**NATIONAL INFORMATICS CENTRE**

ANDROID DEVELOPMENT

**Training Material**

**National Informatics Centre, Kerala State Centre**

**Training at Guawahati, Assam**

**4th to 8th June 2018**

The materials included in this document are collected from various sources including Internet and is consolidated only for the purpose of capcity building of NIC officials, which should not be copied, reproduced in any format including electronic medium without the knowledge and written consent of National Informatics Centre, Kerala State Centre. This document is for internal circulation in NIC only.

Approved By : T Mohana Dhas, SIO, NIC Kerala

Content Provided By : T Mohana Dhas, Sr.TD & SIO

Manoj P A, TD

Andrews Varghese, TD

Vipin Bose, SSA

Syam Krishna B G, Scientific Officer

Edited by : Manoj P A

**Table of Contents**

1. **Introduction to Mobile App Development**
2. **Application Architecture**
3. **UI Design**
4. **Permissions & Shared Preferences**
5. **Advanced UI Design**
6. **Intoduction to Webservice with Android App**
7. **Secure Web APIs for login based Mobile Apps**
8. **Deployment in Goole play Store**
9. **Push Notification**
10. **SQLite**
11. **Loading Map**
12. **Progressive Web Applications**
13. **Mobile App Security**
14. **Best Practices**

**Introduction to Mobile App Development**

Mobile application Development is a term used to denote the act or process by which application software is developed for mobile devices, such as personal digital assistance, enterprise digital assistance or mobile phones. These applications can be pre-installed on phones during manufacturing platforms, or delivered as web applications using server-side or client-side processing (e.g., JavaScript) to provide an "application-like" experience within a [Web browser.](https://en.wikipedia.org/wiki/Web_browser) Application software developers also must consider a long array of screen sizes, hardware specifications, and configurations because of intense competition in mobile software and changes within each of the platforms. Mobile app development has been steadily growing, and we can’t think of a software solution without a mobile interface nowadays.

**About Android**

Android is a [mobile operating system](https://en.wikipedia.org/wiki/Mobile_operating_system) developed by [Google,](https://en.wikipedia.org/wiki/Google) based on the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) and designed primarily for [touch screen](https://en.wikipedia.org/wiki/Touchscreen) mobile devices such as [smart phones](https://en.wikipedia.org/wiki/Smartphone) and [tablets.](https://en.wikipedia.org/wiki/Tablet_computer) The [user interface](https://en.wikipedia.org/wiki/User_interface) is mainly based on [direct manipulation](https://en.wikipedia.org/wiki/Direct_manipulation_interface) using touch gestures – such as swiping, tapping and pinching – to manipulate on-screen objects, along with a [virtualkeyboard](https://en.wikipedia.org/wiki/Virtual_keyboard) for text input. Google has further developed [Android TV](https://en.wikipedia.org/wiki/Android_TV) for televisions, [Android Auto](https://en.wikipedia.org/wiki/Android_Auto) for cars and [Android Wear](https://en.wikipedia.org/wiki/Android_Wear) for wrist watches, each with specialized user interfaces. Android variants are also used on [notebooks,](https://en.wikipedia.org/wiki/Laptop) [game consoles,](https://en.wikipedia.org/wiki/Video_game_console) [digital cameras,](https://en.wikipedia.org/wiki/Digital_camera) and other electronic devices.

**Android API Levels (Versions and Version Name)**



**Android Editors and IDEs**

The Integrated Development Environment (IDE) is basically software that allows you to create custom software, and here we discuss about IDEs for mobile application.

IDEs, as its name indicates, normally contain an editor with assistance to create your codes, compiler, debugging console, and simulators with a graphical user interace. This is where we do most of the app development work such as; creating, designing, testing and finetuning your mobile app.

We will discuss here a few IDE tools used for development of mobile applications; native as well as hybrid.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Android Development IDEs* |  |  |
| o [Android Studio](https://developer.android.com/studio/index.html) | o [Eclipse](https://eclipse.org/downloads/) | o [Ionic](https://www.jetbrains.com/idea/features/android.html) |
| o [DroidEdit](http://www.droidedit.com/) | o [Android-IDE](http://www.android-ide.com/) | o [Cordova](https://cordova.apache.org/) |
| o [Corona](http://coronalabs.com/) | o [Titanium](http://www.appcelerator.com/titanium/titanium-sdk/) | o [Xamarin](https://www.xamarin.com/) |
| o [Sencha Touch](http://www.cppdroid.info/) |  |  |

During this training programme, we shall be using the IDE Android Studio for app development learning process.

*Android Language Resources*

There are number of languages you could use to develop Android applications, but the best is Java, which is the base for app development. Many languages are used to develop Android as well as hybrid applications. You can develop mobile applications using for C and C++ also with the [Android Native Development Kit](https://developer.android.com/tools/sdk/ndk/index.html) to create Android apps, but while this always means an increased app complexity, it doesn’t always have the pay off of better performance. The recent development in Android world is that Kotlin has been announced as one of the language supported in Android Studio for App development.

*Basic system requirements for working in Android Studio*

*Microsoft Windows*

* Microsoft® Windows® 7/8/10 (32- or 64-bit)
* 3 GB RAM minimum (8 GB RAM recommended); plus 1 GB for the Android Emulator.
* 2 GB of available disk space minimum (4 GB Recommended); 500 MB for IDE + 1.5 GB for Android SDK and emulator system image.
* 1280 x 800 minimum Display Screen resolution.

* For accelerated emulator: Intel® processor with support for Intel® VT-x, Intel® EM64T (Intel® 64), and Execute Disable (XD) Bit functionality

*Apple Mac*

* Mac® OS X® 10.10 (Yosemite) or higher, up to 10.12 (macOS Sierra)
* 3 GB RAM minimum (8 GB RAM recommended); plus 1 GB for the Android Emulator
* 2 GB of available disk space minimum (4 GB Recommended); 500 MB for IDE + 1.5 GB for Android SDK and emulator system image.
* 1280 x 800 minimum Display screen resolution

*Linux*

* GNOME or KDE desktop
* Tested on Ubuntu® 14.04 LTS, Trusty Tahr (64-bit distribution capable of running 32-bit applications)
* 64-bit distribution capable of running 32-bit applications
* GNU C Library (glibc) 2.19 or later
* 3 GB RAM minimum (8 GB RAM recommended); plus 1 GB for the Android Emulator.
* 2 GB of available disk space minimum (4 GB Recommended); 500 MB for IDE + 1.5 GB for Android SDK and emulator system image.
* 1280 x 800 minimum Display screen resolution
* For accelerated emulator: Intel® processor with support for Intel® VT-x, Intel® EM64T (Intel® 64), and Execute Disable (XD) Bit functionality, or AMD processor with support for AMD Virtualization™

(AMD-V™)

**JVM vs DVM**

|  |  |  |
| --- | --- | --- |
|  | DVM | JVM |
| Architecture | Register | Stack |
| OS Support | Android | Multiple |
| Re- Tools | Few | Many |
| Executables | APK | JAR |
| Constant-Pool | Per Application | Per Class |
| Memory | Less | More |

**Android Architecture**

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



## Linux kernel

At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches. This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

## Libraries

On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

## Android Libraries

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows −

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

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

## Android Runtime

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

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

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

## Application Framework

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

The Android framework includes the following key services −

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

## Applications

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

**Application Structure**

**Android Core Building Blocks**

An android **component** is simply a piece of code that has a well defined life cycle e.g. Activity,



Receiver, Service etc.

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.

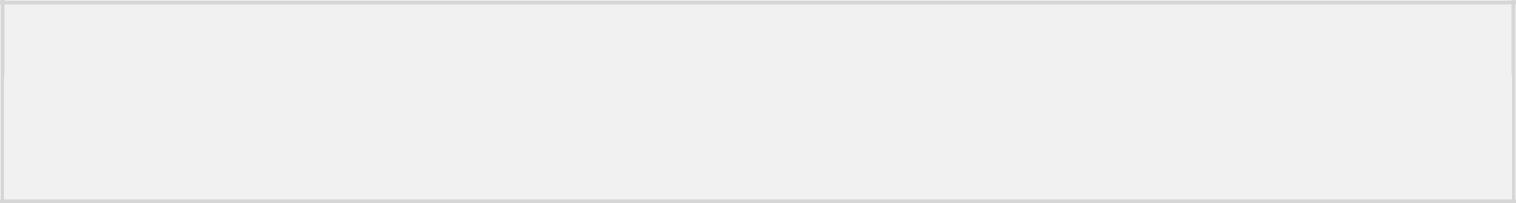
*Intent*

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.

***National Informatics Centre***

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



Intent intent=new Intent(Intent.ACTION\_VIEW); intent.setData(Uri.parse("http://www.javatpoint.com")); startActivity(intent);

*Service*

Service is a background process 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*

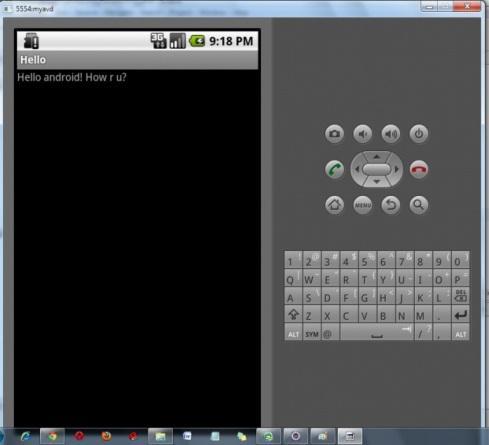
Content Providers are used to share data between the applications.

*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.

**Android Virtual Device (AVD)**

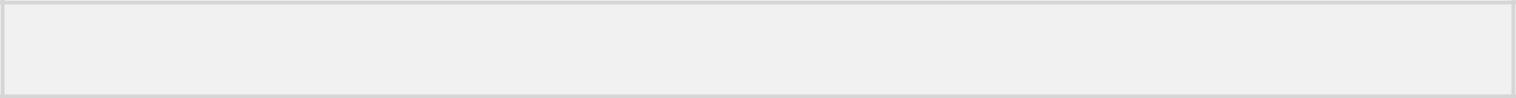
It is used to test the android application without a mobile phone, tablet or any such device. It can be created in different configurations to emulate different types of real devices.

***Android Emulator***

**Android Emulator** is used to run, debug and test the androidapplication. If you don't have the real device, it is be the best way to run, debug and test the application.

It uses an open source processor emulator technology called QEMU (Quick Emulator).

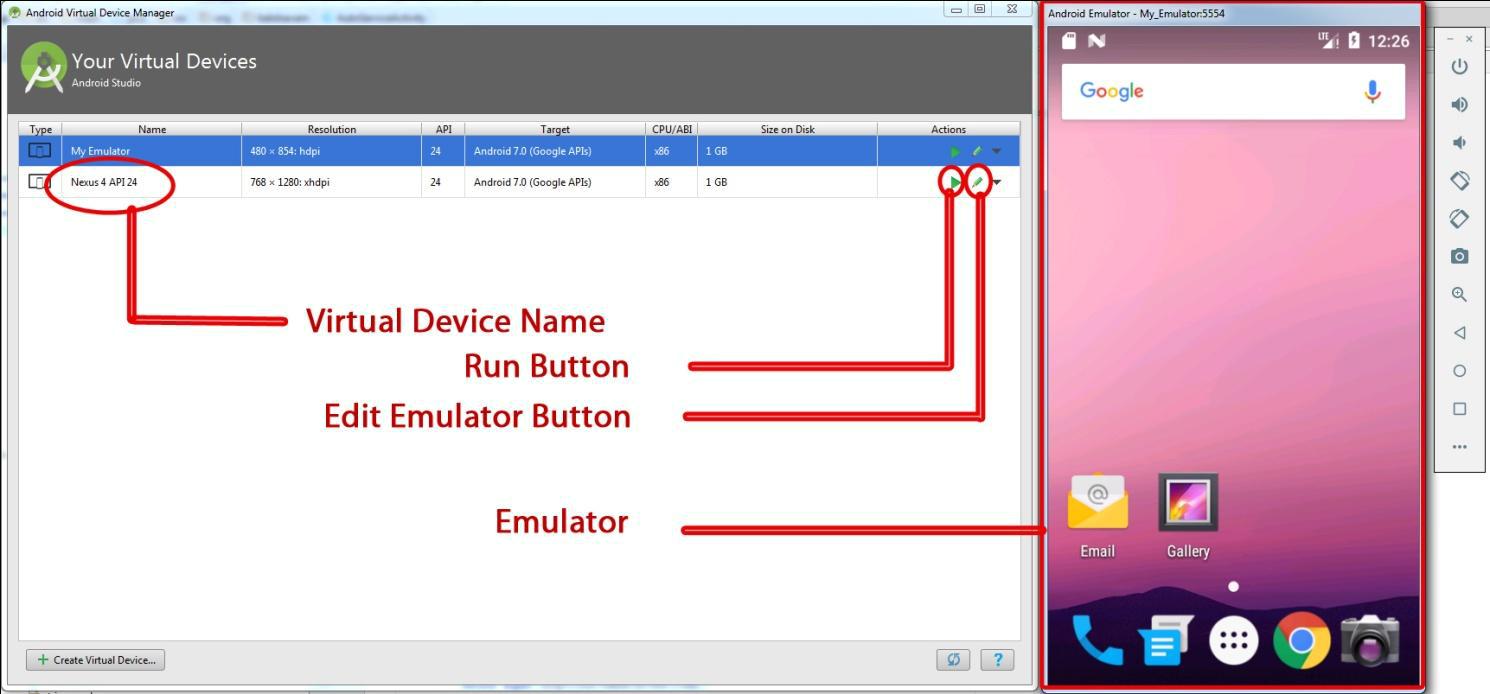
The emulator tool enables you to start the emulator from the command line. You need to write:



emulator -avd <AVD NAME>

In case of Eclipse IDE, you can create AVD by Window menu > AVD Manager > New.

The image below shows a typical android emulator;



**Build.gradle Class**

In Android Studio, Gradle is a custom build tool used to build android packages (apk files) by managing dependencies and providing custom build logic. APK file generally contains Byte code and Resources such as images, UI, xml etc.

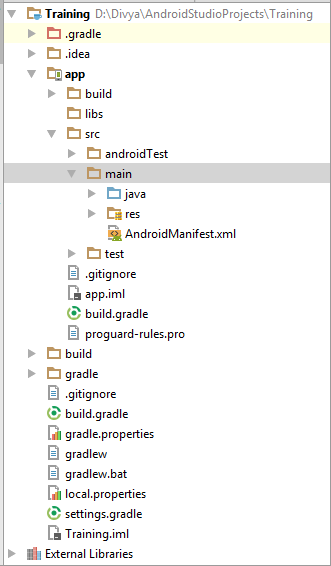
Usually, a Java project has a version and a target JRE on which it is compiled. The version and sourceCompatibility property and the other dependencies can be set in the build.gradle file.

apply **plugin**: **'com.android.application'**  
  
android {  
compileSdkVersion 25  
buildToolsVersion **"25.0.2"**  
defaultConfig {  
applicationId **"cs.app.nic.com.seminarclass"**  
minSdkVersion 15  
targetSdkVersion 25  
versionCode 1  
versionName **"1.0"**  
testInstrumentationRunner **"android.support.test.runner.AndroidJUnitRunner"**  
}  
buildTypes {  
release {  
minifyEnabled **false**  
proguardFiles getDefaultProguardFile(**'proguard-android.txt'**), **'proguard-rules.pro'**  
}  
}

packagingOptions{  
  
  
exclude **'META-INF/DEPENDENCIES.txt'**  
exclude **'META-INF/LICENSE.txt'**  
exclude **'META-INF/NOTICE.txt'**  
exclude **'META-INF/NOTICE'**  
exclude **'META-INF/LICENSE'**  
exclude **'META-INF/DEPENDENCIES'**  
exclude **'META-INF/notice.txt'**  
exclude **'META-INF/license.txt'**  
exclude **'META-INF/dependencies.txt'**  
  
}  
}  
  
dependencies {  
compile fileTree(include: [**'\*.jar'**], dir: **'libs'**)  
androidTestCompile(**'com.android.support.test.espresso:espresso-core:2.2.2'**, {  
exclude group: **'com.android.support'**, module: **'support-annotations'**  
})  
compile **'com.android.support:appcompat-v7:25.3.1'**  
compile **'com.android.support.constraint:constraint-layout:1.0.2'**  
compile **'com.android.support:design:25.3.1'**  
testCompile **'junit:junit:4.12'**  
compile files(**'libs/commons-codec-1.9.jar'**)  
compile files(**'libs/commons-logging-1.2.jar'**)  
compile files(**'libs/httpclient-4.5.2.jar'**)  
compile files(**'libs/httpcore-4.4.4.jar'**)  
compile **'com.google.android.gms:play-services:6.5.87'**  
}

The image below provides a general view of a project with the important components to be taken care during our learning programme.

Appname

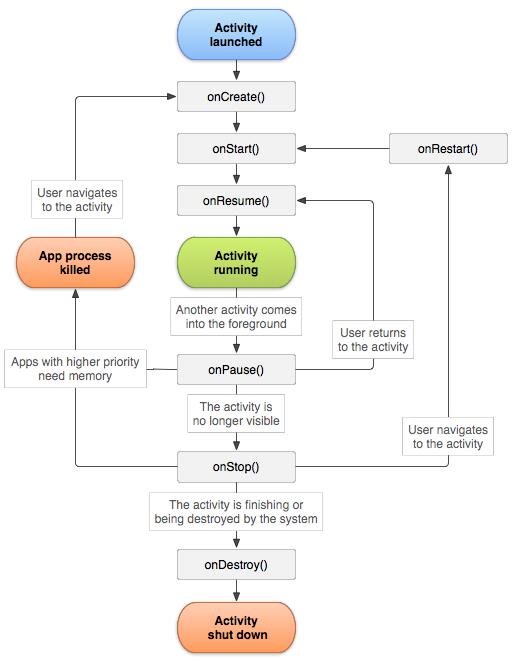


contains layout,drawable,values folders

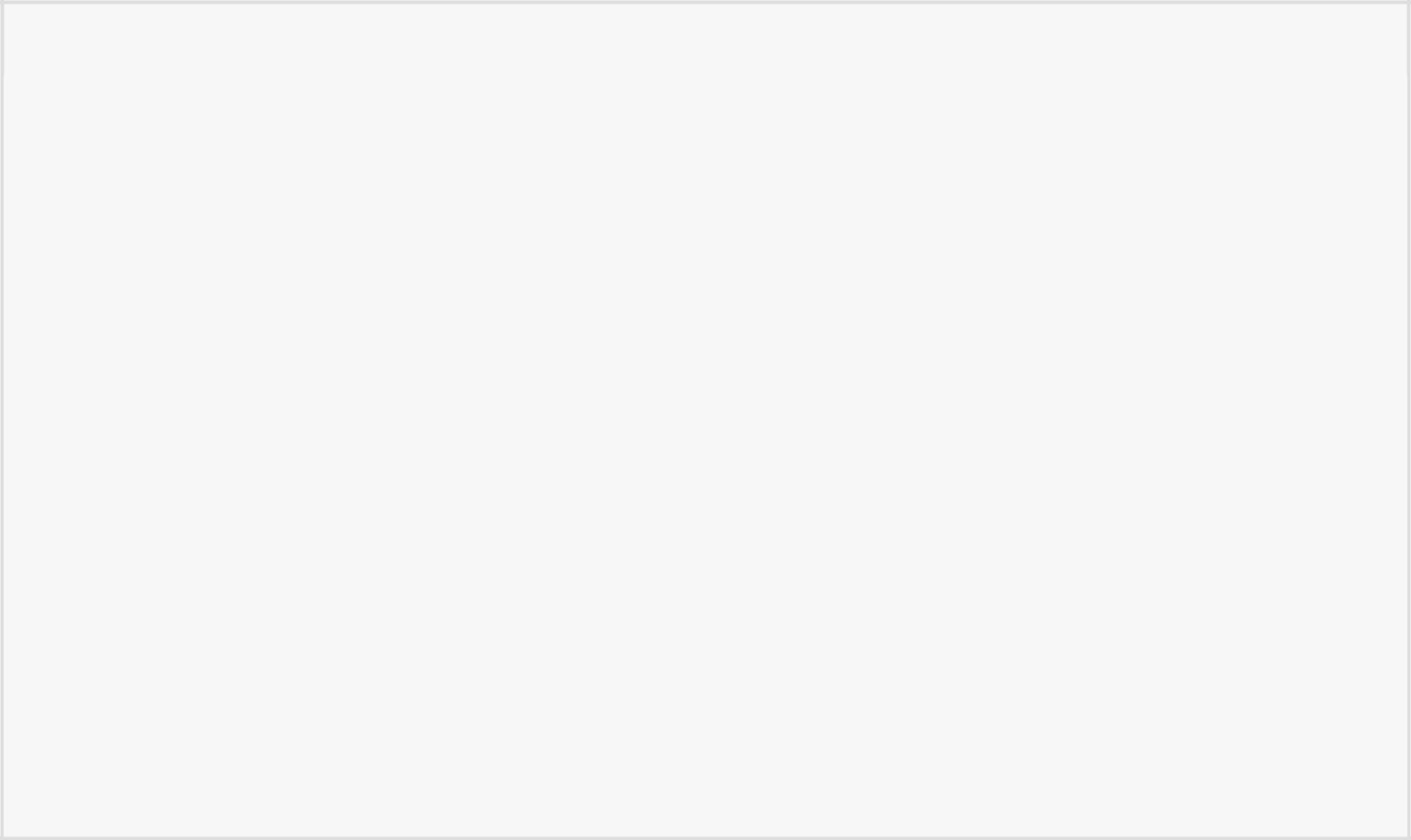
build.gradle

Android Manifest.xml

**Android Activity Life Cycle**

****

To navigate transitions between stages of the activity lifecycle, the Activity class provides a core set of six callbacks: [onCreate(),](https://developer.android.com/reference/android/app/Activity.html#onCreate(android.os.Bundle)) [onStart()](https://developer.android.com/reference/android/app/Activity.html#onStart()) , [onResume(),](https://developer.android.com/reference/android/app/Activity.html#onResume()) [onPause(),](https://developer.android.com/reference/android/app/Activity.html#onPause()) [onStop()](https://developer.android.com/reference/android/app/Activity.html#onStop()) and [onDestroy()](https://developer.android.com/reference/android/app/Activity.html#onDestroy()). The system invokes each of these callbacks as an activity enters a new state.

@Override

public void onCreate(BundlesavedInstanceState) {

// call the super class onCreate to complete the creation of activity

like

// the view hierarchy super.onCreate(savedInstanceState);

* recovering the instance state if (savedInstanceState!= null) {

mGameState = savedInstanceState.getString(GAME\_STATE\_KEY);

}

* set the user interface layout for this Activity
* the layout file is defined in the project

res/layout/main\_activity.xml file setContentView(R.layout.main\_activity);

// initialize member TextView so we can manipulate it later mTextView = (TextView) findViewById(R.id.text\_view);

}

**How to install Android Studio**

Download Android Studio exe file :

android-studio-bundle-162.3871768-windows from

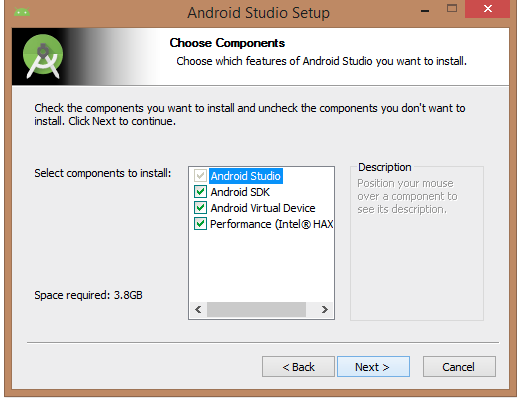
https://developer.android.com/studio/index.html

On clicking the set up file it shows as such

:

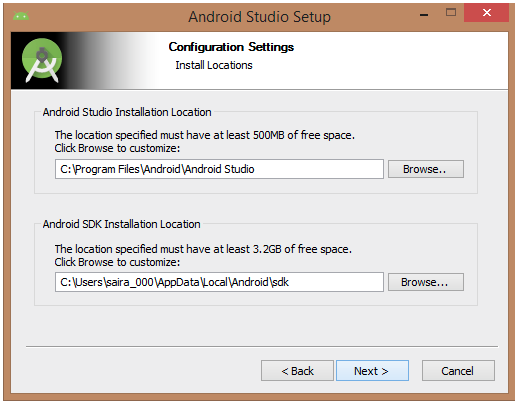


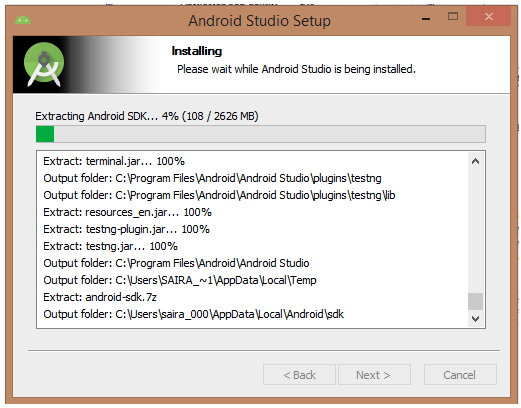
Click next to continue



Select the marked ones as shown in above figure. Click next to proceed.

