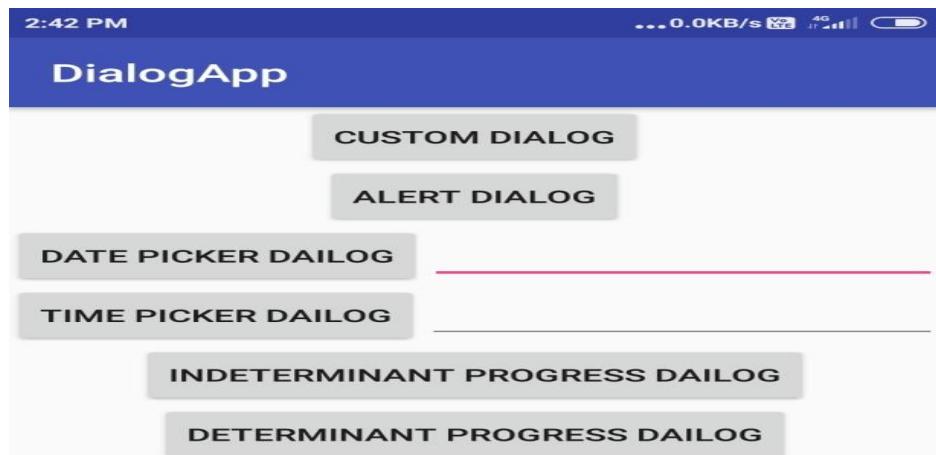
DatePickerDialog dpd=new
    DatePickerDialog(MainActivity.this, listener,2020,4,20);
dpd.show();
}

public void timepickerdialog(View view){
    java.util.Calendar cal=java.util.Calendar.getInstance();
    final int hour=cal.get(java.util.Calendar.HOUR);
    final int min=cal.get(java.util.Calendar.MINUTE);
    TimePickerDialog tpd=new TimePickerDialog(this, new
        TimePickerDialog.OnTimeSetListener() {
            @Override
            public void onTimeSet(TimePicker timePicker, int i, int i1) {
                EditText editText2=(EditText)findViewById(R.id.timepicker);
                editText2.setText(i + ":" + i1 );
            }
        });
}

```

```
        }  
    },hour,min,false);  
    tpd.show();  
}  
  
public void indeterminent(View view){  
    ProgressDialog pDailog=new ProgressDialog(MainActivity.this);  
    pDailog.setTitle("Message");  
    pDailog.setMessage("Please Wait while page is loading");  
    pDailog.setIcon(R.mipmap.ic_launcher_round);  
    pDailog.setProgressStyle(ProgressDialog.STYLE_SPINNER);  
    pDailog.show();  
}  
  
public void determinant(View view){  
    ProgressDialog pDailog=new ProgressDialog(MainActivity.this);  
    pDailog.setTitle("Message");  
    pDailog.setMessage("Downloading");  
    pDailog.setIcon(R.mipmap.ic_launcher_round);  
    pDailog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);  
    pDailog.show();  
}  
}
```

OUTPUT:



Google Maps:

- 1) Create a project, add google-play-services:maps as a library project.

Right click APP folder → open module Setting → dependency, select(+) symbol.

Click on + symbol and choose (Library dependency) select
“com.google.android.gms:play-service-maps:16.1.0”

- 2) Create a fragment UI component in Activity xml with the following name.

```
<fragment  
    android:name="com.google.android.gms.maps.SupportMapFragment"  
...../>
```

- 3) Use the following code in Activity to get the SupportMapFragment into Activity.

```
SupportMapFragment frag=(SupportMapFragment)  
        getSupportFragmentManager().findFragmentById(R.id.XXX);
```

- 4) Get the GoogleMap object from SupportMapFragment.

```
frag.getMapAsync(new OnMapReadyCallback() {  
    @Override  
    public void onMapReady(GoogleMap googleMap) {  
        //write the logic, what you want to perform on GoogleMap  
    }  
});
```

- 5) To work with any Google-API we have to get an API key from Google, go through the following URL to get an API key.

<http://code.google.com/apis/console>
API Key:

Example Like:

AIzaSyAOeIcUosQIJDFuZNCU0TkA-oQNWSfeZg

AIzaSyAXQpjHUvOxg97SutBJN2itPpcCBd7IwkY

- 6) Configure the API in manifest.xml with the following tag inside <application> tag.

```
    <meta-data  
        android:name=""  
        android:value="AIzaSyAOeIcUosQIJDFuZNCU0TkA-oQNWSfeZg" />
```

- 7) Set the following method to GoogleMap to change the Map style.
`googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);`

- 8) Use the following code to place a marker (location) on a Map.

```
MarkerOptions mOption=new MarkerOptions();
mOption.position(new LatLng(lati, longi));
googleMap.addMarker(mOption);
```

- 9) Use the following method to apply Zoom & move the goole map to a specific location.