Select the path for sdk and android studio to be installed.





Once the installation is finished the following screen will be seen:



**APK Files**

Android Package Kit (APK) is the package file format used to distribute and install application software and middleware onto Android operating system. APK files are ZIP file formatted packages based on the JAR file format, with .apk file extensions.

**UI Design in Android**

* 1. **Android Layout Types**

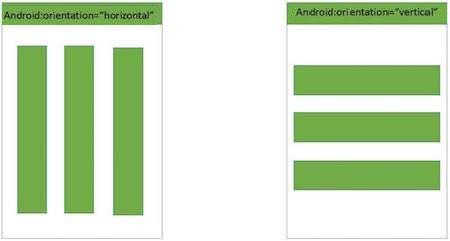
There are number of Layouts provided by Android which you will use in almost all the Android applications to provide different view, look and feel.

Some of them are:

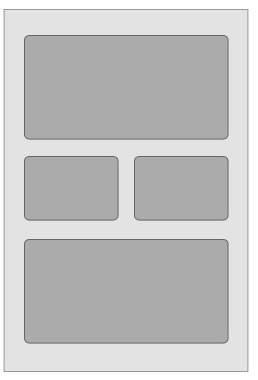
* + LinearLayout
  + Relative Layout
  + Table Layout
  + Absolute Layout
  + Frame Layout

***a.*** [Linear Layout](https://www.tutorialspoint.com/android/android_linear_layout.htm)

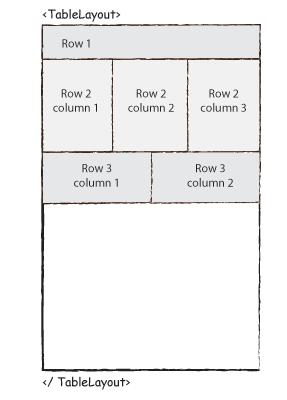
Android Linear Layout is a view group that aligns all children in a single direction, vertically or horizontally.



b. [Relative Layout](https://www.tutorialspoint.com/android/android_relative_layout.htm)

Android Relative Layout enables you to specify how child views are positioned relative to each other. The position of each view can be specified as relative to sibling elements or relative to the parent.

c. [Table Layout](https://www.tutorialspoint.com/android/android_table_layout.htm)



Android Table Layout going to be arranged groups of views into rows and columns. You will use the <Table Row> element to build a row in the table. Each row has zero or more cells;

each cell can hold one View object.

d. [Absolute Layout](https://www.tutorialspoint.com/android/android_table_layout.htm)

An Absolute Layout lets you specify exact locations

(x/y coordinates) of its children. Absolute layouts are

less flexible and harder to maintain than other types of layouts without absolute positioning.

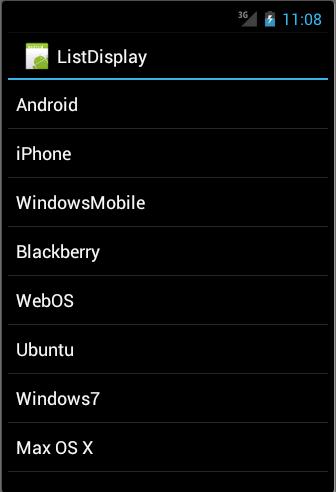
e. [Frame Layout](https://www.tutorialspoint.com/android/android_table_layout.htm)

Frame Layout is designed to block out an area on the screen to display a single item. Generally, FrameLayout should be used to hold a single child view, because it can be difficult to organize child views in a way that's scalable to different screen sizes without the children overlapping each other.

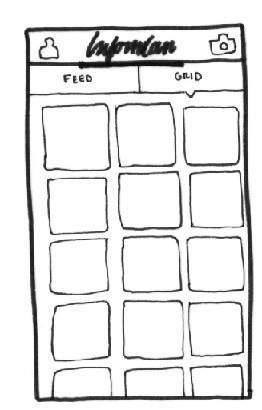
**2. Android UI Controls**

There are number of UI controls provided by Android that allow you to build the graphical user interface for your app.

|  |  |  |
| --- | --- | --- |
| [TextView](https://developer.android.com/studio/index.html) | EditText | AutoCompleteTextView |
| Button | ImageButton | CheckBox |
| [ToggleButton](http://coronalabs.com/) | [RadioButton](http://www.appcelerator.com/titanium/titanium-sdk/) | [ProgressBar](https://www.xamarin.com/) |
| Spinner | TimePicker | DatePicker |

[List View](https://www.tutorialspoint.com/android/android_list_view.htm)

Android List View is a view which groups several items and display them in vertical scrollable list. The list items are automatically inserted to the list using an Adapter that pulls content from a source such as an array or database.



[GridView](https://www.tutorialspoint.com/android/android_grid_view.htm)

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

**2. Units Of Measurements**

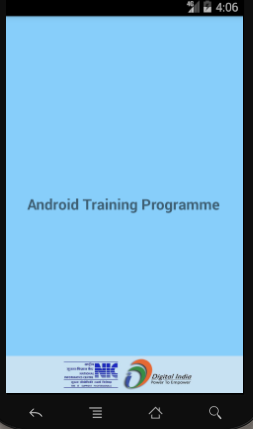
When you are specifying the size of an element on an Android UI, you should remember the following units of measurement.

|  |  |
| --- | --- |
| **Sr.No** | **Unit & description** |
| 1 | **Dp**  Density-independent pixel. 1 dp is equivalent to one pixel on a 160 dpi screen. |
| 2 | **sp**  Scale-independent pixel. This is similar to dp and is recommended for specifying font sizes |
| 3 | **pt**  Point. A point is defined to be 1/72 of an inch, based on the physical screen size. |
| 4 | **px**  Pixel. Corresponds to actual pixels on the screen |

**3. Screen Densities**

|  |  |
| --- | --- |
| **Sr.No** | **Density&Dpi** |
| **1** | **Low density (ldpi)**  120 dpi |
| **2** | **Medium density (mdpi)**  160 dpi |
| **3** | **High density (hdpi)**  240 dpi |
| **4** | **Extra High density (xhdpi)**  320dpi |

**Layout Design**

****

First create a layout as shown in above figure for splash screen

Create an Activity "Splash Activity".

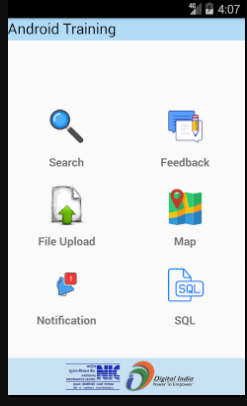
a. Add below code in "activity\_splash.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:orientation="vertical"android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:background="#87CEFA"**>  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_centerInParent="true"**  
**android:text="Android Training Programme"**  
**android:layout\_centerHorizontal="true"**  
**android:textSize="20dip"**  
**android:textStyle="bold"**  
/>  
  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
</**RelativeLayout**>

b. Add below code in "Splashactivity.java" file.

**package** cs.app.nic.com.seminarclass;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.os.Handler;  
**import** android.support.annotation.Nullable;  
**import** android.support.v7.app.AppCompatActivity;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/13/2017.*  
*\*/*  
  
**public class** Splashactivity **extends** AppCompatActivity{  
@Override  
**protected void** onCreate(@Nullable Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_splash***);  
**new** Handler().postDelayed(**new** Runnable() {  
  
@Override  
**public void** run() {  
Intent in = **new** Intent(Splashactivity.**this**, SlidingMenuActivity.**class**);  
startActivity(in);  
  
  
finish();  
}  
}, 2\*1000);  
  
}  
  
@Override  
**protected void** onDestroy() {  
**super**.onDestroy();  
}  
}

Create a Menu Page using gridview as shown in figure:



a. Add below code in "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="cs.app.nic.com.seminarclass.MainActivity"**>  
  
<**include**  
**android:id="@+id/ti"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_width="fill\_parent"**  
**layout="@layout/activity\_title"**  
  
/>  
<**LinearLayout**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="fill\_parent"**  
**android:layout\_below="@+id/ti"**>  
<**GridView**  
**android:id="@+id/grid"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_gravity="center"**  
**android:numColumns="2"**  
/>  
</**LinearLayout**>  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
</**RelativeLayout**>

b. Add below code in "MainActivity.java" file.

**package** cs.app.nic.com.seminarclass;  
  
  
**import** android.support.annotation.Nullable;  
**import** android.support.v4.app.Fragment;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.AdapterView;  
**import** android.widget.GridView;  
  
**public class** MainActivity **extends** Fragment {  
GridView **grid**;  
  
@Nullable  
@Override  
**public** View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {  
  
View view = inflater.inflate(R.layout.***activity\_main***, container, **false**);  
**grid**=(GridView)view.findViewById(R.id.***grid***);  
**grid**.setAdapter(**new** ImageAdapter(getActivity()));  
  
**return** view;  
  
}  
  
  
}

c. Add below code in "Image Adapter.java" file.

**package** cs.app.nic.com.seminarclass;  
**package** cs.app.nic.com.seminarclass;  
  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.BaseAdapter;  
**import** android.widget.ImageView;  
**import** android.widget.ListAdapter;  
**import** android.widget.TextView;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/12/2017.*  
*\*/*  
  
**class** ImageAdapter **extends** BaseAdapter {  
  
**private** Context **context**;  
**private int** [] **imageId**={R.drawable.***search***,R.drawable.***ffedback***,R.drawable.***fileupload***,R.drawable.***map***,R.drawable.***notification***,R.drawable.***sql***};  
**private** String[]**mobileValues**=**new** String[]{**"Search"**,**"Feedback"**,**"File Upload"**,**"Map"**,**"Notification"**,**"SQL"**};

*// Constructor*  
**public** ImageAdapter(Context c){  
**this**.**context** = c;  
  
}  
  
@Override  
**public int** getCount() {  
**return mobileValues**.**length**;  
}  
  
@Override  
**public** Object getItem(**int** position) {  
**return mobileValues**[position];  
}  
  
@Override  
**public long** getItemId(**int** position) {  
**return** 0;  
}  
  
@Override  
**public** View getView(**final int** position, View convertView, ViewGroup parent) {  
LayoutInflater inflater = (LayoutInflater) **context**  
.getSystemService(Context.***LAYOUT\_INFLATER\_SERVICE***);  
  
View gridView;  
  
**if** (convertView == **null**) {  
  
gridView = **new** View(**context**);  
  
*// get layout from mobile.xml*  
gridView = inflater.inflate(R.layout.***activity\_custom***, **null**);  
  
*// set value into textview*  
TextView textView = (TextView) gridView  
.findViewById(R.id.***grid\_item\_label***);  
textView.setText(**mobileValues**[position]);  
  
*// set image based on selected text*  
ImageView imageView = (ImageView) gridView  
.findViewById(R.id.***grid\_item\_image***);  
  
imageView.setImageResource(**imageId**[position]);  
gridView.setOnClickListener(**new** View.OnClickListener() {  
  
@Override  
**public void** onClick(View v) {  
*//* ***TODO Auto-generated method stub***  
*// Toast.makeText(context, "You Clicked "+result[position], Toast.LENGTH\_LONG).show()*  
**if**(position==0)  
{  
Intent in1 = **new** Intent(**context**, SearchActivity.**class**);  
**context**.startActivity(in1);  
  
}**else if**(position==1){  
Intent in2 = **new** Intent(**context**, Feedback.**class**);  
**context**.startActivity(in2);  
  
}  
**else if**(position==2){  
Intent in2 = **new** Intent(**context**, FileUpload.**class**);  
**context**.startActivity(in2);  
  
}**else if**(position==3){  
Intent in2 = **new** Intent(**context**, MapActivity.**class**);  
**context**.startActivity(in2);  
  
}  
**else if**(position==4){  
Intent in4 = **new** Intent(**context**, NotificationActivity.**class**);  
**context**.startActivity(in4);  
  
}  
**else**{  
Intent in4 = **new** Intent(**context**, SqlActivity.**class**);  
**context**.startActivity(in4);  
}  
}  
});  
  
} **else** {  
gridView = (View) convertView;  
}  
  
**return** gridView;  
}  
  
}

In the above examples we are placing a header in all pages.Let see the xml code for placing header and footer

d. Add below code in "activity\_title.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:orientation="vertical"android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:background="#B2DCF8"**>  
  
<**TextView**  
**android:id="@+id/txthead"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="30dp"**  
**android:text="Android Training"**  
**android:textSize="20dp"**  
**android:layout\_centerVertical="true"**  
**android:textColor="#000000"**  
**android:layout\_alignParentLeft="true"**  
/>  
<**TextView**  
**android:id="@+id/txttitile"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="20dp"**  
**android:layout\_centerVertical="true"**  
**android:textColor="#000000"**  
**android:layout\_alignParentRight="true"**/>  
  
</**RelativeLayout**>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Fragments**

A [Fragment](https://developer.android.com/reference/android/app/Fragment.html) represents a behavior or a portion of user interface in an [Activity](https://developer.android.com/reference/android/app/Activity.html). You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities. You can think of a fragment as a modular section of an activity, which has its own lifecycle, receives its own input events, and which you can add or remove while the activity is running (sort of like a "sub activity" that you can reuse in different activities).

a. Add below code in "MainActivity. java" file.

**package** cs.app.nic.com.volleyservice;  
  
**import** android.app.Activity;  
**import** android.os.Bundle;  
**import** android.support.annotation.Nullable;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/24/2017.*  
*\*/*  
  
**public class** MainActivity **extends** Activity {  
@Override  
**protected void** onCreate(@Nullable Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_main***);  
}  
}

b. Add below code in "activity\_main.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**LinearLayout**  
**xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:orientation="horizontal"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**>  
  
<**fragment**  
**android:id="@+id/frag1"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="fill\_parent"**  
**android:name="cs.app.nic.com.volleyservice.fragment1"**  
**android:layout\_weight="1"**></**fragment**>  
<**fragment**  
**android:id="@+id/frag2"**  
**android:name="cs.app.nic.com.volleyservice.fragment2"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="fill\_parent"**  
**android:layout\_weight="1"**></**fragment**>  
  
</**LinearLayout**>

c. Add below code in "fragment1.java" file.

**package** cs.app.nic.com.volleyservice;  
  
**import** android.app.Fragment;  
**import** android.app.FragmentManager;  
**import** android.app.FragmentTransaction;  
**import** android.os.Bundle;  
**import** android.support.annotation.Nullable;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.AdapterView;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.ListView;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/24/2017.*  
*\*/*  
  
**public class** fragment1 **extends** Fragment{  
@Nullable  
  
ListView **list**;  
String[] **val**={**"hai"**,**"helo"**,**"bye"**};  
**public** View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {  
  
View view=inflater.inflate(R.layout.***activity\_fragment1***,container, **false**);  
  
  
**list**=(ListView)view.findViewById(R.id.***l***);  
ArrayAdapter<String> adapter = **new** ArrayAdapter<String>(getActivity(),  
android.R.layout.***simple\_list\_item\_1***, **val**);  
**list**.setAdapter(adapter);  
**list**.setOnItemClickListener(**new** AdapterView.OnItemClickListener() {  
@Override  
**public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
  
fragment2 fragment= **new** fragment2();  
Bundle bundle = **new** Bundle();  
bundle.putString(**"val"**,**list**.getItemAtPosition(position).toString());  
fragment.setArguments(bundle);  
FragmentManager fragmentManager = getFragmentManager();  
  
FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();  
fragmentTransaction.replace(R.id.***frag2***,fragment);  
fragmentTransaction.commit();  
}  
  
  
});  
  
**return** view;  
}  
}

d. Add below code in "activity\_fragment1.java" file.

*<?***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="#DB7093"**>  
<**ListView**  
**android:id="@+id/l"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**>  
  
</**ListView**>  
</**LinearLayout**>

e. Add below code in "fragment2.java" file.

**package** cs.app.nic.com.volleyservice;  
  
**import** android.app.Fragment;  
**import** android.graphics.Color;  
**import** android.os.Bundle;  
**import** android.support.annotation.ColorInt;  
**import** android.support.annotation.Nullable;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.TextView;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/24/2017.*  
*\*/*  
  
**public class** fragment2 **extends** Fragment{  
@Nullable  
TextView **txt**;  
  
@Override  
**public** View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {  
  
View view=inflater.inflate(R.layout.***activity\_fragment2***,container,**false**);  
**txt**=(TextView)view.findViewById(R.id.***txt***);  
  
Bundle args = getArguments();  
**if** (args != **null**){  
**txt**.setText(args.getString(**"val"**));  
**txt**.setTextColor(Color.*parseColor*(**"#00FF00"**));  
}  
**return** view;  
}  
}

f. Add below code in "activity\_fragment2.java" file.

*<?***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="#E9967A"**>  
  
<**TextView**  
**android:id="@+id/txt"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_gravity="center"**/>  
  
</**LinearLayout**>

**Permissions**

Beginning in Android 6.0 (API level 23), users grant permissions to apps while the app is running, not when they install the app. This approach streamlines the app install process, since the user does not need to grant permissions when they install or update the app. It also gives the user more control over the app's functionality; for example, a user could choose to give a camera app access to the camera but not to the device location. The user can revoke the permissions at any time, by going to the app's Settings screen.

System permissions are divided into two categories, *normal* and *dangerous:*

* Normal permissions do not directly risk the user's privacy. If your app lists a normal permission in its manifest, the system grants the permission automatically.
* Dangerous permissions can give the app access to the user's confidential data. If your app lists a normal permission in its manifest, the system grants the permission automatically. If you list a dangerous permission, the user has to explicitly give approval to your app.

**Note:** Beginning with Android 6.0 (API level 23), users can revoke permissions from any app at any time, even if the app targets a lower API level. You should test your app to verify that it behaves properly when it's missing a needed permission, regardless of what API level your app targets.

**Check For Permissions**

If your app needs a dangerous permission, you must check whether you have that permission every time you perform an operation that requires that permission. The user is always free to revoke the permission, so even if the app used the camera yesterday, it can't assume it still has that permission today.

To check if you have a permission, call the

[ContextCompat.checkSelfPermission()](https://developer.android.com/reference/android/support/v4/content/ContextCompat.html#checkSelfPermission%2528android.content.Context,%2520java.lang.String%2529) method. For example, this snippet shows how to check if the activity has permission to write to the calendar:

// Assume thisActivity is the current activity

int permissionCheck = ContextCompat.checkSelfPermission(thisActivity,  
Manifest.permission.WRITE\_CALENDAR);

If the app has the permission, the method returns

[PackageManager.PERMISSION\_GRANTED](https://developer.android.com/reference/android/content/pm/PackageManager.html#PERMISSION_GRANTED),

and the app can proceed with the operation. If the app does not have the permission, the method returns [PERMISSION\_DENIED](https://developer.android.com/reference/android/content/pm/PackageManager.html#PERMISSION_DENIED), and the app has to explicitly ask the user for permission.

**Request Permissions**

If your app needs a dangerous permission that was listed in the app manifest, it must ask the user to grant the permission. Android provides several methods you can use to request permission. Calling these methods brings up a standard Android dialog, which you cannot customize.

**Shared Preferences**

**SharedPreferences** is an API from **Android** SDK to store and retrieve application preferences. **SharedPreferences** are simply sets of data values that stored persistently.

How to store value:

SharedPreferences pref = getApplicationContext().getSharedPreferences("ids", 0);

SharedPreferences.Editor editor = pref.edit();

editor.putString("regId", gettoken);//get token contains value

editor.commit();

How to getvalue:

SharedPreferences pref = getApplicationContext().getSharedPreferences("ids", 0);

String regId = pref.getString("regId", null);

Log.e(TAG, "Firebase reg id: " + regId);

**Advanced UIDesign**

**Material Design**

For implementing material design we have add this library to app gradle:

compile **'com.android.support:design:25.3.1'**

Lets see an example:

activity\_search.xml

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:orientation="vertical"**>

<**include**  
**android:id="@+id/head"**  
**layout="@layout/activity\_title"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
  
/>  
  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_app"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/head"**  
**android:layout\_marginTop="10dp"**>  
  
<**EditText**  
**android:id="@+id/input\_app"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Application Number\*"**/>  
  
</**android.support.design.widget.TextInputLayout**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_mob"  
android:layout\_width="match\_parent"  
android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/input\_layout\_app"**  
**android:layout\_marginTop="10dp"**>  
  
<**EditText**  
**android:id="@+id/input\_mob"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Mobile Number Given In The Application\*"**  
**android:inputType="number"**/>  
  
</**android.support.design.widget.TextInputLayout**>  
  
<**Button**  
**android:id="@+id/btnsub"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/input\_layout\_mob"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_marginTop="10dp"**  
**android:background="@drawable/buttoncustom"**  
**android:onClick="onSearchClick"**  
**android:text="Submit"**  
**android:textColor="#000000"**/>  
  
<**LinearLayout**  
**android:id="@+id/details"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/btnsub"**  
**android:orientation="vertical"**  
**android:visibility="invisible"**>  
  
<**TextView**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="10dp"**  
**android:background="#B2DCF8"**  
**android:gravity="center"**  
**android:text="Grievance Details"**  
**android:textColor="#000000"**  
**android:textSize="18dp"**/>  
  
<**TableLayout**  
**android:id="@+id/table"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="20dp"**>  
  
<**TableRow**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:gravity="center|center\_horizontal"**>  
  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Application Number"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
  
<**TextView**  
**android:id="@+id/appnum"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
</**TableRow**>  
  
<**TableRow**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:gravity="center|center\_horizontal"**>  
  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Applicant Name"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
  
<**TextView**  
**android:id="@+id/appname"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
</**TableRow**>  
  
<**TableRow**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:gravity="center|center\_horizontal"**>  
  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
  
**android:layout\_height="wrap\_content"**  
**android:text="Status"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
  
<**TextView**  
**android:id="@+id/appsta"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
</**TableRow**>  
</**TableLayout**>  
  
<**TableLayout**  
**android:id="@+id/tablelayout\_doc\_list"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:background="#ffffff"**  
  
**android:orientation="vertical"**  
**android:shrinkColumns="\*"**  
**android:stretchColumns="\*"**>  
  
  
</**TableLayout**>  
</**LinearLayout**>  
  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
</**RelativeLayout**>

Edittext with floating labels

Android provides two new widgets for displaying cards and lists with material design.

For Recycler view we have to add :

compile **'com.android.support:recyclerview-v7:27.1.1'**

For Card view we have to add :

compile **'com.android.support:cardview-v7:27.1.1'**

Lets see an example of Recycler view :

a. Add below code in "activity\_recycle.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout**  
**xmlns:android="http://schemas.android.com/apk/res/android"**  
**xmlns:tools="http://schemas.android.com/tools"**  
**xmlns:app="http://schemas.android.com/apk/res-auto"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"**  
**tools:showIn="@layout/activity\_main"**  
**tools:context=".MainActivity"**>  
  
<**android.support.v7.widget.RecyclerView**  
**android:id="@+id/recycler\_view"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:scrollbars="vertical"**/>  
  
</**RelativeLayout**>

b. Add below code in "RecyclerviewExample.java" file.

**package** cs.app.nic.com.seminarclass;  
  
*/\*\**  
*\* Created by asg4.dev13 on 8/14/2017.*  
*\*/*  
  
  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.support.v7.widget.DefaultItemAnimator;  
**import** android.support.v7.widget.LinearLayoutManager;  
**import** android.support.v7.widget.RecyclerView;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
 **public class** RecyclerviewExample **extends** AppCompatActivity {  
**private** List<listvalues>**movieList** = **new** ArrayList<>();  
**private** RecyclerView **recyclerView**;  
**private** listadapter **mAdapter**;  
  
@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_recycle***);  
**recyclerView** = (RecyclerView) findViewById(R.id.***recycler\_view***);  
  
**mAdapter** = **new** listadapter(**movieList**);  
RecyclerView.LayoutManager mLayoutManager = **new** LinearLayoutManager(getApplicationContext());  
**recyclerView**.setLayoutManager(mLayoutManager);  
**recyclerView**.setItemAnimator(**new** DefaultItemAnimator());  
**recyclerView**.setAdapter(**mAdapter**);  
  
prepareMovieData();  
}  
  
**private void** prepareMovieData() {  
listvalues movie = **new** listvalues(**"Mad Max: Fury Road"**, **"Action & Adventure"**, **"2015"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Inside Out"**, **"Animation, Kids & Family"**, **"2015"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Star Wars: Episode VII - The Force Awakens"**, **"Action"**, **"2015"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Shaun the Sheep"**, **"Animation"**, **"2015"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"The Martian"**, **"Science Fiction & Fantasy"**, **"2015"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Mission: Impossible Rogue Nation"**, **"Action"**, **"2015"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Up"**, **"Animation"**, **"2009"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Star Trek"**, **"Science Fiction"**, **"2009"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"The LEGO Movie"**, **"Animation"**, **"2014"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Iron Man"**, **"Action & Adventure"**, **"2008"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Aliens"**, **"Science Fiction"**, **"1986"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Chicken Run"**, **"Animation"**, **"2000"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Back to the Future"**, **"Science Fiction"**, **"1985"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Raiders of the Lost Ark"**, **"Action & Adventure"**, **"1981"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Goldfinger"**, **"Action & Adventure"**, **"1965"**);  
**movieList**.add(movie);  
  
movie = **new** listvalues(**"Guardians of the Galaxy"**, **"Science Fiction & Fantasy"**, **"2014"**);  
**movieList**.add(movie);  
  
**mAdapter**.notifyDataSetChanged();  
}  
}

c. Add below code in "listvalues.java" file.

**package** cs.app.nic.com.seminarclass;  
  
*/\*\**  
*\* Created by asg4.dev13 on 7/6/2017.*  
*\*/*  
  
**public class** listvalues {  
**private** String **title**, **genre**, **year**;  
  
**public** listvalues() {  
}  
  
**public** listvalues(String title, String genre, String year) {  
**this**.**title** = title;  
**this**.**genre** = genre;  
**this**.**year** = year;  
}  
  
**public** String getTitle() {  
**return title**;  
}  
  
**public void** setTitle(String title) {  
**this**.**title** = title;  
}  
  
**public** String getGenre() {  
**return genre**;  
}  
  
**public void** setGenre(String genre) {  
**this**.**genre** = genre;  
}  
  
**public** String getYear() {  
**return year**;  
}  
  
**public void** setYear(String year) {  
**this**.**year** = year;  
}  
}

d. Add below code in "listadapter.java" file.

**package** cs.app.nic.com.seminarclass;  
  
**import** android.support.v7.widget.RecyclerView;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.TextView;  
  
**import** java.util.List;  
  
*/\*\**  
*\* Created by asg4.dev13 on 7/6/2017.*  
*\*/*  
  
**public class** listadapter **extends** RecyclerView.Adapter<listadapter.MyViewHolder>{  
  
  
**private** List<listvalues>**moviesList**;  
  
**public class** MyViewHolder **extends** RecyclerView.ViewHolder {  
**public** TextView **title**, **year**, **genre**;  
  
**public** MyViewHolder(View itemView) {  
**super**(itemView);  
**title** = (TextView) itemView.findViewById(R.id.***title***);  
**genre** = (TextView) itemView.findViewById(R.id.***genre***);  
**year** = (TextView) itemView.findViewById(R.id.***year***);  
}  
}  
**public** listadapter(List<listvalues> moviesList) {  
**this**.**moviesList** = moviesList;  
}  
  
@Override  
**public** MyViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
View itemView = LayoutInflater.*from*(parent.getContext())  
.inflate(R.layout.***listrow***, parent, **false**);  
  
**return new** MyViewHolder(itemView);  
}  
  
@Override  
**public void** onBindViewHolder(MyViewHolder holder, **int** position) {  
listvalues movie = **moviesList**.get(position);  
holder.**title**.setText(movie.getTitle());  
holder.**genre**.setText(movie.getGenre());  
holder.**year**.setText(movie.getYear());  
}  
  
@Override  
**public int** getItemCount() {  
**return moviesList**.size();  
}  
}

**Android Sliding Menu Using Navigation Drawer**

You might have noticed that lot of android applications introduced a sliding panel menu to navigate between major modules of the application. Previously this kind of UI was done using some third party libraries where a list view and some swiping gestures used to achieve this. But now android itself officially introduced sliding panel menu by introducing a newer concept called Navigation Drawer in which we combine [DrawerLayout](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.html) and [NavigationView](https://developer.android.com/reference/android/support/design/widget/NavigationView.html) to achieve the desired output.

a. Add below code in "SlidingMenuActivity" file.

**package** cs.app.nic.com.seminarclass;  
  
**import** android.app.ActionBar;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.support.design.widget.NavigationView;  
**import** android.support.v4.app.FragmentTransaction;  
**import** android.support.v4.view.GravityCompat;  
**import** android.support.v4.widget.DrawerLayout;  
**import** android.support.v7.app.ActionBarDrawerToggle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.support.v7.widget.Toolbar;  
**import** android.view.Gravity;  
**import** android.view.MenuItem;  
**import** android.widget.ImageView;  
  
*/\*\**  
*\* Created by asg4.dev13 on 5/25/2017.*  
*\*/*  
  
**public class** SlidingMenuActivity **extends** AppCompatActivity **implements** NavigationView.OnNavigationItemSelectedListener {  
DrawerLayout **drawer**;  
  
  
NavigationView **navigationView**;  
**private static final** String ***TAG\_Menu*** = **"menu"**;  
**private static final** String ***TAG\_Search*** = **"search"**;  
**private static final** String ***TAG\_Recycle*** = **"inddai"**;  
**private static final** String ***TAG\_map***= **"indwee"**;  
**private static final** String ***TAG\_notification*** = **"indmon"**;  
**private static final** String ***TAG\_sql*** = **"indyea"**;  
**private static final** String ***TAG\_feed*** = **"feed"**;  
**private static final** String ***TAG\_upload***= **"upload"**;  
  
  
**int navItemIndex** = 0;  
  
**private static** String *CURRENT\_TAG* = ***TAG\_Menu***;  
  
**private** String[] **activityTitles**;  
  
  
@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_sidemenu***);  
Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
setSupportActionBar(toolbar);  
**drawer** = (DrawerLayout) findViewById(R.id.***drawer\_layout***);  
  
  
  
**activityTitles** = getResources().getStringArray(R.array.***nav\_item\_activity\_titles***);  
ActionBarDrawerToggle toggle = **new** ActionBarDrawerToggle(  
**this**, **drawer**, toolbar, R.string.***navigation\_drawer\_open***, R.string.***navigation\_drawer\_close***);  
**drawer**.setDrawerListener(toggle);  
toggle.syncState();  
  
**navigationView** = (NavigationView) findViewById(R.id.***nav\_view***);  
**navigationView**.setNavigationItemSelectedListener(**this**);  
  
  
**if** (savedInstanceState == **null**) {  
**navItemIndex** = 0;  
loadHomeFragment();  
}  
}  
  
**private void** loadHomeFragment() {  
  
  
selectNavMenu();  
  
*// set toolbar title*  
*// setToolbarTitle();*  
  
*// if user select the current navigation menu again, don't do anything*  
*// just close the navigation drawer*  
**if** (getSupportFragmentManager().findFragmentByTag(*CURRENT\_TAG*) != **null**) {  
**drawer**.closeDrawers();  
  
*// show or hide the fab button*  
*// toggleFab();*  
**return**;  
}  
MainActivity fragment = **new** MainActivity();  
FragmentTransaction fragmentTransaction = getSupportFragmentManager().beginTransaction();  
fragmentTransaction.setCustomAnimations(android.R.anim.***fade\_in***,  
android.R.anim.***fade\_out***);  
fragmentTransaction.replace(R.id.***frame***, fragment);  
fragmentTransaction.commitAllowingStateLoss();  
  
*//invalidateOptionsMenu();*  
}  
  
  
**private void** selectNavMenu() {  
**navigationView**.getMenu().getItem(**navItemIndex**).setChecked(**true**);  
}  
  
@Override  
**public void** onBackPressed() {  
  
**if** (**drawer**.isDrawerOpen(GravityCompat.***START***)) {  
**drawer**.closeDrawer(GravityCompat.***START***);  
} **else** {  
  
  
**super**.onBackPressed();  
}  
}  
  
  
@Override  
**public boolean** onOptionsItemSelected(MenuItem item) {  
*// Handle action bar item clicks here. The action bar will*  
*// automatically handle clicks on the Home/Up button, so long*  
*// as you specify a parent activity in AndroidManifest.xml.*  
**int** id = item.getItemId();  
  
**return super**.onOptionsItemSelected(item);  
}  
  
@SuppressWarnings(**"StatementWithEmptyBody"**)  
  
**public boolean** onNavigationItemSelected(MenuItem item) {  
*// Handle navigation view item clicks here.*  
**int** id = item.getItemId();  
  
**if** (id == R.id.***search***) {  
**navItemIndex** = 0;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,SearchActivity.**class**);  
startActivity(i);  
  
*CURRENT\_TAG* = ***TAG\_Search***;  
  
  
}  
**else if** (id == R.id.***Recyclerview***) {  
**navItemIndex** = 1;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,RecyclerviewExample.**class**);  
startActivity(i);  
*CURRENT\_TAG* = ***TAG\_Recycle***;  
  
  
}  
**else if** (id == R.id.***feed***) {  
**navItemIndex** = 1;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,Feedback.**class**);  
startActivity(i);  
*CURRENT\_TAG* = ***TAG\_feed***;  
  
  
}  
**else if** (id == R.id.***fileupload***) {  
**navItemIndex** = 1;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,FileUpload.**class**);  
startActivity(i);  
*CURRENT\_TAG* = ***TAG\_upload***;  
  
  
}  
**else if** (id == R.id.***Map***) {  
**navItemIndex** = 1;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,MapActivity.**class**);  
startActivity(i);  
*CURRENT\_TAG* = ***TAG\_map***;  
  
  
}  
**else if** (id == R.id.***sql***) {  
**navItemIndex** = 1;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,SqlActivity.**class**);  
startActivity(i);  
*CURRENT\_TAG* = ***TAG\_sql***;  
  
  
}  
**else if** (id == R.id.***noti***) {  
**navItemIndex** = 1;  
Intent i=**new** Intent(SlidingMenuActivity.**this**,NotificationActivity.**class**);  
startActivity(i);  
*CURRENT\_TAG* = ***TAG\_notification***;  
  
  
}  
  
  
  
  
**drawer**.closeDrawer(GravityCompat.***START***);  
loadHomeFragment();  
**return true**;  
}  
  
  
}

b. Add below code in "activity\_sidemenu.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**android.support.v4.widget.DrawerLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**xmlns:app="http://schemas.android.com/apk/res-auto"**  
**xmlns:tools="http://schemas.android.com/tools"**  
**android:id="@+id/drawer\_layout"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:fitsSystemWindows="true"**  
**tools:openDrawer="start"**>  
  
<**include**  
**layout="@layout/app\_bar\_main"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**/>  
  
<**android.support.design.widget.NavigationView**  
**android:id="@+id/nav\_view"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="match\_parent"**  
**android:layout\_gravity="start"**  
**android:fitsSystemWindows="true"**

**app:headerLayout="@layout/activity\_navheadermain"**

**app:menu="@menu/activity\_main\_drawer"**/>  
  
</**android.support.v4.widget.DrawerLayout**>

c. Add below code in "app\_bar\_main.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**android.support.design.widget.CoordinatorLayout 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=".SlidingMenuActivity"**>  
  
<**android.support.design.widget.AppBarLayout**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:theme="@style/AppTheme.AppBarOverlay"**  
>  
  
<**android.support.v7.widget.Toolbar**  
**android:id="@+id/toolbar"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="?attr/actionBarSize"**  
**android:background="#4E8EE1"**  
**app:contentInsetStartWithNavigation="0dp"**  
/>  
*<!--app:theme="@style/ToolbarColoredBackArrow"-->*  
  
  
</**android.support.design.widget.AppBarLayout**>  
  
<**FrameLayout**  
**android:id="@+id/frame"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"**  
>  
  
</**FrameLayout**>  
  
  
</**android.support.design.widget.CoordinatorLayout**>

d. Add below code in "activity\_navheadermain.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"**  
**android:id="@+id/view\_container"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="@dimen/nav\_header\_height"**  
**android:gravity="bottom"**  
**android:orientation="vertical"**  
**android:background="#4E8EE1"**  
**android:theme="@style/ThemeOverlay.AppCompat.Dark"**>  
  
  
  
<**LinearLayout**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_centerVertical="true"**  
**android:orientation="vertical"**  
**android:padding="@dimen/activity\_horizontal\_margin"**>  
  
<**ImageView**  
**android:id="@+id/img\_profile"**  
**android:layout\_width="@dimen/profile\_width"**  
**android:layout\_height="@dimen/profile\_height"**  
**android:paddingTop="@dimen/nav\_header\_vertical\_spacing"**  
**app:srcCompat="@mipmap/ic\_launcher"**/>  
  
<**TextView**  
**android:id="@+id/name"**  
**android:text="Android Training"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:paddingTop="@dimen/nav\_header\_vertical\_spacing"**  
**android:textAppearance="@style/TextAppearance.AppCompat.Body1"**/>  
  
  
  
</**LinearLayout**>  
</**RelativeLayout**>

Create a menu folder under res and name as:activity\_main\_drawer.xml

e. Add below code in "activity\_main\_drawer.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**menu xmlns:android="http://schemas.android.com/apk/res/android"**>  
  
<**group android:checkableBehavior="single"**>  
  
<**item**  
**android:id="@+id/search"**  
  
  
**android:title="Search"**/>  
  
  
<**item**  
**android:id="@+id/Recyclerview"**  
  
  
**android:title="Recyclerview"**/>  
<**item**  
**android:id="@+id/fileupload"**  
  
  
**android:title="File Upload"**/>  
<**item**  
**android:id="@+id/feed"**  
  
  
**android:title="Enter Feedback"**/>  
</**group**>  
<**item android:title=""**>  
<**menu**>  
<**item**  
**android:id="@+id/Map"**  
  
  
**android:title="Load Map"**/>  
  
  
<**item**  
**android:id="@+id/noti"**  
  
  
**android:title="Notification"**/>  
<**item**  
**android:id="@+id/sql"**  
  
  
**android:title="SQL"**/>  
  
</**menu**>  
</**item**>  
</**menu**>

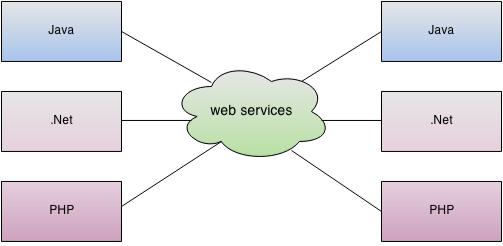
**Introduction to webservice**

**What is Web Service?**

A Web Service can be defined in following ways:

* A client server application or application component for communication.
* A method of communication between two devices over network.
* A software system for interoperable machine to machine communication.
* A collection of standards or protocols for exchanging information between two devices or application.

Let's understand it by the figure given below:



As you can see in the figure, java, .net or PHP applications can communicate with other applications through web service over the network. For example, java application can interact with Java, .Net

and PHP applications. So web service is a language independent way of communication.

**Types of Web Services**

There are mainly two types of web services.

* + SOAP web services.
  + RESTful web services.

**SOAP Web Services**

SOAP stands for Simple Object Access Protocol. It is a XML-based protocol for accessing web services.SOAP is a W3C recommendation for communication between two applications.SOAP is XML based protocol. It is platform independent and language independent. By using SOAP, you will be able to interact with other programming language applications.

a. Advantages of Soap Web Services

**WS Security**: SOAP defines its own security known as WS Security.

**Language and Platform independent**: SOAP web services can bewritten in any programming language and executed in any platform.

b. Disadvantages of Soap Web Services

Slow**: SOAP uses XML format that must be parsed to be read. Itdefines many standards that must be followed while developing the SOAP applications. So it is slow and consumes more bandwidth and resource.**

WSDL dependent**:** **SOAP** **uses** **WSDL(Web** **Services** **Description**

**Language) and doesn't have any other mechanism to discover the service.**

**RESTful Web Services**

REST stands for Representational State Transfer.

REST is an architectural style not a protocol.

a. Advantages of RESTful Web Services

**Fast**: RESTful Web Services are fast because there is no strictspecification like SOAP. It consumes less bandwidth and resource.

**Language and Platform independent**: RESTful web services can bewritten in any programming language and executed in any platform.

**I can use SOAP**: RESTful web services can use SOAP web services asthe implementation.

**Permits different data format**: RESTful web service permits differentdata format such as Plain Text, HTML, XML and JSO

**Volley library** can be used for calling webservice within an app. If we are using volley library mention the following line in build.gradle

compile com.android.volley:volley:1.0.0

Advantages of using volley library:

1. Volley automatically schedule all network requests. It means that Volley will be taking care of all the network requests your app executes for fetching response or image from web.
2. Volley provides transparent disk and memory caching.
3. Volley provides powerful cancellation request API. It means that you can cancel a single request or you can set blocks or scopes of requests to cancel.
4. Volley provides powerful customization abilities.
5. Volley provides Debugging and tracing tools

While calling webservice we have to give certain permissions in Android Manifest.xml

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**manifest xmlns:android="http://schemas.android.com/apk/res/android"**  
**package="cs.app.nic.com.seminarclass"**>  
<**uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"**/>  
<**uses-permission android:name="android.permission.INTERNET"**/>  
<**uses-permission android:name="com.google.android.providers.gsf.permission.READ\_GSERVICES"**/>  
<**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=".Splashactivity"**  
**android:theme="@style/AppTheme.NoActionBar"**>  
<**intent-filter**>  
<**action android:name="android.intent.action.MAIN"**/>  
  
<**category android:name="android.intent.category.LAUNCHER"**  
/>  
</**intent-filter**>  
</**activity**>  
<**activity android:name=".MainActivity"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".MapActivity"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".SqlActivity"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".RegisterActivity"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".NotificationActivity"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".SearchActivity"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".Feedback"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**activity android:name=".FileUpload"**  
**android:theme="@style/AppTheme.NoActionBar"**/>  
<**meta-data**  
**android:name="com.google.android.maps.v2.API\_KEY"**  
**android:value="AIzaSyCvm8FJDa9ZTMIRmdi1ASmiLC7Wpb-cbjY"**/>  
  
</**application**>  
  
</**manifest**

Create four common classes used to dvelop the app:

*1.Base64Utils*

*It is used for encoding purpouses*

**package** cs.app.nic.com.seminarclass;  
  
**public class** Base64Utils {  
  
**private static byte**[] *mBase64EncMap*, *mBase64DecMap*;  
  
*/\*\**  
*\* Class initializer. Initializes the Base64 alphabet (specified in*  
*\* RFC-2045).*  
*\*/*  
  
**static** {  
**byte**[] base64Map = { (**byte**) **'A'**, (**byte**) **'B'**, (**byte**) **'C'**, (**byte**) **'D'**,  
(**byte**) **'E'**, (**byte**) **'F'**, (**byte**) **'G'**, (**byte**) **'H'**, (**byte**) **'I'**,  
(**byte**) **'J'**, (**byte**) **'K'**, (**byte**) **'L'**, (**byte**) **'M'**, (**byte**) **'N'**,  
(**byte**) **'O'**, (**byte**) **'P'**, (**byte**) **'Q'**, (**byte**) **'R'**, (**byte**) **'S'**,  
(**byte**) **'T'**, (**byte**) **'U'**, (**byte**) **'V'**, (**byte**) **'W'**, (**byte**) **'X'**,  
(**byte**) **'Y'**, (**byte**) **'Z'**, (**byte**) **'a'**, (**byte**) **'b'**, (**byte**) **'c'**,  
(**byte**) **'d'**, (**byte**) **'e'**, (**byte**) **'f'**, (**byte**) **'g'**, (**byte**) **'h'**,  
(**byte**) **'i'**, (**byte**) **'j'**, (**byte**) **'k'**, (**byte**) **'l'**, (**byte**) **'m'**,  
(**byte**) **'n'**, (**byte**) **'o'**, (**byte**) **'p'**, (**byte**) **'q'**, (**byte**) **'r'**,  
(**byte**) **'s'**, (**byte**) **'t'**, (**byte**) **'u'**, (**byte**) **'v'**, (**byte**) **'w'**,  
(**byte**) **'x'**, (**byte**) **'y'**, (**byte**) **'z'**, (**byte**) **'0'**, (**byte**) **'1'**,  
(**byte**) **'2'**, (**byte**) **'3'**, (**byte**) **'4'**, (**byte**) **'5'**, (**byte**) **'6'**,  
(**byte**) **'7'**, (**byte**) **'8'**, (**byte**) **'9'**, (**byte**) **'+'**, (**byte**) **'/'** };  
*mBase64EncMap* = base64Map;  
*mBase64DecMap* = **new byte**[128];  
**for** (**int** i = 0; i <*mBase64EncMap*.**length**; i++)  
*mBase64DecMap*[*mBase64EncMap*[i]] = (**byte**) i;  
}  
  
*/\*\**  
*\* This class isn't meant to be instantiated.*  
*\*/*  
  
**private** Base64Utils() {  
}  
  
*/\*\**  
*\* Encodes the given byte[] using the Base64-encoding, as specified in*  
*\* RFC-2045 (Section 6.8).*  
*\**   
*\** ***@param aData***  
*\* the data to be encoded*  
*\** ***@return*** *the Base64-encoded <var>aData</var>*  
*\** ***@exception*** *IllegalArgumentException*  
*\* if NULL or empty array is passed*  
*\*/*  
  
**public static** String base64Encode(**byte**[] aData) {  
**if** ((aData == **null**) || (aData.**length** == 0))  
**throw new** IllegalArgumentException(  
  
**"Cannot encode NULL or empty byte array."**);  
  
**byte** encodedBuf[] = **new byte**[((aData.**length** + 2) / 3) \* 4];  
  
*// 3-byte to 4-byte conversion*  
**int** srcIndex, destIndex;  
**for** (srcIndex = 0, destIndex = 0; srcIndex < aData.**length** - 2; srcIndex += 3) {  
encodedBuf[destIndex++] =  
  
*mBase64EncMap*[(aData[srcIndex] >>>2) &077];  
encodedBuf[destIndex++] = *mBase64EncMap*[  
  
(aData[srcIndex + 1] >>>4) &017 | (aData[srcIndex] <<4) &077];  
encodedBuf[destIndex++] = *mBase64EncMap*[  
  
(aData[srcIndex + 2] >>>6) &003 | (aData[srcIndex + 1] <<2)  
&077];  
encodedBuf[destIndex++] = *mBase64EncMap*[  
  
aData[srcIndex + 2] &077];  
}  
  
*// Convert the last 1 or 2 bytes*  
**if** (srcIndex < aData.**length**) {  
encodedBuf[destIndex++] =  
  
*mBase64EncMap*[(aData[srcIndex] >>>2) &077];  
**if** (srcIndex < aData.**length** - 1) {  
encodedBuf[destIndex++] = *mBase64EncMap*[(aData[srcIndex + 1] >>>4)  
&017 | (aData[srcIndex] <<4) &077];  
encodedBuf[destIndex++] =  
  
*mBase64EncMap*[(aData[srcIndex + 1] <<2) &077];  
} **else** {  
encodedBuf[destIndex++] =  
  
*mBase64EncMap*[(aData[srcIndex] <<4) &077];  
}  
}  
  
*// Add padding to the end of encoded data*  
**while** (destIndex < encodedBuf.**length**) {  
encodedBuf[destIndex] = (**byte**) **'='**;  
destIndex++;  
}  
  
String result = **new** String(encodedBuf);  
**return** result;  
}  
  
*/\*\**  
*\* Decodes the given Base64-encoded data, as specified in RFC-2045 (Section*  
*\* 6.8).*  
*\**   
*\** ***@param aData***  
*\* the Base64-encoded aData.*  
*\** ***@return*** *the decoded <var>aData</var>.*  
*\** ***@exception*** *IllegalArgumentException*  
*\* if NULL or empty data is passed*  
*\*/*  
  
**public static byte**[] base64Decode(String aData) {  
**if** ((aData == **null**) || (aData.length() == 0))  
**throw new** IllegalArgumentException(  
  
**"Cannot decode NULL or empty string."**);  
  
**byte**[] data = aData.getBytes();  
  
*// Skip padding from the end of encoded data*  
**int** tail = data.**length**;  
**while** (data[tail - 1] == **'='**)  
tail--;  
  
**byte** decodedBuf[] = **new byte**[tail - data.**length** / 4];  
  
*// ASCII-printable to 0-63 conversion*  
**for** (**int** i = 0; i < data.**length**; i++)  
data[i] = *mBase64DecMap*[data[i]];  
  
*// 4-byte to 3-byte conversion*  
**int** srcIndex, destIndex;  
**for** (srcIndex = 0, destIndex = 0; destIndex < decodedBuf.**length** - 2; srcIndex += 4, destIndex += 3) {  
decodedBuf[destIndex] = (**byte**)  
  
(((data[srcIndex] <<2) &255) | ((data[srcIndex + 1] >>>4) &003));  
decodedBuf[destIndex + 1] = (**byte**)  
  
(((data[srcIndex + 1] <<4) &255) | ((data[srcIndex + 2] >>>2) &017));  
decodedBuf[destIndex + 2] = (**byte**)  
  
(((data[srcIndex + 2] <<6) &255) | (data[srcIndex + 3] &077));  
}  
  
*// Handle last 1 or 2 bytes*  
**if** (destIndex < decodedBuf.**length**)  
decodedBuf[destIndex] = (**byte**) (  
  
((data[srcIndex] <<2) &255) | ((data[srcIndex + 1] >>>4) &003));  
**if** (++destIndex < decodedBuf.**length**)  
decodedBuf[destIndex] = (**byte**)  
  
(((data[srcIndex + 1] <<4) &255) | ((data[srcIndex + 2] >>>2) &017));  
  
**return** decodedBuf;  
}  
  
}

*2.Utils*

*In this class we have written some common function we are using repeatedly*

**package** cs.app.nic.com.seminarclass;  
  
**import** android.app.Activity;  
**import** android.app.AlertDialog;  
**import** android.app.Dialog;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.graphics.Color;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.view.View;  
**import** android.view.Window;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
  
  
*/\*\**  
*\* Created by asg4.dev13 on 4/17/2017.*  
*\*/*  
  
**public class** Utils {  
**static** Context *context*;  
**private static** Activity *activity*;  
  
**private static** AlertDialog *dialog*;  
**public** Utils(Context con) {  
*context* = con;  
*activity* = (Activity) con;  
  
}  
  
**public void** alertButton(**final** String data, **final boolean** closeactivity) {  
*activity*.runOnUiThread(**new** Runnable() {  
**public void** run() {  
  