```
googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(new
    LatLng(lati, longi), 15f));
```

- 10) Use the following code to customize the icon, set the title
`mOption.icon (BitmapDescriptorFactory.fromResource(R.drawable.car));`
`mOption.title("title here");`

Activity_main.xml file:

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

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <fragment
        android:id="@+id/fragment"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.9"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="0.1"
        android:orientation="horizontal">
```

```
<TextView  
    android:id="@+id/tv_latitude"  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.5"  
    android:text="Latitude"  
    android:gravity="center"/>  
  
<TextView  
    android:id="@+id/tv_longitude"  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="0.5"  
    android:text="Longitude"  
    android:gravity="center"/>  
  
</LinearLayout>  
  
</LinearLayout>
```

MainActivity.java:

```
package com.example.hi.googlemaptest;  
import android.Manifest;  
import android.content.Context;  
import android.content.pm.PackageManager;  
import android.location.Location;  
import android.location.LocationListener;  
import android.location.LocationManager;  
import android.support.v4.app.ActivityCompat;  
import android.support.v4.content.ContextCompat;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.widget.TextView;  
import com.google.android.gms.maps.CameraUpdateFactory;  
import com.google.android.gms.maps.GoogleMap;  
import com.google.android.gms.maps.OnMapReadyCallback;  
import com.google.android.gms.maps.SupportMapFragment;  
import com.google.android.gms.maps.model.BitmapDescriptorFactory;  
import com.google.android.gms.maps.model.LatLng;  
import com.google.android.gms.maps.model.MarkerOptions;  
import org.w3c.dom.Text;  
public class MainActivity extends AppCompatActivity {  
    SupportMapFragment smFragment;  
    TextView latitude, longitude;
```

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

    latitude=(TextView)findViewById(R.id.tv_latitude);
    longitude=(TextView)findViewById(R.id.tv_longitude);

    int coarse_loc_status=
        ContextCompat.checkSelfPermission(MainActivity.this,
            Manifest.permission.ACCESS_COARSE_LOCATION);

    int fine_loc_status=ContextCompat.checkSelfPermission(MainActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION);

    if(coarse_loc_status== PackageManager.PERMISSION_GRANTED &&
        fine_loc_status==PackageManager.PERMISSION_GRANTED) {

    }
    else {
        ActivityCompat.requestPermissions(MainActivity.this,
            new String[] {Manifest.permission.ACCESS_COARSE_LOCATION,
                Manifest.permission.ACCESS_FINE_LOCATION}, 123);
    }

    smFragment=(SupportMapFragment) getSupportFragmentManager().
        findFragmentById(R.id.fragment);

    smFragment.getMapAsync(new OnMapReadyCallback() {
        @Override
        public void onMapReady(final GoogleMap googleMap) {
            final LocationManager lManager=(LocationManager)
                getSystemService(Context.LOCATION_SERVICE);

            lManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

            lManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,
                1000, 1, new LocationListener() {
                    @Override
                    public void onLocationChanged(Location location) {
                        double lati=location.getLatitude();
                        double longi=location.getLongitude();
                        latitude.setText(String.valueOf(lati));
                        longitude.setText(String.valueOf(longi));
                    }
                });
        }
    });
}
```

```
MarkerOptions mOption=new MarkerOptions();
    mOption.position(new LatLng(lati, longi));

mOption.icon(BitmapDescriptorFactory.fromResource(R.drawable.car));
    mOption.title("Simha-9000666090");
    googleMap.addMarker(mOption);

googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(new
    LatLng(lati, longi),15f));

// googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);

}

@Override
public void onStatusChanged(String provider, int status, Bundle extras) {

}

@Override
public void onProviderEnabled(String provider) {

}

@Override
public void onProviderDisabled(String provider) {

    });

    });
}
}
```

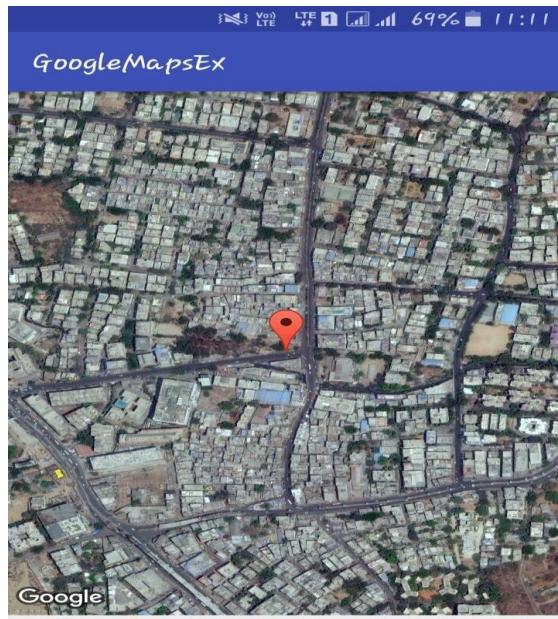
androidManifest.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.developer.googlemaptest">
    <uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission
```

```
    android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.INTERNET"/>
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category
                android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <meta-data
        android:name="com.google.android.geo.API_KEY"
        android:value="AIzaSyC3nGvEpq5bBVvj7ozUYx1blxFFAgY2W3Y"/>
</application>

</manifest>
```

OUTPUT:



***** **THE END** *****
***** &&&&& *****