**final** Dialog dialog = **new** Dialog(*activity*);  
  
*//final CustomDialog dialog = new CustomDialog(activity, R.style.myCoolDialog);*  
dialog.requestWindowFeature(Window.***FEATURE\_NO\_TITLE***);  
dialog.setContentView(R.layout.***activity\_alertcust***);  
*// dialog.setd*  
*// dialog.getWindow().setTitleColor(Color.parseColor("#8B4513"));*  
  
*// dialog.setTitle("Message....");*  
*//*  
  
*// set the custom dialog components - text, image and button*  
TextView text = (TextView) dialog.findViewById(R.id.***txt\_data***);  
text.setText(data);  
  
  
Button dialogButton = (Button) dialog  
.findViewById(R.id.***btnok***);  
*// if button is clicked, close the custom dialog*  
dialogButton.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
  
**if** (closeactivity == **true**) {  
dialog.dismiss();  
  
*activity*.finish();  
} **else** {  
dialog.dismiss();  
  
}  
}  
});  
  
dialog.show();  
  
}  
});  
  
}  
**public void** alertSingleButton(**final** String data, **final** EditText edt) {  
*activity*.runOnUiThread(**new** Runnable() {  
**public void** run() {  
**if** (*dialog* == **null**) {  
  
**final** Dialog dialog = **new** Dialog(*activity*);  
dialog.setContentView(R.layout.***activity\_alert***);  
dialog.setTitle(**"Message...."**);  
  
*// set the custom dialog components - text, image and button*  
TextView text = (TextView) dialog.findViewById(R.id.***txt\_data***);  
text.setText(data);  
  
  
Button dialogButton = (Button) dialog  
.findViewById(R.id.***btnok***);  
*// if button is clicked, close the custom dialog*  
dialogButton.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
  
**if** (edt != **null**) {  
dialog.dismiss();  
edt.requestFocus();  
} **else** {  
dialog.dismiss();  
}  
}  
});  
  
dialog.show();  
  
}  
}  
});  
}  
  
  
  
  
**public static boolean** isOnline(Context context) {  
  
ConnectivityManager connectivity = (ConnectivityManager) context  
.getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
**if** (connectivity != **null**) {  
NetworkInfo[] info = connectivity.getAllNetworkInfo();  
**if** (info != **null**)  
**for** (**int** i = 0; i < info.**length**; i++)  
**if** (info[i].getState() == NetworkInfo.State.***CONNECTED***) {  
**return true**;  
}  
}  
**return false**;  
}  
  
  
}

*2.Commonclass*

*In this class we have written some common function we are using for handling ssl exception.*

**package** cs.app.nic.com.seminarclass;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.util.Log;  
  
**import** com.android.volley.toolbox.HurlStack;  
  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.net.HttpURLConnection;  
**import** java.net.URL;  
**import** java.security.KeyManagementException;  
**import** java.security.KeyStore;  
**import** java.security.KeyStoreException;  
**import** java.security.NoSuchAlgorithmException;  
**import** java.security.cert.Certificate;  
**import** java.security.cert.CertificateException;  
**import** java.security.cert.CertificateFactory;  
**import** java.security.cert.X509Certificate;  
  
**import** javax.net.ssl.HostnameVerifier;  
**import** javax.net.ssl.HttpsURLConnection;  
**import** javax.net.ssl.SSLContext;  
**import** javax.net.ssl.SSLSession;  
**import** javax.net.ssl.SSLSocketFactory;  
**import** javax.net.ssl.TrustManager;  
**import** javax.net.ssl.TrustManagerFactory;  
**import** javax.net.ssl.X509TrustManager;  
  
*/\*\**  
*\* Created by asg4.dev13 on 7/11/2017.*  
*\*/*  
  
**public class** Commonclass {  
  
**static** Context *context*;  
**private static** Activity *activity*;  
  
**public** Commonclass(Context con) {  
*context* = con;  
*activity* = (Activity) con;  
}  
  
  
  
**public static** HostnameVerifier getHostnameVerifier() {  
**return new** HostnameVerifier() {  
@Override  
**public boolean** verify(String hostname, SSLSession session) {  
  
  
HostnameVerifier hv = HttpsURLConnection.*getDefaultHostnameVerifier*();  
  
**return true**;  
}  
};  
}  
  
**private static** TrustManager[] getWrappedTrustManagers(TrustManager[] trustManagers) {  
**final** X509TrustManager originalTrustManager = (X509TrustManager) trustManagers[0];  
**return new** TrustManager[]{  
**new** X509TrustManager() {  
**public** X509Certificate[] getAcceptedIssuers() {  
**return** originalTrustManager.getAcceptedIssuers();  
}  
  
**public void** checkClientTrusted(X509Certificate[] certs, String authType) {  
**try** {  
**if** (certs != **null** && certs.**length** >0) {  
certs[0].checkValidity();  
} **else** {  
originalTrustManager.checkClientTrusted(certs, authType);  
}  
} **catch** (CertificateException e) {  
Log.*w*(**"checkClientTrusted"**, e.toString());  
}  
}  
  
**public void** checkServerTrusted(X509Certificate[] certs, String authType) {  
**try** {  
**if** (certs != **null** && certs.**length** >0) {  
certs[0].checkValidity();  
} **else** {  
originalTrustManager.checkServerTrusted(certs, authType);  
}  
} **catch** (CertificateException e) {  
Log.*w*(**"checkServerTrusted"**, e.toString());  
}  
}  
}  
};  
}  
**public static** SSLSocketFactory getSSLSocketFactorys()  
**public static** SSLSocketFactory getSSLSocketFactory()  
**throws** CertificateException, KeyStoreException, IOException, NoSuchAlgorithmException, KeyManagementException {  
  
  
CertificateFactory cf = CertificateFactory.*getInstance*(**"X.509"**);  
InputStream caInput = *activity*.getResources().openRawResource(R.raw.***pglsgdkeralagovin***);  
  
Certificate ca = cf.generateCertificate(caInput);  
caInput.close();  
  
KeyStore keyStore = KeyStore.*getInstance*(**"BKS"**);  
keyStore.load(**null**, **null**);  
keyStore.setCertificateEntry(**"ca"**, ca);  
  
String tmfAlgorithm = TrustManagerFactory.*getDefaultAlgorithm*();  
TrustManagerFactory tmf = TrustManagerFactory.*getInstance*(tmfAlgorithm);  
tmf.init(keyStore);  
  
TrustManager[] wrappedTrustManagers = *getWrappedTrustManagers*(tmf.getTrustManagers());  
  
SSLContext sslContext = SSLContext.*getInstance*(**"TLS"**);  
sslContext.init(**null**, wrappedTrustManagers, **null**);  
  
  
**return** sslContext.getSocketFactory();  
}  
  
**public static** HurlStack hurlStack(){  
  
HurlStack hurlStack = **new** HurlStack() {  
@Override  
**protected** HttpURLConnection createConnection(URL url) **throws** IOException {  
HttpsURLConnection httpsURLConnection = (HttpsURLConnection) **super**.createConnection(url);  
**try** {  
httpsURLConnection.setSSLSocketFactory(Commonclass.*getSSLSocketFactory*());  
httpsURLConnection.setHostnameVerifier(Commonclass.*getHostnameVerifier*());  
} **catch** (Exception e) {  
System.***out***.println(**"Exc"**+e);  
}  
**return** httpsURLConnection;  
}  
};  
  
**return** hurlStack;  
  
}  
}

**Access web service**

Create an Activity "SearchActivity".Here we are giving application number and phone number as input and we will get the status of application.



a. Add below code in "activity\_search.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:orientation="vertical"**>  
  
  
<**include**  
**android:id="@+id/head"**  
**layout="@layout/activity\_title"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
  
/>  
  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_app"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/head"**  
**android:layout\_marginTop="10dp"**>  
  
<**EditText**  
**android:id="@+id/input\_app"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Application Number\*"**/>  
  
</**android.support.design.widget.TextInputLayout**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_mob"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/input\_layout\_app"**  
**android:layout\_marginTop="10dp"**>  
  
<**EditText**  
**android:id="@+id/input\_mob"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Mobile Number Given In The Application\*"**  
**android:inputType="number"**/>  
  
</**android.support.design.widget.TextInputLayout**>  
  
<**Button**  
**android:id="@+id/btnsub"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/input\_layout\_mob"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_marginTop="10dp"**  
**android:background="@drawable/buttoncustom"**  
**android:onClick="onSearchClick"**  
**android:text="Submit"**  
**android:textColor="#000000"**/>  
  
<**LinearLayout**  
**android:id="@+id/details"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_below="@+id/btnsub"**  
**android:orientation="vertical"**  
**android:visibility="invisible"**>  
  
<**TextView**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="10dp"**  
**android:background="#B2DCF8"**  
**android:gravity="center"**  
**android:text="Grievance Details"**  
**android:textColor="#000000"**  
**android:textSize="18dp"**/>  
  
<**TableLayout**  
**android:id="@+id/table"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="20dp"**>  
  
<**TableRow**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:gravity="center|center\_horizontal"**>  
  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Application Number"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
  
<**TextView**  
**android:id="@+id/appnum"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
</**TableRow**>  
  
<**TableRow**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:gravity="center|center\_horizontal"**>  
  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Applicant Name"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
  
<**TextView**  
**android:id="@+id/appname"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
</**TableRow**>  
  
<**TableRow**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:gravity="center|center\_horizontal"**>  
  
  
<**TextView**  
**android:layout\_width="wrap\_content"**  
  
**android:layout\_height="wrap\_content"**  
**android:text="Status"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
  
<**TextView**  
**android:id="@+id/appsta"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:textSize="14dp"**  
**android:textStyle="bold"**/>  
</**TableRow**>  
</**TableLayout**>  
  
<**TableLayout**  
**android:id="@+id/tablelayout\_doc\_list"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:background="#ffffff"**  
  
**android:orientation="vertical"**  
**android:shrinkColumns="\*"**  
**android:stretchColumns="\*"**>  
  
</**TableLayout**>  
</**LinearLayout**>  
  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
</**RelativeLayout**>

In drawable put buttoncustom.xml

b. Add below code in "buttoncustom.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**shape xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:shape="rectangle"**>  
*<!-- Gradient Bg for listrow -->*  
<**gradient**  
**android:startColor="#87CEFA"**  
**android:centerColor="#87CEFA"**  
**android:endColor="#87CEFA"**  
**android:angle="180"**/>  
</**shape**>

c. Add below code in "SearchActivity.java" file.

**package** cs.app.nic.com.seminarclass;  
  
**import** android.app.AlertDialog;  
**import** android.app.ProgressDialog;  
**import** android.content.ActivityNotFoundException;  
**import** android.content.Intent;  
**import** android.net.Uri;  
**import** android.os.AsyncTask;  
**import** android.os.Bundle;  
**import** android.os.Environment;  
**import** android.support.annotation.Nullable;  
**import** android.support.design.widget.TextInputLayout;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.LinearLayout;  
**import** android.widget.TableLayout;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.android.volley.DefaultRetryPolicy;  
**import** com.android.volley.Request;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.VolleyError;  
**import** com.android.volley.toolbox.HurlStack;  
**import** com.android.volley.toolbox.StringRequest;  
**import** com.android.volley.toolbox.Volley;  
**import** com.google.android.gms.analytics.h;  
  
**import** org.apache.http.HttpResponse;  
**import** org.apache.http.client.HttpClient;  
**import** org.apache.http.client.ResponseHandler;  
**import** org.apache.http.client.methods.HttpGet;  
**import** org.apache.http.impl.client.BasicResponseHandler;  
**import** org.apache.http.util.ByteArrayBuffer;  
**import** org.json.JSONArray;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.BufferedInputStream;  
**import** java.io.File;  
**import** java.io.FileOutputStream;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.net.URL;  
**import** java.net.URLEncoder;  
**import** java.util.HashMap;  
**import** java.util.Map;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/13/2017.*  
*\*/*  
  
**public class** SearchActivity **extends** AppCompatActivity {  
**private** EditText **inputapp**, **inputmob**;  
**private** TextInputLayout **inputlayoutapp**, **inputlayoutmob**;  
**private** Button **btnok**;  
TextView **txt\_nam**, **txt\_num**, **txt\_sta**;  
Utils **util**;  
String **appno**, **mobno**;  
ProgressDialog **pdialog**;  
**private** TableLayout **tblelayout\_docList**;  
**int certid**;  
String **no**;  
**static** String *url1*;  
**private** ProgressDialog **DownloadDialog**;  
**protected** String **fileName** = **""**;  
HurlStack **hurlstack**;  
RequestQueue **requestQueue**;  
String **url**;  
Map<String, String>**params**;  
  
@Override  
**protected void** onCreate(@Nullable Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_search***);  
  
**inputapp** = (EditText) findViewById(R.id.***input\_app***);  
**inputmob** = (EditText) findViewById(R.id.***input\_mob***);  
**inputlayoutapp** = (TextInputLayout) findViewById(R.id.***input\_layout\_app***);  
**inputlayoutmob** = (TextInputLayout) findViewById(R.id.***input\_layout\_mob***);  
**btnok** = (Button) findViewById(R.id.***btnsub***);  
**txt\_num** = (TextView) findViewById(R.id.***appnum***);  
**txt\_nam** = (TextView) findViewById(R.id.***appname***);  
**txt\_sta** = (TextView) findViewById(R.id.***appsta***);  
View includedLayout = findViewById(R.id.***head***);  
**tblelayout\_docList** = (TableLayout) findViewById(R.id.***tablelayout\_doc\_list***);  
**tblelayout\_docList**.setVisibility(View.***GONE***);  
  
TextView txttitle = (TextView) includedLayout.findViewById(R.id.***txttitile***);  
txttitle.setText(**"Search"**);  
**util** = **new** Utils(SearchActivity.**this**);  
**hurlstack**=Commonclass.*hurlStack*();  
}  
  
**public void** onSearchClick(View view) {  
  
**appno** = **inputapp**.getText().toString();  
**mobno** = **inputmob**.getText().toString();  
**tblelayout\_docList**.setVisibility(View.***GONE***);  
  
**if** (**inputapp**.getText().toString().equals(**""**)) {  
**util**.alertButton(**"Please Enter Application number"**, **false**);  
  
} **else if** (**inputmob**.getText().toString().equals(**""**)) {  
**util**.alertButton(**"Please Enter Mobile Number"**, **false**);  
} **else** {  
  
**if** (Utils.*isOnline*(SearchActivity.**this**)) {  
**pdialog** = **new** ProgressDialog(SearchActivity.**this**);  
  
**pdialog** = **new** ProgressDialog(SearchActivity.**this**);  
**pdialog**.setMessage(**"Loading..."**);  
**pdialog**.setCancelable(**false**);  
**pdialog**.show();  
  
  
LoadService();  
  
} **else** {  
**pdialog**.dismiss();  
  
Toast.*makeText*(getApplicationContext(),  
**"No Internet Connection!!!"**, Toast.***LENGTH\_SHORT***).show();  
}  
}  
  
}  
  
**private void** LoadService() {

**url** = "http://103.251.43.122/pgmobapp/service/mobileapp/getstatus?mobileNo=" + mobno + "&applNo=" + appno;

if (**url**.contains("https")) {

**requestQueue** = Volley.newRequestQueue(this, hurlstack);

} else {

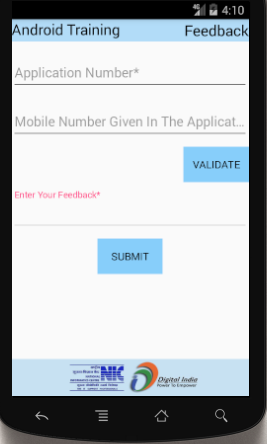
**requestQueue** = Volley.newRequestQueue(this);

}

**final** StringRequest jsonObjectRequest = **new** StringRequest(Request.Method.***GET***, **url**, **new** Response.Listener<String>() {

**public void** onResponse(String response) {  
**try** {  
**pdialog**.dismiss();  
JSONObject json = **new** JSONObject(response);  
  
**if** (json.has(**"Status Response"**)) {  
JSONObject object1 = json.getJSONObject(**"Status Response"**);  
JSONArray object11 = object1.getJSONArray(**"Status"**);  
  
**if** (!(object11.equals(**null**) || object11.equals(**""**))) {  
  
**no** = (String) object11.getJSONObject(0).get(**"Application No"**);  
String apname = (String) object11.getJSONObject(1).get(**"Applicant Name"**);  
String stat = (String) object11.getJSONObject(2).get(**"Status"**);  
**certid**=object11.getJSONObject(3).getInt(**"CertId"**);  
  
System.***out***.println(**"number is"**+ **no**);  
  
  
LinearLayout table = (LinearLayout) findViewById(R.id.***details***);  
table.setVisibility(View.***VISIBLE***);  
**txt\_nam**.setText(**":"**+ apname);  
**txt\_sta**.setText(**":"**+ stat);  
**txt\_num**.setText(**":"**+ **no**);  
**inputapp**.setText(**""**);  
**inputmob**.setText(**""**);  
  
**if**(stat.equals(**"Redressed/Disposed"**)){  
**tblelayout\_docList**.setVisibility(View.***VISIBLE***);  
  
}  
  
}  
} **else** {  
  
JSONObject st = json.getJSONObject(**"Result"**);  
String re = (String) st.get(**"error"**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(SearchActivity.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(re);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
}  
}  
  
**catch** (Exception e) {  
  
**pdialog**.dismiss();  
e.printStackTrace();  
}  
}  
}, **new** Response.ErrorListener() {  
@Override  
**public void** onErrorResponse(VolleyError e) {  
e.printStackTrace();  
}  
});  
  
jsonObjectRequest.setRetryPolicy(**new** DefaultRetryPolicy(10000,  
DefaultRetryPolicy.***DEFAULT\_MAX\_RETRIES***,  
DefaultRetryPolicy.***DEFAULT\_BACKOFF\_MULT***));  
**requestQueue**.getCache().clear();  
**requestQueue**.add(jsonObjectRequest);  
}  
  
  
 }

**Feedback Form**

****

This is for submitting feedback against an already filed application. First check whether the application number exists or not. If exists, submit the feedback.

Create an Activity "Feedback".

a. Add below code in "activity\_feed" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:orientation="vertical"android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**>  
  
<**include**  
**android:id="@+id/head"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_width="fill\_parent"**  
**layout="@layout/activity\_title"**  
  
/>  
  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_app"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/head"**>  
  
<**EditText**  
**android:id="@+id/input\_app"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Application Number\*"**/>  
  
</**android.support.design.widget.TextInputLayout**>  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_mob"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/input\_layout\_app"**>  
  
<**EditText**  
**android:id="@+id/input\_mob"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Mobile Number Given In The Application\*"**  
**android:inputType="number"**/>  
  
</**android.support.design.widget.TextInputLayout**>  
  
<**Button**  
**android:id="@+id/btnsub"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Validate"**  
**android:layout\_below="@+id/input\_layout\_mob"**  
**android:textColor="#000000"**  
**android:background="@drawable/buttoncustom"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_alignParentRight="true"**/>  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/input\_layout\_feed"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/btnsub"**  
>  
  
<**EditText**  
**android:id="@+id/input\_feed"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:hint="Enter Your Feedback\*"**  
  
  
/>  
</**android.support.design.widget.TextInputLayout**>  
<**Button**  
**android:id="@+id/btnfeed"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Submit"**  
**android:textColor="#000000"**  
**android:background="@drawable/buttoncustom"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/input\_layout\_feed"**  
**android:layout\_centerHorizontal="true"**  
**android:enabled="false"**/>  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
  
</**RelativeLayout**>

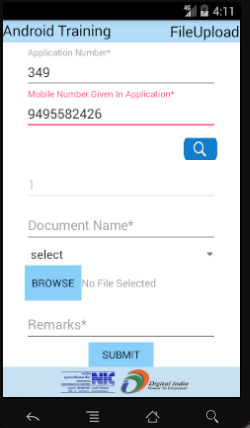
b. Add below code in "Feedback.java" file.

**package** nic.kerala.training;  
  
**import** android.app.ProgressDialog;  
**import** android.content.Context;  
**import** android.os.Bundle;  
**import** android.support.annotation.Nullable;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.view.inputmethod.InputMethodManager;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.LinearLayout;  
**import** android.widget.Toast;  
  
**import** com.android.volley.DefaultRetryPolicy;  
**import** com.android.volley.Request;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.VolleyError;  
**import** com.android.volley.toolbox.HurlStack;  
**import** com.android.volley.toolbox.StringRequest;  
**import** com.android.volley.toolbox.Volley;  
  
**import** java.net.URLEncoder;  
**import** java.util.HashMap;  
**import** java.util.Map;  
  
*/\*\**  
*\* Created by asg4.dev13 on 4/19/2017.*  
*\*/*  
  
**public class** Feedback**extends** AppCompatActivity {  
EditText **edt\_mobno**, **edt\_appno**, **edt\_feed**;  
Button **btn\_veri**, **btn\_feed**;  
Utils **util**;  
**private** ProgressDialog **pDialog** = **null**;  
LinearLayout **line**;  
HurlStack **hurlstack**;  
RequestQueue **requestQueue**;  
String **url**;  
Map<String, String>**params**;  
String **feed**;  
String **apNO**;  
String **mobno**;  
  
@Override  
**protected void** onCreate(@Nullable Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_feed***);  
**btn\_veri** = (Button) findViewById(R.id.***btnsub***);  
**edt\_appno** = (EditText) findViewById(R.id.***input\_app***);  
**edt\_mobno** = (EditText) findViewById(R.id.***input\_mob***);  
**edt\_feed** = (EditText) findViewById(R.id.***input\_feed***);  
**btn\_feed** = (Button) findViewById(R.id.***btnfeed***);  
**line** = (LinearLayout) findViewById(R.id.***linear***);  
**hurlstack** = Commonclass.*hurlStacks*();  
**util** = **new** Utils(Feedback.**this**);  
  
  
**btn\_veri**.setOnClickListener(**new** View.OnClickListener() {  
  
  
@Override  
**public void** onClick(View v) {  
  
hidekeyboard();  
**if** (**edt\_appno**.getText().toString().equals(**""**) || **edt\_mobno**.getText().toString().equals(**""**)) {  
**util**.alertSingleButton(**"Please Enter the details"**, **null**);  
} **else** {  
  
**apNO** = **edt\_appno**.getText().toString().trim();  
**mobno** = **edt\_mobno**.getText().toString().trim();  
  
  
**if** (Utils.*isOnline*(Feedback.**this**)) {  
  
  
**pDialog** = **new** ProgressDialog(Feedback.**this**);  
**pDialog**.setMessage(**"Loading..."**);  
**pDialog**.setCancelable(**false**);  
**pDialog**.show();  
verify();  
  
  
} **else** {  
  
Toast.*makeText*(getApplicationContext(),  
**"No Internet Connection!!!"**, Toast.***LENGTH\_SHORT***).show();  
}  
  
}  
}  
  
  
});  
  
  
**btn\_feed**.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
hidekeyboard();  
  
**if** (**edt\_feed**.getText().toString().equals(**""**)) {  
**util**.alertSingleButton(**"Please Enter feedback"**, **null**);  
} **else** {  
**feed** = **edt\_feed**.getText().toString();  
**apNO** = **edt\_appno**.getText().toString().trim();  
  
  
**try** {  
  
  
**if** (Utils.*isOnline*(Feedback.**this**)) {  
  
  
**pDialog** = **new** ProgressDialog(Feedback.**this**);  
**pDialog**.setMessage(**"Loading..."**);  
**pDialog**.setCancelable(**false**);  
**pDialog**.show();  
  
save();  
  
  
} **else** {  
  
Toast.*makeText*(getApplicationContext(),  
**"No Internet Connection!!!"**, Toast.***LENGTH\_SHORT***).show();  
}  
} **catch** (Exception e) {  
e.printStackTrace();  
}  
}  
}  
});  
  
  
}  
  
**private void** hidekeyboard() {  
  
InputMethodManager imm = (InputMethodManager) getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(getCurrentFocus().getWindowToken(),  
InputMethodManager.***RESULT\_UNCHANGED\_SHOWN***);  
}  
  
**private void** save() {  
**url** = **"http://103.251.43.122/pgmobapp/service/mobileapp/savePgFeedback"**;  
**if** (**url**.contains(**"https"**)) {  
**requestQueue** = Volley.*newRequestQueue*(**this**, **hurlstack**);  
} **else** {  
**requestQueue** = Volley.*newRequestQueue*(**this**);  
}  
  
**final** StringRequest jsonObjectRequest = **new** StringRequest(Request.Method.***POST***, **url**, **new** Response.Listener<String>() {  
  
**public void** onResponse(String response) {  
**try** {  
  
String Output = (response.trim() == **null**) ? **"submitted"**: response.trim();  
**if** (Output.equals(**"Feedback submitted Successfully"**)) {  
**pDialog**.dismiss();  
**util**.alertButton(Output, **true**);  
  
} **else** {  
**pDialog**.dismiss();  
**util**.alertButton(Output, **true**);  
  
}  
} **catch** (Exception e) {  
**pDialog**.dismiss();  
e.printStackTrace();  
}  
}  
}, **new** Response.ErrorListener() {  
@Override  
**public void** onErrorResponse(VolleyError e) {  
e.printStackTrace();  
}  
}) {  
@Override  
  
**public** Map<String, String> getParams() {  
**try** {  
**params** = **new** HashMap<>();  
**params**.put(**"apNo"**, URLEncoder  
.*encode*(**apNO**, **"UTF-8"**));  
**params**.put(**"Feedback"**, **feed**);  
  
} **catch** (Exception e) {  
  
}  
**return params**;  
}  
};  
  
jsonObjectRequest.setRetryPolicy(**new** DefaultRetryPolicy(10000,  
DefaultRetryPolicy.***DEFAULT\_MAX\_RETRIES***,  
DefaultRetryPolicy.***DEFAULT\_BACKOFF\_MULT***));  
**requestQueue**.getCache().clear();  
**requestQueue**.add(jsonObjectRequest);  
}  
  
**private void** verify() {  
**url** = **"http://103.251.43.122/pgmobapp/service/mobileapp/VerifyPgFeedback"**;  
**if** (**url**.contains(**"https"**)) {  
**requestQueue** = Volley.*newRequestQueue*(**this**, **hurlstack**);  
} **else** {  
**requestQueue** = Volley.*newRequestQueue*(**this**);  
}  
  
**final** StringRequest jsonObjectRequest = **new** StringRequest(Request.Method.***POST***, **url**, **new** Response.Listener<String>() {  
  
**public void** onResponse(String response) {  
**try** {  
String output = (response.trim() == **null**) ? **"Submit"**: response.trim();  
System.***out***.println(**"output"**+ output);  
**if** (output.equals(**"Success"**)) {  
**pDialog**.dismiss();  
**util**.alertButton(**"Now You Can Enter Your Feedback"**, **false**);  
  
**btn\_veri**.setEnabled(**false**);  
  
**edt\_appno**.setEnabled(**false**);  
**edt\_mobno**.setEnabled(**false**);  
**line**.setVisibility(View.***VISIBLE***);  
  
  
} **else** {  
**pDialog**.dismiss();  
**util**.alertButton(output, **true**);  
  
}  
} **catch** (Exception e) {  
**pDialog**.dismiss();  
e.printStackTrace();  
**util**.alertButton(**"Connection Error"**,**true**);  
}  
}  
}, **new** Response.ErrorListener() {  
@Override  
**public void** onErrorResponse(VolleyError e) {  
**pDialog**.dismiss();  
**util**.alertButton(**"Connection Error"**,**true**);  
e.printStackTrace();  
}  
}) {  
@Override  
  
**public** Map<String, String> getParams() {  
**try** {  
**params** = **new** HashMap<>();  
**params**.put(**"apNo"**, URLEncoder.*encode*(**apNO**, **"UTF-8"**));  
**params**.put(**"mbNo"**, URLEncoder.*encode*(**mobno**, **"UTF-8"**));  
  
} **catch** (Exception e) {  
  
}  
**return params**;  
}  
};  
  
jsonObjectRequest.setRetryPolicy(**new** DefaultRetryPolicy(10000,  
DefaultRetryPolicy.***DEFAULT\_MAX\_RETRIES***,  
DefaultRetryPolicy.***DEFAULT\_BACKOFF\_MULT***));  
**requestQueue**.getCache().clear();  
**requestQueue**.add(jsonObjectRequest);  
}  
  
  
}

**File Upload**

Documents such as image, pdf can be uploaded to the server.

Create an Activity "FileUpload".



a. Add below code in "activity\_fileupload" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**xmlns:tools="http://schemas.android.com/tools"**  
**android:id="@+id/activity\_application"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:weightSum="10"**  
**android:background="#fff"**  
**android:orientation="vertical"**  
>  
  
  
<**include**  
**android:id="@+id/head"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_width="fill\_parent"**  
**layout="@layout/activity\_title"**  
  
/>  
  
<**ScrollView**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_weight="7.3"**  
**android:layout\_below="@+id/head"**  
>  
  
<**LinearLayout**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:layout\_weight="7"**  
**android:background="#FFFF"**  
**android:orientation="vertical"**>  
  
  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/TextInputLayoutfbapplno"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginEnd="30dp"**  
**android:layout\_marginLeft="30dp"**  
**android:layout\_marginRight="30dp"**  
**android:layout\_marginStart="30dp"**  
**android:layout\_marginTop="15dp"**  
**android:background="#FFFF"**>  
  
<**EditText**  
**android:id="@+id/fbapplno"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:inputType="phone"**  
**android:maxLength="5"**  
  
**android:hint="Application Number\*"**></**EditText**>  
</**android.support.design.widget.TextInputLayout**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/TextInputLayoutfbmblno"**  
  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_marginEnd="30dp"**  
**android:layout\_marginLeft="30dp"**  
**android:layout\_marginRight="30dp"**  
**android:layout\_marginStart="30dp"**  
**android:background="#FFFF"**>  
  
<**EditText**  
**android:id="@+id/fbmblno"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:inputType="phone"**  
**android:maxLength="10"**  
**android:hint="Mobile Number Given In Application\*"**></**EditText**>  
</**android.support.design.widget.TextInputLayout**>  
  
<**ImageButton**  
**android:id="@+id/button\_verify"**  
**android:onClick="onVerify"**  
**android:layout\_width="45dp"**  
**android:layout\_height="30dp"**  
**android:layout\_gravity="right"**  
**android:background="@drawable/btnsearch"**  
**android:layout\_marginEnd="30dp"**  
**android:layout\_marginRight="30dp"**  
**android:layout\_marginTop="10dp"**  
/>  
  
<**LinearLayout**  
**android:id="@+id/linear"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:orientation="vertical"**  
**android:gravity="center"**>  
  
  
<**TableLayout**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:layout\_marginEnd="30dp"**  
**android:layout\_marginLeft="30dp"**  
**android:layout\_marginRight="30dp"**  
**android:layout\_marginStart="30dp"**  
**android:background="#FFFF"**  
**android:id="@+id/tableinsert1"**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
  
**android:background="#FFFF"**>  
<**EditText**  
**android:id="@+id/slno1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:text="1"**></**EditText**>  
  
</**android.support.design.widget.TextInputLayout**>  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/TextInputLayoutdocname1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
  
**android:background="#FFFF"**>  
<**EditText**  
**android:id="@+id/docname1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:inputType="text"**  
**android:hint="Document Name\*"**></**EditText**>  
  
</**android.support.design.widget.TextInputLayout**>  
  
<**Spinner**  
**android:id="@+id/attchtype1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="30dp"**  
**android:hint="Attach Type"**  
>  
</**Spinner**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
  
**android:background="#FFFF"**  
**android:checkableBehavior="single"**  
>  
<**TableRow**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**>  
  
<**Button**  
**android:id="@+id/browse1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:onClick="browse"**  
**android:background="@drawable/buttoncustom"**  
**android:textColor="#000000"**  
**android:layout\_weight="12"**  
**android:text="Browse"**></**Button**>  
<**TextView**  
**android:id="@+id/file\_path1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_weight="5"**  
**android:hint="No File Selected"**></**TextView**>  
</**TableRow**>  
</**android.support.design.widget.TextInputLayout**>  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/TextInputLayoutattachremark1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:background="#FFFF"**>  
<**EditText**  
**android:id="@+id/attachremark1"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:inputType="text"**  
**android:hint="Remarks\*"**></**EditText**>  
  
</**android.support.design.widget.TextInputLayout**>  
</**TableLayout**>  
  
  
<**Button**  
**android:id="@+id/filesuploades"**  
**android:layout\_width="wrap\_content"**  
**android:onClick="FinalFileUpload"**  
**android:layout\_height="33dp"**  
**android:text="Submit"**  
**android:textColor="#000000"**  
**android:background="@drawable/buttoncustom"**  
**android:layout\_marginBottom="40dp"**/>  
</**LinearLayout**>  
  
</**LinearLayout**>  
</**ScrollView**>  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="40dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
</**RelativeLayout**>

**b**. Add below code in "Filepath.java" file.

import android.content.ContentUris;

import android.content.Context;

import android.database.Cursor;

import android.net.Uri;

import android.os.Build;

import android.os.Environment;

import android.provider.DocumentsContract;

import android.provider.MediaStore;

public class FilePath

{

/\*\*

\* Method for return file path of Gallery image

\*

\* @param context

\* @param uri

\* @return path of the selected image file from gallery

\*/

public static String getPath(final Context context, final Uri uri)

{

//check here to KITKAT or new version

final boolean isKitKat = Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT;

// DocumentProvider

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT) {

if (isKitKat && DocumentsContract.isDocumentUri(context, uri)) {

// ExternalStorageProvider

if (isExternalStorageDocument(uri)) {

final String docId = DocumentsContract.getDocumentId(uri);

final String[] split = docId.split(":");

final String type = split[0];

if ("primary".equalsIgnoreCase(type)) {

return Environment.getExternalStorageDirectory() + "/" + split[1];

}

}

//DownloadsProvider

else if (isDownloadsDocument(uri)) {

final String id = DocumentsContract.getDocumentId(uri);

final Uri contentUri = ContentUris.withAppendedId(

Uri.parse("content://downloads/public\_downloads"), Long.valueOf(id));

return getDataColumn(context, contentUri, null, null);

}

// MediaProvider

else if (isMediaDocument(uri)) {

final String docId = DocumentsContract.getDocumentId(uri);

final String[] split = docId.split(":");

final String type = split[0];

Uri contentUri = null;

if ("image".equals(type)) {

contentUri = MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI;

} else if ("video".equals(type)) {

contentUri = MediaStore.Video.Media.EXTERNAL\_CONTENT\_URI;

} else if ("audio".equals(type)) {

contentUri = MediaStore.Audio.Media.EXTERNAL\_CONTENT\_URI;

}

final String selection = "\_id=?";

final String[] selectionArgs = new String[] {

split[1]

};

return getDataColumn(context, contentUri, selection, selectionArgs);

}

}

// MediaStore (and general)

else if ("content".equalsIgnoreCase(uri.getScheme())) {

// Return the remote address

if (isGooglePhotosUri(uri))

return uri.getLastPathSegment();

return getDataColumn(context, uri, null, null);

}

// File

else if ("file".equalsIgnoreCase(uri.getScheme())) {

return uri.getPath();

}

}

return null;

}

/\*\*

\* Get the value of the data column for this Uri. This is useful for

\* MediaStore Uris, and other file-based ContentProviders.

\*

\* @param context The context.

\* @param uri The Uri to query.

\* @param selection (Optional) Filter used in the query.

\* @param selectionArgs (Optional) Selection arguments used in the query.

\* @return The value of the \_data column, which is typically a file path.

\*/

public static String getDataColumn(Context context, Uri uri, String selection,

String[] selectionArgs) {

Cursor cursor = null;

final String column = "\_data";

final String[] projection = {

column

};

try {

cursor = context.getContentResolver().query(uri, projection, selection, selectionArgs,

null);

if (cursor != null && cursor.moveToFirst()) {

final int index = cursor.getColumnIndexOrThrow(column);

return cursor.getString(index);

}

} finally {

if (cursor != null)

cursor.close();

}

return null;

}

/\*\*

\* @param uri The Uri to check.

\* @return Whether the Uri authority is ExternalStorageProvider.

\*/

public static boolean isExternalStorageDocument(Uri uri) {

return "com.android.externalstorage.documents".equals(uri.getAuthority());

}

/\*\*

\* @param uri The Uri to check.

\* @return Whether the Uri authority is DownloadsProvider.

\*/

public static boolean isDownloadsDocument(Uri uri) {

return "com.android.providers.downloads.documents".equals(uri.getAuthority());

}

/\*\*

\* @param uri The Uri to check.

\* @return Whether the Uri authority is MediaProvider.

\*/

public static boolean isMediaDocument(Uri uri) {

return "com.android.providers.media.documents".equals(uri.getAuthority());

}

/\*\*

\* @param uri The Uri to check.

\* @return Whether the Uri authority is Google Photos.

\*/

public static boolean isGooglePhotosUri(Uri uri) {

return "com.google.android.apps.photos.content".equals(uri.getAuthority());

}

}

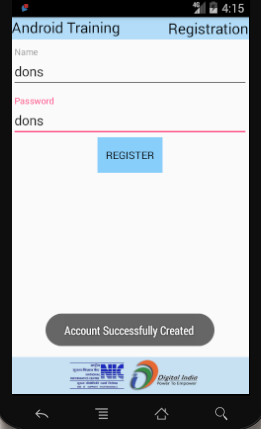
**C**. Add below code in "FileUpload.java" file.

**package** nic.kerala.training;  
  
**import** android.Manifest;  
**import** android.app.ProgressDialog;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.database.Cursor;  
**import** android.net.Uri;  
**import** android.os.Bundle;  
**import** android.provider.MediaStore;  
**import** android.support.design.widget.TextInputLayout;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.text.TextUtils;  
**import** android.view.MotionEvent;  
**import** android.view.View;  
**import** android.view.inputmethod.InputMethodManager;  
**import** android.widget.AdapterView;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.ImageButton;  
**import** android.widget.LinearLayout;  
**import** android.widget.Spinner;  
**import** android.widget.TableLayout;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.android.volley.DefaultRetryPolicy;  
**import** com.android.volley.Request;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.VolleyError;  
**import** com.android.volley.toolbox.HurlStack;  
**import** com.android.volley.toolbox.StringRequest;  
**import** com.android.volley.toolbox.Volley;  
  
**import** org.json.JSONArray;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.ByteArrayOutputStream;  
**import** java.io.DataOutputStream;  
**import** java.io.File;  
**import** java.io.FileInputStream;  
**import** java.io.FileNotFoundException;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.net.HttpURLConnection;  
**import** java.net.URLEncoder;  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
**import** java.util.regex.Pattern;  
  
**public class** FileUpload**extends** AppCompatActivity {  
  
**private int click** = 0;  
Map<String, String>**params**;  
**private int clickremove** = 0;  
**private** Spinner **attchtype1**, **attchtype2**, **attchtype3**, **attchtype4**, **attchtype5**;  
*//private HTTPURLConnection service;*  
**private** TableLayout **tableinsert1**, **tableinsert2**, **tableinsert3**, **tableinsert4**, **tableinsert5**, **one**;  
*/\*private TextView attachno1,attachno2,attachno3,attachno4,attachno5,*  
*selected1,selected2,selected3,selected4,selected5,\*/*  
**private** TextView **filePath**, **filePath1**, **filePath2**, **filePath3**, **filePath4**, **filePath5**;  
**private** ProgressDialog **pDialog** = **null**;  
String **encodedString**;  
*//private TextView attachno,DocumentName,AttachmentType,selected,attachRemarks,captcha;*  
**private** LinearLayout **linear**;  
**private static final int *SELECT\_FILE1*** = 1;  
**private static final int *SELECT\_FILE2*** = 2;  
**private static final int *SELECT\_FILE3*** = 3;  
**private static final int *SELECT\_FILE4*** = 4;  
**private static final int *SELECT\_FILE5*** = 5;  
  
String **url**;  
HurlStack **hurlstack**;  
RequestQueue **requestQueue**;  
  
**private** String **selectedPath1** = **""**, **selectedPath2** = **""**, **selectedPath3** = **""**, **selectedPath4** = **""**, **selectedPath5** = **""**;  
**private** String **selectedPath1s** = **""**, **selectedPath2s** = **""**, **selectedPath3s** = **""**, **selectedPath4s** = **""**, **selectedPath5s** = **""**;  
  
Utils **util**;  
**private** TextInputLayout **TextInputLayoutdocname1**, **TextInputLayoutdocname2**, **TextInputLayoutdocname3**,  
**TextInputLayoutdocname4**, **TextInputLayoutdocname5**,  
**TextInputLayoutattachremark1**, **TextInputLayoutattachremark2**, **TextInputLayoutattachremark3**,  
**TextInputLayoutattachremark4**, **TextInputLayoutattachremark5**;  
  
**private** EditText **applno**, **mblno**;  
**private** String **apNo**, **mbNo**;  
**private** TextView **applicationno**, **mobileno**;  
**private boolean flag**;  
**private** String **msg** = **""**;  
**private** ProgressDialog **progressDialog**;  
**private** EditText **slno1**, **slno2**, **slno3**, **slno4**, **slno5**,  
**attachremark1**, **attachremark2**, **attachremark3**, **attachremark4**, **attachremark5**,  
**docname1**, **docname2**, **docname3**, **docname4**, **docname5**;  
**final int MY\_PERMISSIONS\_REQUEST\_WRITE\_EXTERNAL\_STORAGE** = 1;  
**private** String **attchtypecode1** = **""**, **attchtypecode2** = **""**, **attchtypecode3** = **""**, **attchtypecode4** = **""**, **attchtypecode5** = **""**,  
**attchtypecode1s** = **""**, **attchtypecode2s** = **""**, **attchtypecode3s** = **""**, **attchtypecode4s** = **""**, **attchtypecode5s** = **""**;  
**private** String **applNo**, **mobileNumber**;  
**private** String **slno1s**, **slno2s**, **slno3s**, **slno4s**, **slno5s**,  
**attachremark1s**, **attachremark2s**, **attachremark3s**, **attachremark4s**, **attachremark5s**,  
**docname1s**, **docname2s**, **docname3s**, **docname4s**, **docname5s**;  
String **attachrem**, **attachtypes**, **docnames**, **slnos**;  
  
**private** String **sprattchtype**;  
**private** Button **filesuploades**;  
**private** Button **addbutton**, **removebutton**,  
**browse1**, **browse2**, **browse3**, **browse4**, **browse5**;  
ImageButton **button\_verify**;  
  
**private static final int *PICK\_FILE\_REQUEST*** = 1;  
**private static final** String ***TAG*** = FileUpload.**class**.getSimpleName();  
**private** String **selectedFilePath**;  
**private** ProgressDialog **dialog**;  
**private int serverResponseCode** = 0;  
**private** File **selectedFile**;  
**private** HttpURLConnection **connection**;  
**private** DataOutputStream **dataOutputStream**;  
**private** String **Filename** = **""**;  
**private** String **aplno**, **mobno**;  
String **Typename**;  
  
List<String>**Mylist** = **new** ArrayList<String>();  
  
  
**private** TextInputLayout **TextInputLayoutfbmblno**, **TextInputLayoutfbapplno**;  
  
  
@Override  
**protected void** onCreate(Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_file\_upload***);  
  
**TextInputLayoutfbapplno** = (TextInputLayout) findViewById(R.id.***TextInputLayoutfbapplno***);  
**util** = **new** Utils(FileUpload.**this**);  
**TextInputLayoutfbmblno** = (TextInputLayout) findViewById(R.id.***TextInputLayoutfbmblno***);  
  
  
**hurlstack** = Commonclass.*hurlStack*();  
**slno1** = (EditText) findViewById(R.id.***slno1***);  
**slno2** = (EditText) findViewById(R.id.***slno2***);  
**slno3** = (EditText) findViewById(R.id.***slno3***);  
**slno4** = (EditText) findViewById(R.id.***slno4***);  
**slno5** = (EditText) findViewById(R.id.***slno5***);  
**attachremark1** = (EditText) findViewById(R.id.***attachremark1***);  
**attachremark2** = (EditText) findViewById(R.id.***attachremark2***);  
**attachremark3** = (EditText) findViewById(R.id.***attachremark3***);  
**attachremark4** = (EditText) findViewById(R.id.***attachremark4***);  
**attachremark5** = (EditText) findViewById(R.id.***attachremark5***);  
**docname1** = (EditText) findViewById(R.id.***docname1***);  
**docname2** = (EditText) findViewById(R.id.***docname2***);  
**docname3** = (EditText) findViewById(R.id.***docname3***);  
**docname4** = (EditText) findViewById(R.id.***docname4***);  
**docname5** = (EditText) findViewById(R.id.***docname5***);  
**attchtype1** = (Spinner) findViewById(R.id.***attchtype1***);  
**attchtype2** = (Spinner) findViewById(R.id.***attchtype2***);  
**attchtype3** = (Spinner) findViewById(R.id.***attchtype3***);  
**attchtype4** = (Spinner) findViewById(R.id.***attchtype4***);  
**attchtype5** = (Spinner) findViewById(R.id.***attchtype5***);  
**tableinsert1** = (TableLayout) findViewById(R.id.***tableinsert1***);  
**tableinsert2** = (TableLayout) findViewById(R.id.***tableinsert2***);  
**tableinsert3** = (TableLayout) findViewById(R.id.***tableinsert3***);  
**tableinsert4** = (TableLayout) findViewById(R.id.***tableinsert4***);  
**tableinsert5** = (TableLayout) findViewById(R.id.***tableinsert5***);  
**button\_verify** = (ImageButton) findViewById(R.id.***button\_verify***);  
**linear** = (LinearLayout) findViewById(R.id.***linear***);  
  
**TextInputLayoutdocname1** = (TextInputLayout) findViewById(R.id.***TextInputLayoutdocname1***);  
**TextInputLayoutdocname2** = (TextInputLayout) findViewById(R.id.***TextInputLayoutdocname2***);  
**TextInputLayoutdocname3** = (TextInputLayout) findViewById(R.id.***TextInputLayoutdocname3***);  
**TextInputLayoutdocname4** = (TextInputLayout) findViewById(R.id.***TextInputLayoutdocname4***);  
**TextInputLayoutdocname5** = (TextInputLayout) findViewById(R.id.***TextInputLayoutdocname5***);  
**TextInputLayoutattachremark1** = (TextInputLayout) findViewById(R.id.***TextInputLayoutattachremark1***);  
**TextInputLayoutattachremark2** = (TextInputLayout) findViewById(R.id.***TextInputLayoutattachremark2***);  
**TextInputLayoutattachremark3** = (TextInputLayout) findViewById(R.id.***TextInputLayoutattachremark3***);  
**TextInputLayoutattachremark4** = (TextInputLayout) findViewById(R.id.***TextInputLayoutattachremark4***);  
**TextInputLayoutattachremark5** = (TextInputLayout) findViewById(R.id.***TextInputLayoutattachremark5***);  
**filePath1** = (TextView) findViewById(R.id.***file\_path1***);  
**filePath2** = (TextView) findViewById(R.id.***file\_path2***);  
**filePath3** = (TextView) findViewById(R.id.***file\_path3***);  
**filePath4** = (TextView) findViewById(R.id.***file\_path4***);  
**filePath5** = (TextView) findViewById(R.id.***file\_path5***);  
**applno** = (EditText) findViewById(R.id.***fbapplno***);  
**mblno** = (EditText) findViewById(R.id.***fbmblno***);  
View buttonAdd = findViewById(R.id.***addbutton***);  
View browse1 = findViewById(R.id.***browse1***);  
View browse2 = findViewById(R.id.***browse2***);  
View browse3 = findViewById(R.id.***browse3***);  
View browse4 = findViewById(R.id.***browse4***);  
View browse5 = findViewById(R.id.***browse5***);  
View buttonRemove = findViewById(R.id.***removebutton***);  
**filesuploades** = (Button) findViewById(R.id.***filesuploades***);  
**tableinsert1**.setVisibility(View.***VISIBLE***);  
**tableinsert2**.setVisibility(View.***GONE***);  
**tableinsert3**.setVisibility(View.***GONE***);  
**tableinsert4**.setVisibility(View.***GONE***);  
**tableinsert5**.setVisibility(View.***GONE***);  
**slno1**.setText(**"1"**);  
**slno1**.setEnabled(**false**);  
**slno2**.setText(**"2"**);  
**slno2**.setEnabled(**false**);  
**slno3**.setText(**"3"**);  
**slno3**.setEnabled(**false**);  
**slno4**.setText(**"4"**);  
**slno4**.setEnabled(**false**);  
**slno5**.setText(**"5"**);  
**slno5**.setEnabled(**false**);  
  
  
**attchtype1**.setOnTouchListener(**new** View.OnTouchListener() {  
  
@Override  
**public boolean** onTouch(View v, MotionEvent event) {  
InputMethodManager imm = (InputMethodManager) getApplicationContext().getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(**attchtype1**.getWindowToken(), 0);  
**return false**;  
}  
});  
  
**attchtype2**.setOnTouchListener(**new** View.OnTouchListener() {  
  
@Override  
**public boolean** onTouch(View v, MotionEvent event) {  
InputMethodManager imm = (InputMethodManager) getApplicationContext().getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(**attchtype2**.getWindowToken(), 0);  
**return false**;  
}  
});  
  
**attchtype3**.setOnTouchListener(**new** View.OnTouchListener() {  
  
@Override  
**public boolean** onTouch(View v, MotionEvent event) {  
InputMethodManager imm = (InputMethodManager) getApplicationContext().getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(**attchtype3**.getWindowToken(), 0);  
**return false**;  
}  
});  
  
**attchtype4**.setOnTouchListener(**new** View.OnTouchListener() {  
  
@Override  
**public boolean** onTouch(View v, MotionEvent event) {  
InputMethodManager imm = (InputMethodManager) getApplicationContext().getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(**attchtype4**.getWindowToken(), 0);  
**return false**;  
}  
});  
  
**attchtype5**.setOnTouchListener(**new** View.OnTouchListener() {  
  
@Override  
**public boolean** onTouch(View v, MotionEvent event) {  
InputMethodManager imm = (InputMethodManager) getApplicationContext().getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(**attchtype5**.getWindowToken(), 0);  
**return false**;  
}  
});  
loadAttachment1();  
loadAttachment2();  
loadAttachment3();  
loadAttachment4();  
loadAttachment5();  
**linear**.setVisibility(View.***INVISIBLE***);  
  
browse1.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
  
openGallery(***SELECT\_FILE1***);  
}  
});  
browse2.setOnClickListener(**new** View.OnClickListener() {  
@Override  
  
**public void** onClick(View v) {  
  
openGallery(***SELECT\_FILE2***);  
}  
});  
browse3.setOnClickListener(**new** View.OnClickListener() {  
@Override  
  
**public void** onClick(View v) {  
  
openGallery(***SELECT\_FILE3***);  
}  
});  
browse4.setOnClickListener(**new** View.OnClickListener() {  
@Override  
  
**public void** onClick(View v) {  
  
openGallery(***SELECT\_FILE4***);  
}  
});  
browse5.setOnClickListener(**new** View.OnClickListener() {  
@Override  
  
**public void** onClick(View v) {  
  
openGallery(***SELECT\_FILE5***);  
}  
});  
  
}  
  
  
**public void** openGallery(**int** req\_code) {  
**if** (ActivityCompat.*checkSelfPermission*(FileUpload.**this**, Manifest.permission.***WRITE\_EXTERNAL\_STORAGE***) != PackageManager.***PERMISSION\_GRANTED***) {  
  
ActivityCompat.*requestPermissions*(FileUpload.**this**,  
**new** String[]{Manifest.permission.***WRITE\_EXTERNAL\_STORAGE***},  
**MY\_PERMISSIONS\_REQUEST\_WRITE\_EXTERNAL\_STORAGE**);  
  
}  
  
  
Intent intent = **new** Intent();  
intent.setType(**"\*/\*"**);  
intent.setAction(Intent.***ACTION\_GET\_CONTENT***);  
startActivityForResult(Intent.*createChooser*(intent, **"Select file to upload "**), req\_code);  
  
  
}

**public void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
*// ImageView mImageView=new ImageView();*  
**if** (resultCode == ***RESULT\_OK***) {  
Uri selectedImageUri = data.getData();  
**if** (requestCode == ***SELECT\_FILE1***) {  
**selectedPath1** = FilePath.getPath(this, selectedImageUri);  
  
System.***out***.println(**"selectedPath1 "**+ **selectedPath1**);  
**if** (**selectedPath1** != **null**) {  
*//String[] filePathColumn = {MediaStore.Images.Media.DATA};*  
String checktpe = **selectedPath1**.substring(**selectedPath1**.lastIndexOf(**"."**) + 1);  
System.***out***.println(**"h "**+ checktpe);  
**if** (**attchtypecode1**.equals(**"2"**)) {  
**if** ((checktpe.equals(**"jpeg"**)) || (checktpe.equals(**"jpg"**))) {  
**filePath1**.setText(**selectedPath1**.substring(**selectedPath1**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath1**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only jpeg/jpg Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only jpeg/jpg Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode1**.equals(**"1"**)) {  
**if** ((checktpe.equals(**"pdf"**))) {  
**filePath1**.setText(**selectedPath1**.substring(**selectedPath1**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath1**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only pdf files"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only pdf files" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode1**.equals(**"3"**)) {  
**if** ((checktpe.equals(**"mp4"**))) {  
**filePath1**.setText(**selectedPath1**.substring(**selectedPath1**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath1**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only mp4 Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only mp4 Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else** {  
**filePath1**.setText(**""**);  
}  
} **else** {  
  
**filePath1**.setText(**""**);  
}  
}  
**if** (requestCode == ***SELECT\_FILE2***) {  
**selectedPath2** = FilePath.getPath(this, selectedImageUri); **if** (**selectedPath2** != **null**) {  
String checktpe = **selectedPath2**.substring(**selectedPath2**.lastIndexOf(**"."**) + 1);  
**if** (**attchtypecode2**.equals(**"2"**)) {  
**if** ((checktpe.equals(**"jpeg"**)) || (checktpe.equals(**"jpg"**))) {  
**filePath2**.setText(**selectedPath2**.substring(**selectedPath2**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath2**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only jpeg/jpg Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only jpeg/jpg Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode2**.equals(**"1"**)) {  
**if** ((checktpe.equals(**"pdf"**))) {  
**filePath2**.setText(**selectedPath2**.substring(**selectedPath2**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath2**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only pdf files"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only pdf files" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode2**.equals(**"3"**)) {  
**if** ((checktpe.equals(**"mp4"**))) {  
**filePath2**.setText(**selectedPath2**.substring(**selectedPath2**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath2**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only mp4 Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only mp4 Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else** {  
**filePath2**.setText(**""**);  
}  
} **else** {  
**filePath2**.setText(**""**);  
}  
}  
**if** (requestCode == ***SELECT\_FILE3***) {  
**selectedPath3** = FilePath.getPath(this, selectedImageUri); **if** (**selectedPath3** != **null**) {  
String checktpe = **selectedPath3**.substring(**selectedPath3**.lastIndexOf(**"."**) + 1);  
**if** (**attchtypecode3**.equals(**"2"**)) {  
**if** ((checktpe.equals(**"jpeg"**)) || (checktpe.equals(**"jpg"**))) {  
**filePath3**.setText(**selectedPath3**.substring(**selectedPath3**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath3**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only jpeg/jpg Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only jpeg/jpg Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode3**.equals(**"1"**)) {  
**if** ((checktpe.equals(**"pdf"**))) {  
**filePath3**.setText(**selectedPath3**.substring(**selectedPath3**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath3**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only pdf files"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only pdf files" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode3**.equals(**"3"**)) {  
**if** ((checktpe.equals(**"mp4"**))) {  
**filePath3**.setText(**selectedPath3**.substring(**selectedPath3**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath3**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only mp4 Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only mp4 Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else** {  
**filePath3**.setText(**""**);  
}  
} **else** {  
**filePath3**.setText(**""**);  
}  
}  
**if** (requestCode == ***SELECT\_FILE4***) {  
**selectedPath4** = FilePath.getPath(this, selectedImageUri); **if** (**selectedPath4** != **null**) {  
String checktpe = **selectedPath4**.substring(**selectedPath4**.lastIndexOf(**"."**) + 1);  
**if** (**attchtypecode4**.equals(**"2"**)) {  
**if** ((checktpe.equals(**"jpeg"**)) || (checktpe.equals(**"jpg"**))) {  
**filePath4**.setText(**selectedPath4**.substring(**selectedPath4**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath4**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only jpeg/jpg Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only jpeg/jpg Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode4**.equals(**"1"**)) {  
**if** ((checktpe.equals(**"pdf"**))) {  
**filePath4**.setText(**selectedPath4**.substring(**selectedPath4**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath4**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only pdf files"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only pdf files" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode4**.equals(**"3"**)) {  
**if** ((checktpe.equals(**"mp4"**))) {  
**filePath4**.setText(**selectedPath4**.substring(**selectedPath4**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath4**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only mp4 Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only mp4 Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else** {  
**filePath4**.setText(**""**);  
}  
} **else** {  
**filePath4**.setText(**""**);  
}  
  
}  
**if** (requestCode == ***SELECT\_FILE5***) {  
**selectedPath5** = FilePath.getPath(this, selectedImageUri); **if** (**selectedPath5** != **null**) {  
String checktpe = **selectedPath5**.substring(**selectedPath5**.lastIndexOf(**"."**) + 1);  
**if** (**attchtypecode5**.equals(**"2"**)) {  
**if** ((checktpe.equals(**"jpeg"**)) || (checktpe.equals(**"jpg"**))) {  
**filePath5**.setText(**selectedPath5**.substring(**selectedPath5**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath5**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only jpeg/jpg Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only jpeg/jpg Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode5**.equals(**"1"**)) {  
**if** ((checktpe.equals(**"pdf"**))) {  
**filePath5**.setText(**selectedPath5**.substring(**selectedPath5**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath5**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only pdf files"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*//Toast.makeText(getApplicationContext(), "Select only pdf files" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else if** (**attchtypecode5**.equals(**"3"**)) {  
**if** ((checktpe.equals(**"mp4"**))) {  
**filePath5**.setText(**selectedPath5**.substring(**selectedPath5**.lastIndexOf(**"/"**) + 1));  
} **else** {  
**filePath5**.setText(**""**);  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Select only mp4 Type"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
*// Toast.makeText(getApplicationContext(), "Select only mp4 Type" ,Toast.LENGTH\_SHORT).show();*  
}  
} **else** {  
**filePath5**.setText(**""**);  
}  
} **else** {  
**filePath5**.setText(**""**);  
}  
}  
}  
}  
  
**public void** FinalFileUpload(View view) {  
  
  
hidekeyboard();  
**int** rowNumCount = **click** + 1;  
  
**aplno** = **applno**.getText().toString().trim();  
**mobno** = **mblno**.getText().toString().trim();  
*/\* slno1s=slno1.getText().toString().trim();*  
*slno2s=slno2.getText().toString().trim();*  
*slno3s=slno3.getText().toString().trim();*  
*slno4s=slno4.getText().toString().trim();*  
*slno5s=slno5.getText().toString().trim();\*/*  
**slno1s** = **"1"**;  
**slno2s** = **"2"**;  
**slno3s** = **"3"**;  
**slno4s** = **"4"**;  
**slno5s** = **"5"**;  
**docname1s** = **docname1**.getText().toString().trim();  
**attachremark1s** = **attachremark1**.getText().toString().trim();  
**docname2s** = **docname2**.getText().toString().trim();  
**attachremark2s** = **attachremark2**.getText().toString().trim();  
**docname3s** = **docname3**.getText().toString().trim();  
**attachremark3s** = **attachremark3**.getText().toString().trim();  
**docname4s** = **docname4**.getText().toString().trim();  
**attachremark4s** = **attachremark4**.getText().toString().trim();  
**docname5s** = **docname5**.getText().toString().trim();  
**attachremark5s** = **attachremark5**.getText().toString().trim();  
**selectedPath1s** = **selectedPath1**;  
**selectedPath2s** = **selectedPath2**;  
**selectedPath3s** = **selectedPath3**;  
**selectedPath4s** = **selectedPath4**;  
**selectedPath5s** = **selectedPath5**;  
**attchtypecode1s** = **attchtypecode1**;  
**attchtypecode2s** = **attchtypecode2**;  
**attchtypecode3s** = **attchtypecode3**;  
**attchtypecode4s** = **attchtypecode4**;  
**attchtypecode5s** = **attchtypecode5**;  
Pattern specialchar = Pattern.*compile*(**"[#;!£$%^&\*}{@~`?<>.+\_='|:\"\\[\\]\\\\]+"**);  
Pattern alphabets = Pattern.*compile*(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**);  
Pattern digits = Pattern.*compile*(**".\*\\p{Digit}.\*"**);  
**boolean** cancel = **false**;  
View focusView = **null**;  
  
  
**if** (!(TextUtils.*isEmpty*(**attchtypecode1s**))) {  
**if** (**slno1s**.length() >2) {  
  
  
cancel = **true**;  
}  
*/\* if ((TextUtils.isEmpty(attchtypecode1s))) {*  
*cancel = true;*  
  
*}\*/*  
**if** (!(TextUtils.*isEmpty*(**attchtypecode1s**))) {  
**if** (!(Pattern.*matches*(**".\*\\p{Digit}.\*"**, **attchtypecode1s**))) {  
cancel = **true**;  
} **else if** (**attchtypecode1s**.length() >10) {  
cancel = **true**;  
}  
}  
**if** ((TextUtils.*isEmpty*(**docname1s**))) {  
**TextInputLayoutdocname1**.setError(getString(R.string.***error\_field\_required***));  
focusView = **docname1**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**docname1s**))) {  
**if** (Pattern.*matches*(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, **docname1s**)) {  
**if** (!(alphabets.equals(**docname1s**))) {  
focusView = **docname1**;  
**TextInputLayoutdocname1**.setError(getString(R.string.***error\_field\_required***));  
cancel = **true**;  
} **else if** ((digits.equals(**docname1s**))) {  
focusView = **docname1**;  
cancel = **true**;  
**TextInputLayoutdocname1**.setError(getString(R.string.***error\_field\_required***));  
}  
}  
}  
  
**if** ((TextUtils.*isEmpty*(**selectedPath1**))) {  
*/\*Toast.makeText(getApplicationContext(),*  
*"Select file to upload", Toast.LENGTH\_SHORT).show();\*/*  
cancel = **true**;  
  
}  
**if** ((TextUtils.*isEmpty*(**attachremark1s**))) {  
**TextInputLayoutattachremark1**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark1**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**attachremark1s**))) {  
  
**if** (Pattern.*matches*(**"[#;!Â£$%^&\*}{@~`?<>+\_(=)'|:\"\\[\\]\\\\]+"**, **attachremark1s**)) {  
**TextInputLayoutattachremark1**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark1**;  
cancel = **true**;  
} **else if** (Pattern.*matches*(**"[0-9]+"**, **attachremark1s**)) {  
**TextInputLayoutattachremark1**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark1**;  
cancel = **true**;  
  
} **else if** (**attachremark1**.length() >150) {  
**TextInputLayoutattachremark1**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark1**;  
cancel = **true**;  
}  
}  
  
}  
  
**if** (!(TextUtils.*isEmpty*(**attchtypecode2s**))) {  
**if** (**slno2s**.length() >2) {  
  
  
cancel = **true**;  
}  
*/\* if ((TextUtils.isEmpty(attchtypecode2s))) {*  
*cancel = true;*  
  
*}\*/*  
**if** (!(TextUtils.*isEmpty*(**attchtypecode2s**))) {  
**if** (!(Pattern.*matches*(**".\*\\p{Digit}.\*"**, **attchtypecode2s**))) {  
cancel = **true**;  
} **else if** (**attchtypecode2s**.length() >10) {  
cancel = **true**;  
}  
}  
**if** ((TextUtils.*isEmpty*(**selectedPath2**))) {  
*/\*Toast.makeText(getApplicationContext(),*  
*"Select file to upload", Toast.LENGTH\_SHORT).show();\*/*  
cancel = **true**;  
  
}  
  
**if** ((TextUtils.*isEmpty*(**docname2s**))) {  
**TextInputLayoutdocname2**.setError(getString(R.string.***error\_field\_required***));  
focusView = **docname2**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**docname2s**))) {  
**if** (Pattern.*matches*(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, **docname2s**)) {  
**if** (!(alphabets.equals(**docname2s**))) {  
**TextInputLayoutdocname2**.setError(getString(R.string.***error\_field\_required***));  
focusView = **docname2**;  
cancel = **true**;  
} **else if** ((digits.equals(**docname2s**))) {  
focusView = **docname2**;  
**TextInputLayoutdocname2**.setError(getString(R.string.***error\_field\_required***));  
cancel = **true**;  
  
}  
}  
}  
**if** ((TextUtils.*isEmpty*(**attachremark2s**))) {  
**TextInputLayoutattachremark2**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark2**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**attachremark2s**))) {  
  
**if** (Pattern.*matches*(**"[#;!Â£$%^&\*}{@~`?<>+\_(=)'|:\"\\[\\]\\\\]+"**, **attachremark2s**)) {  
**TextInputLayoutattachremark2**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark2**;  
cancel = **true**;  
} **else if** (Pattern.*matches*(**"[0-9]+"**, **attachremark2s**)) {  
**TextInputLayoutattachremark2**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark2**;  
cancel = **true**;  
  
} **else if** (**attachremark2**.length() >150) {  
**TextInputLayoutattachremark2**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark2**;  
cancel = **true**;  
}  
}  
  
}  
  
**if** (!(TextUtils.*isEmpty*(**attchtypecode3s**))) {  
**if** (**slno3s**.length() >2) {  
  
cancel = **true**;  
}  
*/\* if ((TextUtils.isEmpty(attchtypecode3s))) {*  
*cancel = true;*  
  
*}\*/*  
**if** (!(TextUtils.*isEmpty*(**attchtypecode3s**))) {  
**if** (!(Pattern.*matches*(**".\*\\p{Digit}.\*"**, **attchtypecode3s**))) {  
cancel = **true**;  
} **else if** (**attchtypecode3s**.length() >10) {  
cancel = **true**;  
}  
}  
**if** ((TextUtils.*isEmpty*(**docname3s**))) {  
**TextInputLayoutdocname3**.setError(getString(R.string.***error\_field\_required***));  
focusView = **docname3**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**docname3s**))) {  
**if** (Pattern.*matches*(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, **docname3s**)) {  
**if** (!(alphabets.equals(**docname3s**))) {  
focusView = **docname3**;  
cancel = **true**;  
**TextInputLayoutdocname3**.setError(getString(R.string.***error\_field\_required***));  
} **else if** ((digits.equals(**docname3s**))) {  
focusView = **docname3**;  
cancel = **true**;  
**TextInputLayoutdocname3**.setError(getString(R.string.***error\_field\_required***));  
  
}  
}  
}  
  
**if** ((TextUtils.*isEmpty*(**selectedPath3**))) {  
*/\*Toast.makeText(getApplicationContext(),*  
*"Select file to upload", Toast.LENGTH\_SHORT).show();\*/*  
cancel = **true**;  
  
}  
**if** ((TextUtils.*isEmpty*(**attachremark3s**))) {  
**TextInputLayoutattachremark3**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark3**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**attachremark3s**))) {  
  
**if** (Pattern.*matches*(**"[#;!Â£$%^&\*}{@~`?<>+\_(=)'|:\"\\[\\]\\\\]+"**, **attachremark3s**)) {  
**TextInputLayoutattachremark3**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark3**;  
cancel = **true**;  
} **else if** (Pattern.*matches*(**"[0-9]+"**, **attachremark3s**)) {  
**TextInputLayoutattachremark3**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark3**;  
cancel = **true**;  
  
} **else if** (**attachremark3**.length() >150) {  
**TextInputLayoutattachremark3**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark3**;  
cancel = **true**;  
}  
}  
  
}  
**if** (!(TextUtils.*isEmpty*(**attchtypecode4s**))) {  
**if** (**slno4s**.length() >2) {  
  
cancel = **true**;  
}  
*/\* if ((TextUtils.isEmpty(attchtypecode4s))) {*  
*cancel = true;*  
  
*}\*/*  
**if** (!(TextUtils.*isEmpty*(**attchtypecode4s**))) {  
**if** (!(Pattern.*matches*(**".\*\\p{Digit}.\*"**, **attchtypecode4s**))) {  
cancel = **true**;  
} **else if** (**attchtypecode4s**.length() >10) {  
cancel = **true**;  
}  
}  
  
**if** ((TextUtils.*isEmpty*(**selectedPath4**))) {  
*/\* Toast.makeText(getApplicationContext(),*  
*"Select file to upload", Toast.LENGTH\_SHORT).show();\*/*  
cancel = **true**;  
  
}  
**if** ((TextUtils.*isEmpty*(**docname4s**))) {  
**TextInputLayoutdocname4**.setError(getString(R.string.***error\_field\_required***));  
focusView = **docname4**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**docname4s**))) {  
**if** (Pattern.*matches*(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, **docname4s**)) {  
**if** (!(alphabets.equals(**docname4s**))) {  
focusView = **docname4**;  
cancel = **true**;  
**TextInputLayoutdocname4**.setError(getString(R.string.***error\_field\_required***));  
} **else if** ((digits.equals(**docname4s**))) {  
focusView = **docname4**;  
cancel = **true**;  
**TextInputLayoutdocname4**.setError(getString(R.string.***error\_field\_required***));  
  
}  
}  
}  
**if** ((TextUtils.*isEmpty*(**attachremark4s**))) {  
**TextInputLayoutattachremark4**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark4**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**attachremark4s**))) {  
  
**if** (Pattern.*matches*(**"[#;!Â£$%^&\*}{@~`?<>+\_(=)'|:\"\\[\\]\\\\]+"**, **attachremark4s**)) {  
**TextInputLayoutattachremark4**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark4**;  
cancel = **true**;  
} **else if** (Pattern.*matches*(**"[0-9]+"**, **attachremark4s**)) {  
**TextInputLayoutattachremark4**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark4**;  
cancel = **true**;  
  
} **else if** (**attachremark4**.length() >150) {  
**TextInputLayoutattachremark4**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark4**;  
cancel = **true**;  
}  
}  
  
}  
  
**if** (!(TextUtils.*isEmpty*(**attchtypecode5s**))) {  
  
**if** (**slno5s**.length() >2) {  
  
cancel = **true**;  
}  
*/\* if ((TextUtils.isEmpty(attchtypecode5s))) {*  
*cancel = true;*  
  
*}\*/*  
**if** (!(TextUtils.*isEmpty*(**attchtypecode5s**))) {  
**if** (!(Pattern.*matches*(**".\*\\p{Digit}.\*"**, **attchtypecode5s**))) {  
cancel = **true**;  
} **else if** (**attchtypecode5s**.length() >10) {  
cancel = **true**;  
}  
}  
  
**if** ((TextUtils.*isEmpty*(**selectedPath5**))) {  
*/\* Toast.makeText(getApplicationContext(),*  
*"Select file to upload", Toast.LENGTH\_SHORT).show();\*/*  
cancel = **true**;  
  
}  
**if** ((TextUtils.*isEmpty*(**docname5s**))) {  
**TextInputLayoutdocname5**.setError(getString(R.string.***error\_field\_required***));  
focusView = **docname5**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**docname5s**))) {  
**if** (Pattern.*matches*(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, **docname5s**)) {  
**if** (!(alphabets.equals(**docname5s**))) {  
focusView = **docname5**;  
cancel = **true**;  
**TextInputLayoutdocname5**.setError(getString(R.string.***error\_field\_required***));  
} **else if** ((digits.equals(**docname5s**))) {  
focusView = **docname5**;  
cancel = **true**;  
**TextInputLayoutdocname5**.setError(getString(R.string.***error\_field\_required***));  
}  
}  
}  
**if** ((TextUtils.*isEmpty*(**attachremark5s**))) {  
**TextInputLayoutattachremark5**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark5**;  
cancel = **true**;  
  
}  
**if** (!(TextUtils.*isEmpty*(**attachremark5s**))) {  
  
**if** (Pattern.*matches*(**"[#;!Â£$%^&\*}{@~`?<>+\_(=)'|:\"\\[\\]\\\\]+"**, **attachremark5s**)) {  
**TextInputLayoutattachremark5**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark5**;  
cancel = **true**;  
} **else if** (Pattern.*matches*(**"[0-9]+"**, **attachremark5s**)) {  
**TextInputLayoutattachremark5**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark5**;  
cancel = **true**;  
  
} **else if** (**attachremark5**.length() >150) {  
**TextInputLayoutattachremark5**.setError(getString(R.string.***error\_field\_required***));  
focusView = **attachremark5**;  
cancel = **true**;  
}  
}  
  
}  
  
  
**if** (cancel == **false**) {  
**if** (!(**selectedPath1s**.equals(**""**) || **selectedPath1s**.equals(**null**))) {  
  
uploadFile(**selectedPath1s**, **docname1s**, **attachremark1s**, **slno1s**, **aplno**, **mobno**, **attchtypecode1s**);  
}  
  
**if** (!(**selectedPath2s**.equals(**""**) || **selectedPath2s**.equals(**null**))) {  
  
uploadFile(**selectedPath2s**, **docname2s**, **attachremark2s**, **slno2s**, **aplno**, **mobno**, **attchtypecode2s**);  
}  
  
**if** (!(**selectedPath3s**.equals(**""**) || **selectedPath3s**.equals(**null**))) {  
  
uploadFile(**selectedPath3s**, **docname3s**, **attachremark3s**, **slno3s**, **aplno**, **mobno**, **attchtypecode3s**);  
}  
  
**if** (!(**selectedPath4s**.equals(**""**) || **selectedPath4s**.equals(**null**))) {  
  
uploadFile(**selectedPath4s**, **docname4s**, **attachremark4s**, **slno4s**, **aplno**, **mobno**, **attchtypecode4s**);  
}  
  
**if** (!(**selectedPath5s**.equals(**""**) || **selectedPath5s**.equals(**null**))) {  
  
uploadFile(**selectedPath5s**, **docname5s**, **attachremark5s**, **slno5s**, **aplno**, **mobno**, **attchtypecode5s**);  
}  
} **else** {  
android.app.AlertDialog.Builder alert = **new** android.app.AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Enter All Details"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
}  
}  
  
**private void** hidekeyboard() {  
  
InputMethodManager imm = (InputMethodManager) getSystemService(Context.***INPUT\_METHOD\_SERVICE***);  
imm.hideSoftInputFromWindow(getCurrentFocus().getWindowToken(),  
InputMethodManager.***RESULT\_UNCHANGED\_SHOWN***);  
}  
  
  
**public void** uploadFile(String Filename, String docname, String attachremark, String slno, String aplno, String mobno, String attchtypecode) {  
**boolean** flag = **false**;  
InputStream inputStream;  
**try** {  
**docnames** = docname;  
**attachrem** = attachremark;  
**slnos** = slno;  
**attachtypes** = attchtypecode;  
inputStream = **new** FileInputStream(Filename);  
  
*//You can get an inputStream using any IO API*  
**byte**[] bytes;  
**byte**[] buffer = **new byte**[8192];  
**int** bytesRead;  
ByteArrayOutputStream output = **new** ByteArrayOutputStream();  
**try** {  
**while** ((bytesRead = inputStream.read(buffer)) != -1) {  
output.write(buffer, 0, bytesRead);  
}  
} **catch** (IOException e) {  
e.printStackTrace();  
}  
bytes = output.toByteArray();  
**encodedString** = Base64Utils.*base64Encode*(bytes);  
*//Mylist.add(encodedString);*  
**int** filesize = **encodedString**.length();  
**if** (attchtypecode.equals(**"1"**)) {  
**int** maxsize = 102400;  
**if** (filesize >= maxsize || filesize <= 0) {  
flag = **true**;  
}  
}  
**if** (attchtypecode.equals(**"2"**)) {  
**int** maxsize = 204800;  
**if** (filesize >= maxsize || filesize <= 0) {  
flag = **true**;  
}  
}  
**if** (attchtypecode.equals(**"3"**)) {  
**int** maxsize = 5242880;  
**if** (filesize >= maxsize || filesize <= 0) {  
flag = **true**;  
}  
}  
**if** (flag == **false**) {  
**int** index = Filename.lastIndexOf(**"/"**);  
**Typename** = Filename.substring(index + 1);  
System.***out***.println(**"Typename "**+ **Typename**);  
  
  
**if** (Utils.*isOnline*(**this**)) {  
  
**pDialog** = **new** ProgressDialog(FileUpload.**this**);  
pDialog.setMessage(**"Loading..."**);  
pDialog.setCancelable(**false**);  
pDialog.show();  
  
  
upload();  
  
  
} **else** {  
  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"No Internet Connection!!!"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
  
}  
  
} **else** {  
  
**if** (attchtypecode.equals(**"1"**)) {  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Upload files upto 100KB"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
  
}  
**if** (attchtypecode.equals(**"2"**)) {  
  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Upload image upto 200KB"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
  
}  
**if** (attchtypecode.equals(**"3"**)) {  
  
AlertDialog.Builder alert = **new** AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Upload video upto 5MB"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
  
}  
  
}  
  
} **catch** (FileNotFoundException e1) {  
e1.printStackTrace();  
}  
  
}  
  
**private void** upload() {  
url = **"http://103.251.43.122/pgmobapp/service/mobileapp/fileupload"**;  
**if** (url.contains(**"https"**)) {  
requestQueue = Volley.newRequestQueue(**this**, hurlstack);  
} **else** {  
requestQueue = Volley.newRequestQueue(**this**);  
}  
  
**final** StringRequest jsonObjectRequest = **new** StringRequest(Request.Method.POST, url, **new** Response.Listener<String>() {  
  
**public void** onResponse(String response) {  
**try** {  
  
pDialog.dismiss();  
String err = (response.trim() == **null**) ? **"okk"**: response.trim();  
util.alertButton(err, **true**);  
} **catch** (Exception e) {  
pDialog.dismiss();  
e.printStackTrace();  
}  
}  
}, **new** Response.ErrorListener() {  
@Override  
**public void** onErrorResponse(VolleyError e) {  
e.printStackTrace();  
pDialog.dismiss();  
util.alertButton(**"Connection Error"**, **true**);  
}  
}) {  
@Override  
  
**public** Map<String, String> getParams() {  
**try** {  
params = **new** HashMap<>();  
params.put(**"aplno"**, URLEncoder  
.encode(aplno, **"UTF-8"**));  
params.put(**"mobno"**, URLEncoder  
.encode(mobno, **"UTF-8"**));  
params.put(**"attachremark"**, URLEncoder  
.encode(attachrem, **"UTF-8"**));  
  
params.put(**"attchtypecode"**, URLEncoder  
.encode(attachtypes, **"UTF-8"**));  
params.put(**"docname"**, URLEncoder  
.encode(docnames, **"UTF-8"**));  
params.put(**"Typename"**, URLEncoder  
.encode(Typename, **"UTF-8"**));  
params.put(**"slno"**, URLEncoder  
.encode(slnos, **"UTF-8"**));  
params.put(**"encodedString"**, encodedString);  
  
  
} **catch** (Exception e) {  
  
}  
**return** params;  
}  
};  
  
jsonObjectRequest.setRetryPolicy(**new** DefaultRetryPolicy(10000,  
DefaultRetryPolicy.DEFAULT\_MAX\_RETRIES,  
DefaultRetryPolicy.DEFAULT\_BACKOFF\_MULT));  
requestQueue.getCache().clear();  
requestQueue.add(jsonObjectRequest);  
}  
  
  
String MobilePattern = **"[0-9]{10}"**;  
  
**public void** onVerify(View view) {  
  
button\_verify.setOnTouchListener(**new** View.OnTouchListener() {  
  
@Override  
**public boolean** onTouch(View v, MotionEvent event) {  
InputMethodManager imm = (InputMethodManager) getApplicationContext().getSystemService(Context.INPUT\_METHOD\_SERVICE);  
imm.hideSoftInputFromWindow(button\_verify.getWindowToken(), 0);  
**return false**;  
}  
});  
  
applNo = applno.getText().toString().trim();  
mobileNumber = mblno.getText().toString().trim();  
View focusView = **null**;  
flag = **true**;  
linear.setVisibility(View.INVISIBLE);  
  
  
**if** (TextUtils.isEmpty(applNo)) {  
flag = **false**;  
  
TextInputLayoutfbapplno.setError(getString(R.string.ErrorapplcationNo));  
  
} **else** {  
**if** (Pattern.matches(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, applNo)) {  
focusView = applno;  
flag = **false**;  
  
TextInputLayoutfbapplno.setError(getString(R.string.ErrorapplcationNo));  
  
} **else if** (!(Pattern.matches(**".\*\\p{Digit}.\*"**, applNo))) {  
focusView = applno;  
flag = **false**;  
  
TextInputLayoutfbapplno.setError(getString(R.string.ErrorapplcationNo));  
  
} **else if** (applNo.length() >6) {  
focusView = applno;  
flag = **false**;  
  
TextInputLayoutfbapplno.setError(getString(R.string.ErrorapplcationNo));  
  
} **else** {  
TextInputLayoutfbapplno.setError(**null**);  
*// flag = true;*  
}  
}  
  
  
**if** (TextUtils.isEmpty(mobileNumber)) {  
focusView = mblno;  
flag = **false**;  
  
TextInputLayoutfbmblno.setError(getString(R.string.ErrorMobileNumber));  
  
} **else** {  
**if** (Pattern.matches(**".\*[-#;!£$%^&\*}{@~`?\n<>/+\_(=),'|.:\"\\[\\]\\\\].\*"**, mobileNumber)) {  
focusView = mblno;  
flag = **false**;  
  
TextInputLayoutfbmblno.setError(getString(R.string.ErrorMobileNumber));  
  
} **else if** (!(Pattern.matches(**".\*\\p{Digit}.\*"**, mobileNumber))) {  
focusView = mblno;  
flag = **false**;  
  
TextInputLayoutfbmblno.setError(getString(R.string.ErrorMobileNumber));  
  
} **else if** (mobileNumber.length() >11) {  
focusView = mblno;  
flag = **false**;  
  
TextInputLayoutfbmblno.setError(getString(R.string.ErrorMobileNumber));  
  
} **else** {  
TextInputLayoutfbmblno.setError(**null**);  
*//flag = true;*  
}  
}  
  
  
**if** (flag) {  
  
msg = **""**;  
  
**if** (Utils.isOnline(**this**)) {  
  
  
pDialog = **new** ProgressDialog(FileUpload.**this**);  
pDialog.setMessage(**"Loading..."**);  
pDialog.setCancelable(**false**);  
pDialog.show();  
LoadService();  
} **else** {  
  
Toast.makeText(getApplicationContext(),  
**"No Internet Connection!!!"**, Toast.LENGTH\_SHORT).show();  
}  
} **else** {  
  
Toast.makeText(getApplicationContext(), msg, Toast.LENGTH\_SHORT).show();  
msg = **""**;  
  
}  
}  
  
  
**private void** LoadService() {  
url=**"http://103.251.43.122/pgmobapp/service/mobileapp/getstatus?mobileNo="**+  
mobileNumber +  
**"&applNo="**+  
applNo;  
  
**if** (url.contains(**"https"**)) {  
requestQueue = Volley.newRequestQueue(**this**, hurlstack);  
} **else** {  
requestQueue = Volley.newRequestQueue(**this**);  
}  
  
**final** StringRequest jsonObjectRequest = **new** StringRequest(Request.Method.GET, url, **new** Response.Listener<String>() {  
  
**public void** onResponse(String response) {  
**try** {  
pDialog.dismiss();  
JSONObject json = **new** JSONObject(response);  
  
**if** (json.has(**"Status Response"**)) {  
JSONObject object1 = json.getJSONObject(**"Status Response"**);  
JSONArray object11 = object1.getJSONArray(**"Status"**);  
  
**if** (!(object11.equals(**null**) || object11.equals(**""**))) {  
  
  
String stat = (String) object11.getJSONObject(2).get(**"Status"**);  
  
**if** (stat.equalsIgnoreCase(**"Processing"**)) {  
  
android.app.AlertDialog.Builder alert = **new** android.app.AlertDialog.Builder(FileUpload.**this**);  
alert.setTitle(**"Message"**);  
alert.setMessage(**"Now You Can Upload Your Documents"**);  
alert.setPositiveButton(**"OK"**, **null**);  
alert.show();  
  
linear.setVisibility(View.VISIBLE);  
} **else** {  
util.alertButton(**"You application is under Processing.You can't upload Documents"**, **true**);  
}  
  
}  
} **else** {  
  
JSONObject st = json.getJSONObject(**"Result"**);  
String re = (String) st.get(**"error"**);  
util.alertButton(re, **true**);  
  
}  
} **catch** (JSONException e) {  
pDialog.dismiss();  
  
util.alertSingleButton(**"Connection failed..Retry"**, **null**);  
  
}  
}  
  
}, **new** Response.ErrorListener() {  
  
**public void** onErrorResponse(VolleyError error) {  
pDialog.dismiss();  
util.alertButton(**"Connection Error"**, **true**);  
  
}  
});  
  
  
jsonObjectRequest.setRetryPolicy(**new** DefaultRetryPolicy(10000,  
DefaultRetryPolicy.DEFAULT\_MAX\_RETRIES,  
DefaultRetryPolicy.DEFAULT\_BACKOFF\_MULT));  
requestQueue.getCache().clear();  
requestQueue.add(jsonObjectRequest);  
  
}  
  
  
**public void** addFileUpload(View view) {  
click++;  
**if** (click == 1) {  
tableinsert2.setVisibility(view.VISIBLE);  
}  
**if** (click == 2) {  
tableinsert3.setVisibility(view.VISIBLE);  
}  
**if** (click == 3) {  
tableinsert4.setVisibility(view.VISIBLE);  
}  
**if** (click == 4) {  
tableinsert5.setVisibility(view.VISIBLE);  
  
}  
**if** (click >= 5) {  
click = 0;  
}  
  
}  
  
**public void** removeFileUpload(View view) {  
  
**if** (click == 4) {  
tableinsert5.setVisibility(view.GONE);  
click--;  
slno5.setText(**"5"**);  
docname5.setText(**""**);  
attachremark5.setText(**""**);  
filePath5.setText(**"No File Selected"**);  
attchtype5.setSelection(0);  
  
} **else if** (click == 3) {  
tableinsert4.setVisibility(view.GONE);  
click--;  
slno4.setText(**"4"**);  
docname4.setText(**""**);  
attachremark4.setText(**""**);  
filePath4.setText(**"No File Selected"**);  
attchtype4.setSelection(0);  
} **else if** (click == 2) {  
tableinsert3.setVisibility(view.GONE);  
click--;  
slno3.setText(**"3"**);  
docname3.setText(**""**);  
attachremark3.setText(**""**);  
filePath3.setText(**"No File Selected"**);  
attchtype3.setSelection(0);  
  
} **else if** (click == 1) {  
tableinsert2.setVisibility(view.GONE);  
click--;  
slno2.setText(**"2"**);  
docname2.setText(**""**);  
attachremark2.setText(**""**);  
filePath2.setText(**"No File Selected"**);  
attchtype2.setSelection(0);  
}  
  
  
}  
  
**public void** clearFileUpload(View view) {  
slno5.setText(**"5"**);  
docname5.setText(**""**);  
attachremark5.setText(**""**);  
filePath5.setText(**"No File Selected"**);  
attchtype5.setSelection(0);  
  
slno4.setText(**"4"**);  
docname4.setText(**""**);  
attachremark4.setText(**""**);  
filePath4.setText(**"No File Selected"**);  
attchtype4.setSelection(0);  
  
slno3.setText(**"3"**);  
docname3.setText(**""**);  
attachremark3.setText(**""**);  
filePath3.setText(**"No File Selected"**);  
attchtype3.setSelection(0);  
  
slno2.setText(**"2"**);  
docname2.setText(**""**);  
attachremark2.setText(**""**);  
filePath2.setText(**"No File Selected"**);  
attchtype2.setSelection(0);  
  
slno1.setText(**"1"**);  
docname1.setText(**""**);  
attachremark1.setText(**""**);  
filePath1.setText(**"No File Selected"**);  
attchtype1.setSelection(0);  
tableinsert5.setVisibility(view.GONE);  
tableinsert4.setVisibility(view.GONE);  
tableinsert3.setVisibility(view.GONE);  
tableinsert2.setVisibility(view.GONE);  
  
  
}  
  
  
**private void** loadAttachment1() {  
ArrayAdapter<Attachment\_enum> dataAdapter = **new** ArrayAdapter<Attachment\_enum>(**this**,  
android.R.layout.simple\_spinner\_item, Attachment\_enum.values());  
dataAdapter  
.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);  
  
attchtype1.setAdapter(dataAdapter);  
  
**final int** iCurrentSelection = -1;  
attchtype1.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
**public void** onItemSelected(AdapterView<?> parentView, View selectedItemView, **int** position, **long** id) {  
**if** (iCurrentSelection != position) {  
  
*// System.out.println( ( (Attachment\_enum) attchtype1.getSelectedItem() ).value());*  
attchtypecode1 = ((Attachment\_enum) attchtype1.getSelectedItem()).value();  
System.out.println(iCurrentSelection + **" position"**+ attchtypecode1);  
}  
}  
  
@Override  
**public void** onNothingSelected(AdapterView<?> arg0) {  
*// TODO Auto-generated method stub*  
}  
});  
  
}  
  
**private void** loadAttachment2() {  
ArrayAdapter<Attachment\_enum> dataAdapter = **new** ArrayAdapter<Attachment\_enum>(**this**,  
android.R.layout.simple\_spinner\_item, Attachment\_enum.values());  
dataAdapter  
.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);  
  
attchtype2.setAdapter(dataAdapter);  
  
**final int** iCurrentSelection = -1;  
attchtype2.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
**public void** onItemSelected(AdapterView<?> parentView, View selectedItemView, **int** position, **long** id) {  
**if** (iCurrentSelection != position) {  
  
*// System.out.println( ( (Attachment\_enum) attchtype2.getSelectedItem() ).value());*  
attchtypecode2 = ((Attachment\_enum) attchtype2.getSelectedItem()).value();  
System.out.println(iCurrentSelection + **" position"**+ attchtypecode2);  
}  
}  
  
@Override  
**public void** onNothingSelected(AdapterView<?> arg0) {  
*// TODO Auto-generated method stub*  
}  
});  
  
}  
  
**private void** loadAttachment3() {  
ArrayAdapter<Attachment\_enum> dataAdapter = **new** ArrayAdapter<Attachment\_enum>(**this**,  
android.R.layout.simple\_spinner\_item, Attachment\_enum.values());  
dataAdapter  
.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);  
  
attchtype3.setAdapter(dataAdapter);  
  
**final int** iCurrentSelection = -1;  
attchtype3.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
**public void** onItemSelected(AdapterView<?> parentView, View selectedItemView, **int** position, **long** id) {  
**if** (iCurrentSelection != position) {  
  
*//System.out.println( ( (Attachment\_enum) attchtype3.getSelectedItem() ).value());*  
attchtypecode3 = ((Attachment\_enum) attchtype3.getSelectedItem()).value();  
System.out.println(iCurrentSelection + **" position"**+ attchtypecode3);  
}  
}  
  
@Override  
**public void** onNothingSelected(AdapterView<?> arg0) {  
*// TODO Auto-generated method stub*  
}  
});  
  
}  
  
**private void** loadAttachment4() {  
ArrayAdapter<Attachment\_enum> dataAdapter = **new** ArrayAdapter<Attachment\_enum>(**this**,  
android.R.layout.simple\_spinner\_item, Attachment\_enum.values());  
dataAdapter  
.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);  
  
attchtype4.setAdapter(dataAdapter);  
  
**final int** iCurrentSelection = -1;  
attchtype4.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
**public void** onItemSelected(AdapterView<?> parentView, View selectedItemView, **int** position, **long** id) {  
**if** (iCurrentSelection != position) {  
  
*// System.out.println( ( (Attachment\_enum) attchtype4.getSelectedItem() ).value());*  
attchtypecode4 = ((Attachment\_enum) attchtype4.getSelectedItem()).value();  
System.out.println(iCurrentSelection + **" position"**+ attchtypecode4);  
}  
}  
  
@Override  
**public void** onNothingSelected(AdapterView<?> arg0) {  
*// TODO Auto-generated method stub*  
}  
});  
  
}  
  
**private void** loadAttachment5() {  
ArrayAdapter<Attachment\_enum> dataAdapter = **new** ArrayAdapter<Attachment\_enum>(**this**,  
android.R.layout.simple\_spinner\_item, Attachment\_enum.values());  
dataAdapter  
.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);  
  
attchtype5.setAdapter(dataAdapter);  
  
**final int** iCurrentSelection = -1;  
attchtype5.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
**public void** onItemSelected(AdapterView<?> parentView, View selectedItemView, **int** position, **long** id) {  
**if** (iCurrentSelection != position) {  
  
*// System.out.println( ( (Attachment\_enum) attchtype5.getSelectedItem() ).value());*  
attchtypecode5 = ((Attachment\_enum) attchtype5.getSelectedItem()).value();  
System.out.println(iCurrentSelection + **" position"**+ attchtypecode5);  
}  
}  
  
@Override  
**public void** onNothingSelected(AdapterView<?> arg0) {  
*// TODO Auto-generated method stub*  
}  
});  
  
}  
  
  
}

***SQLite***

Android provides several ways to store user and app data. SQLite is one way of storing user data. SQLite is a very light weight database which comes with Android OS. In this example we are creaing a Login form and registration form using sqlite. Create Registration Form



a. Add below code in "activity\_register.xml" file

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:orientation="vertical"android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**>  
  
<**include**  
**android:id="@+id/head"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_width="fill\_parent"**  
**layout="@layout/activity\_title"**  
  
/>  
  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/userlayout"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/head"**>  
  
<**EditText**  
**android:id="@+id/usertext"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:ems="10"**  
**android:hint="Name"**/>  
</**android.support.design.widget.TextInputLayout**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/passlayout"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/userlayout"**>  
  
<**EditText**  
**android:id="@+id/passtext"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:ems="10"**  
**android:hint="Password"**/>  
</**android.support.design.widget.TextInputLayout**>  
  
  
<**Button**  
**android:id="@+id/reg"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Register"**  
**android:textColor="#000000"**  
**android:background="@drawable/buttoncustom"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_below="@+id/passlayout"**/>  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
</**RelativeLayout**>

b. Add below code in "RegisterActivity.java" file

**package** cs.app.nic.com.seminarclass;  
  
**import** android.os.Bundle;  
**import** android.support.annotation.Nullable;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/12/2017.*  
*\*/*  
  
**public class** RegisterActivity **extends** AppCompatActivity{  
Button **btnreg**;  
EditText **edtuser**,**edtpass**;  
String **username**,**password**;  
LoginDataBaseAdapter **loginDataBaseAdapter**;  
@Override  
**protected void** onCreate(@Nullable Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activity\_register***);  
  
  
*// get Instance of Database Adapter*  
**loginDataBaseAdapter**=**new** LoginDataBaseAdapter(RegisterActivity.**this**);  
**loginDataBaseAdapter**=**loginDataBaseAdapter**.open();  
**edtuser**=(EditText)findViewById(R.id.***usertext***);  
**edtpass**=(EditText)findViewById(R.id.***passtext***);  
**btnreg**=(Button)findViewById(R.id.***reg***);  
View includedLayout = findViewById(R.id.***head***);  
  
TextView txttitle= (TextView)includedLayout.findViewById(R.id.***txttitile***);  
txttitle.setText(**"Registration"**);  
  
**btnreg**.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
  
**if**(**edtuser**.getText().toString().equals(**""**)){  
**edtuser**.requestFocus();  
**edtuser**.setError(**"Plaese enter username"**);  
}**else if**(**edtpass**.getText().toString().equals(**""**)){  
**edtpass**.requestFocus();  
**edtpass**.setError(**"Please enter password"**);  
}**else** {  
  
**username** = **edtuser**.getText().toString();  
**password** = **edtpass**.getText().toString();  
  
**loginDataBaseAdapter**.insertEntry(**username**, **password**);  
Toast.*makeText*(getApplicationContext(), **"Account Successfully Created "**, Toast.***LENGTH\_LONG***).show();  
}  
}  
});  
}  
@Override  
**protected void** onDestroy() {  
*//* ***TODO Auto-generated method stub***  
**super**.onDestroy();  
  
**loginDataBaseAdapter**.close();  
}  
}

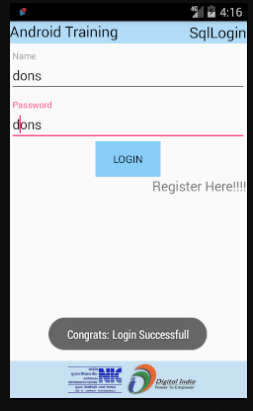
c. Add below code in " DatabaseHelper.java" file

**package** cs.app.nic.com.seminarclass;  
  
**import** android.content.Context;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.database.sqlite.SQLiteOpenHelper;  
**import** android.util.Log;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/12/2017.*  
*\*/*  
  
**public class** DataBaseHelper **extends** SQLiteOpenHelper {  
**public** DataBaseHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, **int** version)  
{  
**super**(context, name, factory, version);  
}  
*// Called when no database exists in disk and the helper class needs*  
*// to create a new one.*  
@Override  
**public void** onCreate(SQLiteDatabase \_db)  
{  
\_db.execSQL(LoginDataBaseAdapter.***DATABASE\_CREATE***);  
  
}  
*// Called when there is a database version mismatch meaning that the version*  
*// of the database on disk needs to be upgraded to the current version.*  
@Override  
**public void** onUpgrade(SQLiteDatabase \_db, **int** \_oldVersion, **int** \_newVersion)  
{  
*// Log the version upgrade.*  
Log.*w*(**"TaskDBAdapter"**, **"Upgrading from version "**+\_oldVersion + **" to "**+\_newVersion + **", which will destroy all old data"**);  
  
*// Upgrade the existing database to conform to the new version. Multiple*  
*// previous versions can be handled by comparing \_oldVersion and \_newVersion*  
*// values.*  
*// The simplest case is to drop the old table and create a new one.*  
\_db.execSQL(**"DROP TABLE IF EXISTS "**+ **"TEMPLATE"**);  
*// Create a new one.*  
onCreate(\_db);  
}  
  
}

d. Add below code in " LoginDatabaseAdapter.java" file

**package** cs.app.nic.com.seminarclass;  
  
**import** android.content.ContentValues;  
**import** android.content.Context;  
**import** android.database.Cursor;  
**import** android.database.SQLException;  
**import** android.database.sqlite.SQLiteDatabase;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/12/2017.*  
*\*/*  
  
**public class** LoginDataBaseAdapter  
{  
  
**static final** String ***DATABASE\_NAME*** = **"login.db"**;  
**static final int *DATABASE\_VERSION*** = 1;  
**public static final int *NAME\_COLUMN*** = 1;  
*//* ***TODO: Create public field for each column in your table.***  
*// SQL Statement to create a new database.*  
**static final** String ***DATABASE\_CREATE*** = **"create table "**+**"LOGIN"**+ **"( "**+**"ID"**+**" integer primary key autoincrement,"**+ **"USERNAME text,PASSWORD text); "**;  
*// Variable to hold the database instance*  
**public** SQLiteDatabase **db**;  
*// Context of the application using the database.*  
**private final** Context **context**;  
*// Database open/upgrade helper*  
**private** DataBaseHelper **dbHelper**;  
**public** LoginDataBaseAdapter(Context \_context)  
{  
**context** = \_context;  
**dbHelper** = **new** DataBaseHelper(**context**, ***DATABASE\_NAME***, **null**, ***DATABASE\_VERSION***);  
}  
**public** LoginDataBaseAdapter open() **throws** SQLException  
{  
**db** = **dbHelper**.getWritableDatabase();  
**return this**;  
}  
**public void** close()  
{  
**db**.close();  
}  
  
**public** SQLiteDatabase getDatabaseInstance()  
{  
**return db**;  
}  
  
**public void** insertEntry(String userName,String password)  
{  
ContentValues newValues = **new** ContentValues();  
*// Assign values for each row.*  
newValues.put(**"USERNAME"**, userName);  
newValues.put(**"PASSWORD"**,password);  
  
*// Insert the row into your table*  
**db**.insert(**"LOGIN"**, **null**, newValues);  
*///Toast.makeText(context, "Reminder Is Successfully Saved", Toast.LENGTH\_LONG).show();*  
}  
**public int** deleteEntry(String UserName)  
{  
*//String id=String.valueOf(ID);*  
String where=**"USERNAME=?"**;  
**int** numberOFEntriesDeleted= **db**.delete(**"LOGIN"**, where, **new** String[]{UserName}) ;  
*// Toast.makeText(context, "Number fo Entry Deleted Successfully : "+numberOFEntriesDeleted, Toast.LENGTH\_LONG).show();*  
**return** numberOFEntriesDeleted;  
}  
**public** String getSinlgeEntry(String userName)  
{  
Cursor cursor=**db**.query(**"LOGIN"**, **null**, **" USERNAME=?"**, **new** String[]{userName}, **null**, **null**, **null**);  
**if**(cursor.getCount()<1) *// UserName Not Exist*  
{  
cursor.close();  
**return "NOT EXIST"**;  
}  
cursor.moveToFirst();  
String password= cursor.getString(cursor.getColumnIndex(**"PASSWORD"**));  
cursor.close();  
**return** password;  
}  
**public void** updateEntry(String userName,String password)  
{  
*// Define the updated row content.*  
ContentValues updatedValues = **new** ContentValues();  
*// Assign values for each row.*  
updatedValues.put(**"USERNAME"**, userName);  
updatedValues.put(**"PASSWORD"**,password);  
  
String where=**"USERNAME = ?"**;  
**db**.update(**"LOGIN"**,updatedValues, where, **new** String[]{userName});  
}  
}

Now create login page when we enter name and password it checks in the db whether the values are present. If available it shows success.



a. Add below code in "LoginPage.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:orientation="vertical"android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**>  
  
<**include**  
**android:id="@+id/head"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_width="fill\_parent"**  
**layout="@layout/activity\_title"**  
  
/>  
  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/userlayout"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_below="@+id/head"**  
**android:layout\_marginTop="10dp"**>  
  
<**EditText**  
**android:id="@+id/usertext"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:ems="10"**  
**android:hint="Name"**/>  
</**android.support.design.widget.TextInputLayout**>  
  
<**android.support.design.widget.TextInputLayout**  
**android:id="@+id/passlayout"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_marginTop="10dp"**  
**android:layout\_below="@+id/userlayout"**>  
  
<**EditText**  
**android:id="@+id/passtext"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="wrap\_content"**  
**android:ems="10"**  
**android:hint="Password"**/>  
</**android.support.design.widget.TextInputLayout**>  
  
  
<**Button**  
**android:id="@+id/login"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="login"**  
**android:textColor="#000000"**  
**android:layout\_centerHorizontal="true"**  
**android:layout\_below="@+id/passlayout"**  
**android:background="@drawable/buttoncustom"**/>  
  
<**TextView**  
**android:id="@+id/textreg"**  
**android:layout\_width="wrap\_content"**  
**android:layout\_height="wrap\_content"**  
**android:text="Register Here!!!!"**  
**android:layout\_below="@+id/login"**  
**android:textSize="18dip"**  
**android:layout\_alignParentRight="true"**/>  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
  
</**RelativeLayout**>

b. Add below code in "SqlActivity.java" file.

**package** cs.app.nic.com.seminarclass;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.support.annotation.Nullable;  
**import** android.support.design.widget.TextInputLayout;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** org.w3c.dom.Text;  
  
*/\*\**  
*\* Created by asg4.dev13 on 6/12/2017.*  
*\*/*  
  
**public class** SqlActivity **extends** AppCompatActivity{  
TextInputLayout **txtlayoutuser**,**txtlayoutpass**;  
LoginDataBaseAdapter **loginDataBaseAdapter**;  
EditText **edtuser**,**edtpass**;  
Button **login**;  
String **username**,**password**;  
TextView **txt**;  
  
@Override  
**protected void** onCreate(@Nullable Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
setContentView(R.layout.***activty\_sqllogin***);  
  
**edtuser**=(EditText)findViewById(R.id.***usertext***);  
**edtpass**=(EditText)findViewById(R.id.***passtext***);  
**login**=(Button)findViewById(R.id.***login***);  
**txt**=(TextView) findViewById(R.id.***textreg***);  
**txtlayoutpass**=(TextInputLayout) findViewById(R.id.***passlayout***);  
**txtlayoutuser**=(TextInputLayout)findViewById(R.id.***userlayout***) ;  
View includedLayout = findViewById(R.id.***head***);  
  
TextView txttitle= (TextView)includedLayout.findViewById(R.id.***txttitile***);  
txttitle.setText(**"SqlLogin"**);  
  
*// create a instance of SQLite Database*  
**loginDataBaseAdapter**=**new** LoginDataBaseAdapter(SqlActivity.**this**);  
**loginDataBaseAdapter**=**loginDataBaseAdapter**.open();  
  
  
**login**.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
**username**=**edtuser**.getText().toString();  
**password**=**edtpass**.getText().toString();  
  
**if**(**edtuser**.getText().toString().equals(**""**)){  
**edtuser**.requestFocus();  
**edtuser**.setError(**"Plese enter username"**);  
  
}**else if**(**edtpass**.getText().toString().equals(**""**)){  
**edtpass**.requestFocus();  
**edtpass**.setError(**"please enter password"**);  
  
}**else**{  
login();  
}  
  
}  
  
**private void** login() {  
*// fetch the Password form database for respective user name*  
String storedPassword=**loginDataBaseAdapter**.getSinlgeEntry(**username**);  
  
*// check if the Stored password matches with Password entered by user*  
**if**(**password**.equals(storedPassword))  
{  
Toast.*makeText*(SqlActivity.**this**, **"Congrats: Login Successfull"**, Toast.***LENGTH\_LONG***).show();  
  
}  
**else**  
{  
Toast.*makeText*(SqlActivity.**this**, **"User Name or Password does not match"**, Toast.***LENGTH\_LONG***).show();  
}  
}  
});  
  
**txt**.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
Intent i= **new** Intent(SqlActivity.**this**,RegisterActivity.**class**);  
startActivity(i);  
}  
});  
  
}  
  
@Override  
**protected void** onDestroy() {  
**super**.onDestroy();  
*// Close The Database*  
**loginDataBaseAdapter**.close();  
}  
}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Google Map Loading**

***Google Map***

First Generate an API key from Google Api console from the link:

***https://console.developers.google.com/apis/library?project=seminar-1497258768606***

***Place the generated Api in manifest:***

<meta-data

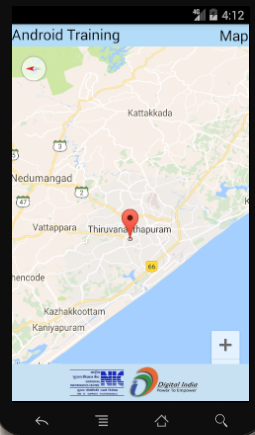
android:name="com.google.android.geo.API\_KEY"

android:value="@string/google\_maps\_key"

/>

Open build.gradle and add following code:

compile **'com.google.android.gms:play-services:6.5.87'**

****

In this exampole we are just loading the google map with given latitude and longitude.

a. Add below code in "activity\_map.xml" file.

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="fill\_parent"**>  
  
<**include**  
**android:id="@+id/head"**  
**android:layout\_height="wrap\_content"**  
**android:layout\_width="fill\_parent"**  
**layout="@layout/activity\_title"**  
  
/>  
  
  
<**fragment**  
**android:id="@+id/map"**  
**android:name="com.google.android.gms.maps.MapFragment"**  
**android:layout\_width="match\_parent"**  
**android:layout\_height="match\_parent"**  
**android:layout\_below="@+id/head"**  
/>  
<**LinearLayout**  
**android:id="@+id/in"**  
**android:layout\_width="fill\_parent"**  
**android:layout\_height="49dp"**  
**android:layout\_alignParentBottom="true"**  
**android:background="@drawable/fooo"**  
**android:orientation="horizontal"**  
  
/>  
  
  
</**RelativeLayout**>

b. Add below code in "MapActivity.java" file.

*package nic.kerala.training;*

*import android.Manifest;*

*import android.content.pm.PackageManager;*

*import android.graphics.Color;*

*import android.location.Location;*

*import android.support.annotation.NonNull;*

*import android.support.v4.app.ActivityCompat;*

*import android.support.v4.app.FragmentActivity;*

*import android.os.Bundle;*

*import android.support.v4.content.ContextCompat;*

*import android.widget.Toast;*

*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.CircleOptions;*

*import com.google.android.gms.maps.model.LatLng;*

*import com.google.android.gms.maps.model.MarkerOptions;*

*public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {*

*private GoogleMap mMap;*

*private static final int LOCATION\_PERMISSION\_REQUEST\_CODE = 1;*

*@Override*

*protected void onCreate(Bundle savedInstanceState) {*

*super.onCreate(savedInstanceState);*

*setContentView(R.layout.activity\_maps);*

*// Obtain the SupportMapFragment and get notified when the map is ready to be used.*

*SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()*

*.findFragmentById(R.id.map);*

*mapFragment.getMapAsync(this);*

*}*

*/\*\**

*\* Manipulates the map once available.*

*\* This callback is triggered when the map is ready to be used.*

*\* This is where we can add markers or lines, add listeners or move the camera. In this case,*

*\* we just add a marker near Sydney, Australia.*

*\* If Google Play services is not installed on the device, the user will be prompted to install*

*\* it inside the SupportMapFragment. This method will only be triggered once the user has*

*\* installed Google Play services and returned to the app.*

*\*/*

*@Override*

*public void onMapReady(GoogleMap googleMap) {*

*mMap = googleMap;*

*// Add a marker in Sydney and move the camera*

*LatLng sydney = new LatLng(-34, 151);*

*mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));*

*mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));*

*}*

*private void enableMyLocationIfPermitted() {*

*if (ContextCompat.checkSelfPermission(this,*

*Manifest.permission.ACCESS\_FINE\_LOCATION)*

*!= PackageManager.PERMISSION\_GRANTED) {*

*ActivityCompat.requestPermissions(this,*

*new String[]{Manifest.permission.ACCESS\_FINE\_LOCATION,*

*Manifest.permission.ACCESS\_FINE\_LOCATION},*

*LOCATION\_PERMISSION\_REQUEST\_CODE);*

*} else if (mMap != null) {*

*mMap.setMyLocationEnabled(true);*

*}*

*}*

*private void showDefaultLocation() {*

*Toast.makeText(this, "Location permission not granted, " +*

*"showing default location",*

*Toast.LENGTH\_SHORT).show();*

*LatLng redmond = new LatLng(47.6739881, -122.121512);*

*mMap.moveCamera(CameraUpdateFactory.newLatLng(redmond));*

*}*

*@Override*

*public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,*

*@NonNull int[] grantResults) {*

*switch (requestCode) {*

*case LOCATION\_PERMISSION\_REQUEST\_CODE: {*

*if (grantResults.length > 0*

*&& grantResults[0] == PackageManager.PERMISSION\_GRANTED) {*

*enableMyLocationIfPermitted();*

*} else {*

*showDefaultLocation();*

*}*

*return;*

*}*

*}*

*}*

*private GoogleMap.OnMyLocationButtonClickListener onMyLocationButtonClickListener =*

*new GoogleMap.OnMyLocationButtonClickListener() {*

*@Override*

*public boolean onMyLocationButtonClick() {*

*mMap.setMinZoomPreference(15);*

*return false;*

*}*

*};*

*private GoogleMap.OnMyLocationClickListener onMyLocationClickListener =*

*new GoogleMap.OnMyLocationClickListener() {*

*@Override*

*public void onMyLocationClick(@NonNull Location location) {*

*mMap.setMinZoomPreference(12);*

*CircleOptions circleOptions = new CircleOptions();*

*circleOptions.center(new LatLng(location.getLatitude(),*

*location.getLongitude()));*

*circleOptions.radius(200);*

*circleOptions.fillColor(Color.RED);*

*circleOptions.strokeWidth(6);*

*mMap.addCircle(circleOptions);*

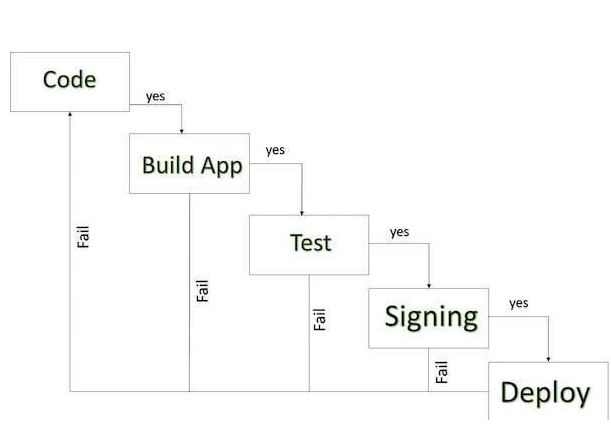
*}*

*};*

*}*

**Deployment of Android App**

Google Play is the premier store for distributing an Android app. After sweating out hours on creating a perfect app, you are now ready to put it out in front of over 1 billion active Android users in more than 190 countries and territories around the world. And, to make the process of publishing your app as smooth as it can be on Google Play Store, you can follow these steps.

****

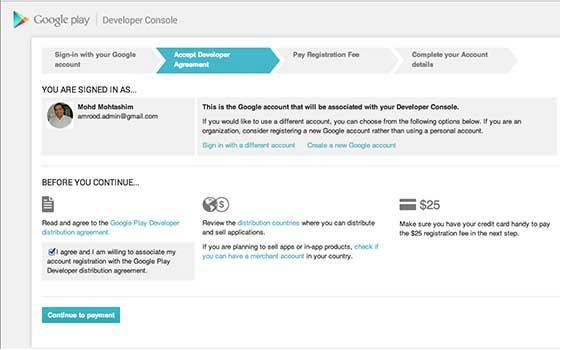
**1. Create an account**

To publish your app in the Google Play Store, you need to have an account with Google. You might already have a personal email account with them, but it is better to have a separate one to manage your app(s). While registering your publisher account, you would have to pay a registration fee of US$25.oo, using credit cards. After this, a verification mail would be sent to you; and then, you can sign in to your Developer Console, where all the action would take place.

**2. Familiarise yourself with Developer Console**

Google Play Developer Console is the starting point and the main dashboard for app publishing operations and tools. Before you go ahead get to know its interface, also familiarise yourself with the list of developer countries and merchant countries. The list of developer countries will tell you about all the locations where distribution to Google Play users is supported. And, if you want to sell apps, or have subscriptions or in-app purchases, then you need to review the list of merchant countries. Apart from this, take a look at the Developer Distribution Agreement, to avoid anything that violates the Google

Play’s terms and conditions.



**3. Fill in the necessary account details**

After this, log in to complete your account details. For example, you need to provide your Developer Name, the name which would be displayed in the Google Play Store. After filling in the details, you will

have to wait for a maximum of 48 hours for the Google Play Developer registration to be processed.

**4. Link your merchant account**

If you have a paid app, or one with subscriptions or in-app purchases, then you need to link your Google payments merchant account to your developer profile. Your linked account can be used for financial and tax identification as well as monthly payouts from sales.

**5. Upload your app**

Before uploading and submitting to Google Play team to publish your app, you need to generate a signed APK and some documents and image assets to fill up the requirements.

You will need to export your application as an APK (Android Package)

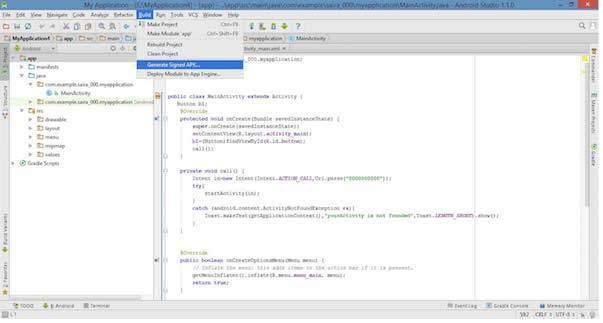
file before you upload it Google Play marketplace.

To export an application, just open that application project in **Android**

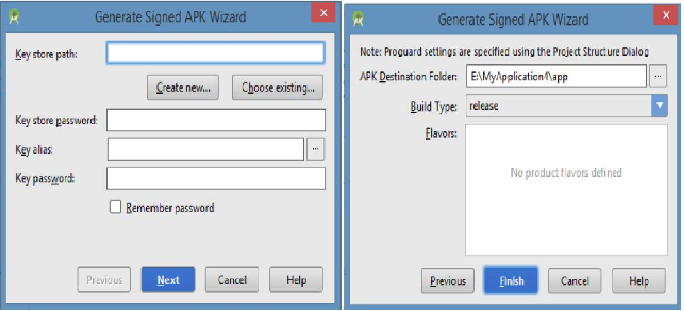
**studio and select Build →Generate Signed APK (As Shown in below**

**figure)** from your Android studio and follow the simple steps to export

your application.



Next select, Generate Signed APK option as shown in the above screen shot and then click it so that you get following screen where you will choose Create new key store to store your application.



Enter your key store path, key store password, key alias and key password to protect your application and click on Next button once again. It will display following screen to let you create an application.

Once you filled up all the information, like app destination, build type and flavors click finish button while creating an application it will show as below. It will generate your Android Application as APK format File

**6. Alpha and Beta testing the app**

Before you launch your app, it is essential to test it with a sample of end users to get their feedback; and the Google Play takes care of that as well. In the “APK” section of the Developer Console, you will find the options related to “Beta Testing” and “Alpha Testing”. After you have uploaded your app’s “.apk” file(s), you can use these options to receive a URL that can be shared with the testers. Using this link, testers can download your app’s alpha or beta version. You can use their feedback to optimise your app and make relevant changes in it before publishing it in the store.

**7. Provide details for store listing**

Now comes the most important part of uploading the app, as this is the place that will determine how it would be seen in Google Play. After uploading the “.apk” file of your app, go to the “Store Listing” tab.

Over there, you need to add the details of the app, like a “Short description” (of 12 to 80 characters) and a “Full description” (of up to

100 characters) of the app. Along with this, add screenshots, a link of the promo video (if you have one), contact details (website name, email id, phone number, etc.), categorisation (application type category and content rating), link to privacy policy and other such important details related to the app. After completing the relevant fields, press the “Save” button. You can update your store listing at any time, so do not panic if you have made some mistake or left out a field while filling up.

**Image Assets** are to be prepared as per the googlel play requirements **and uploaded.Content rating** is one of the mandatory activities to be completedbefore submitting for deployment. The content rating is done based on a questionnaire provided by Play store. Once the rating is given, this can be applied to your app.

A document format is prepared by NIC for getting the app deployed in Google play store account of NIC. This can be referred to get a better knowledge on deployment requirements

**8. Add pricing and distribution details**

Now, move to the next item, which is “Pricing & Distribution” and select whether it is a “Free” or a “Paid” app. Also, select the distribution countries and check the boxes stating that your app complies with the content guidelines. If your app is an educational one, then you can put it in the limelight using “Google Play for Education” option or if it is compatible with an Android TV, then you can add a Leanback launch over here. Once you are done, save the changes and move on to the next step.

**9. Publishing the application**

When all the three tabs- “APK”, “Store Listing” and “Pricing & Distribution”- have been filled and there appears a green check mark next to them, you are all ready to publish the app in Google Play. All you need to do is, click the “Publish this app” button under the “Ready to Publish” drop-down menu at the top right corner of the Developer Console. After you have hit the button, a confirmation bar would show up stating that your app would appear shortly in the Google Play Store. Once your app is published, you can update it as often as you want. You

can even make changes with the pricing, configuration and distribution options at any time.

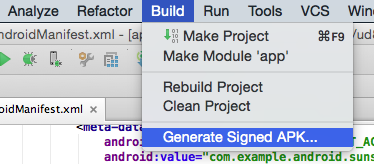
**10. Device Filtering Option**

These are a series of extra options that might not seem to be of much importance as you publish the app, but they can prevent your app from getting negative feedbacks. Through Google Play, you can control the distribution of your app according to the device features that are compatible with it. Allow only those devices to find your app in the Play Store that are compatible with your app. There is also an option to manually filter problematic or non-compatible devices, so make the most of it to stay on the top and filter out any negativity.

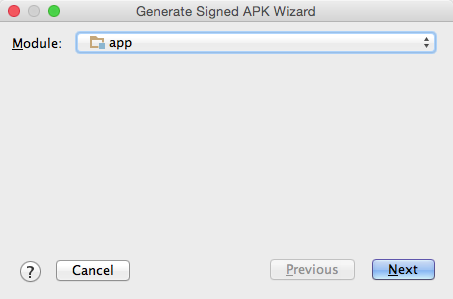
**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Creating a signed APK in Android Studio**

## *Click on* Build > Generate Signed APK *in the menu*



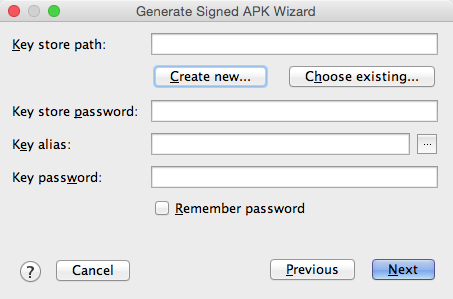
## *Click* Next *to build an APK for the app module*



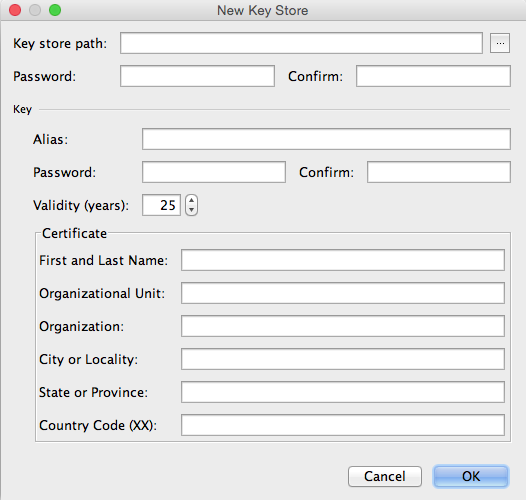
## *Creating a keystore*

A keystore is a file that acts as your signature. If you already have a keystore file, you may skip to Step 5 to finish building your signed APK.

## *Click on* Create New *to create a new keystore file*



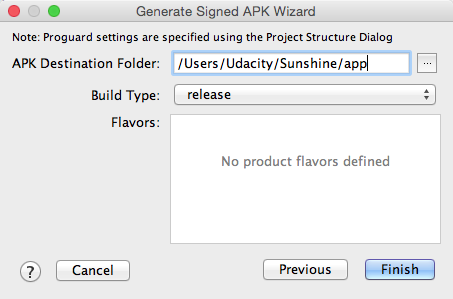
## *Fill in the fields to create your keystore and key as appropriate.*



You should save your keystore in a safe place on your computer. You’ll use this keystore to sign your APK, which is one of the requirements of putting your app on the Play Store. You should not share this file with anyone who doesn’t own your app. Remember to also add a password for your keystore and your key. A keystore can have multiple keys, hence why there are two different password fields here.

## *Generate your signed APK*

In the Generate Signed APK Wizard (see Step 3 for a screenshot), you can choose the appropriate keystore value, such as the one you just created, as well as the password fields. Then click Next. On the following screen, click Finish. After a moment, your signed APK will be created.



**Show Local Notification**

Add below code in "NotificationActivity.java" file.

import android.annotation.SuppressLint;

import android.app.Notification;

import android.app.NotificationChannel;

import android.app.NotificationManager;

import android.app.PendingIntent;

import android.content.Context;

import android.content.Intent;

import android.graphics.Color;

import android.os.Build;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v4.app.NotificationCompat;

import android.support.v7.app.AppCompatActivity;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

/\*\*

\* Created by asg4.dev13 on 6/12/2017.

\*/

public class NotificationActivity extends AppCompatActivity{

Button btn;

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_notification);

btn=(Button)findViewById(R.id.btun);

View includedLayout = findViewById(R.id.head);

TextView txttitle= (TextView)includedLayout.findViewById(R.id.txttitile);

txttitle.setText("Notification");

btn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

addNotification();

}

});

}

private void addNotification() {

NotificationManager notificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE);

String NOTIFICATION\_CHANNEL\_ID = "my\_channel\_id\_01";

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {

@SuppressLint("WrongConstant") NotificationChannel notificationChannel = new NotificationChannel(NOTIFICATION\_CHANNEL\_ID, "My Notifications", NotificationManager.IMPORTANCE\_MAX);

// Configure the notification channel.

notificationChannel.setDescription("Channel description");

notificationChannel.enableLights(true);

notificationChannel.setLightColor(Color.RED);

notificationChannel.setVibrationPattern(new long[]{0, 1000, 500, 1000});

notificationChannel.enableVibration(true);

notificationManager.createNotificationChannel(notificationChannel);

}

NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this, NOTIFICATION\_CHANNEL\_ID);

notificationBuilder.setAutoCancel(true)

.setDefaults(Notification.DEFAULT\_ALL)

.setWhen(System.currentTimeMillis())

.setSmallIcon(R.drawable.notification)

.setTicker("Hearty365")

// .setPriority(Notification.PRIORITY\_MAX)

.setContentTitle("Notifications Example")

.setContentText("This is a test notification")

.setContentInfo("Info");

// notificationManager.notify(/\*notification id\*/1, notificationBuilder.build());

Intent notificationIntent = new Intent(this, SqlActivity.class);

PendingIntent contentIntent = PendingIntent.getActivity(this, 0, notificationIntent,

PendingIntent.FLAG\_UPDATE\_CURRENT);

notificationBuilder.setContentIntent(contentIntent);

// Add as notification

NotificationManager manager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE);

manager.notify(0, notificationBuilder.build());

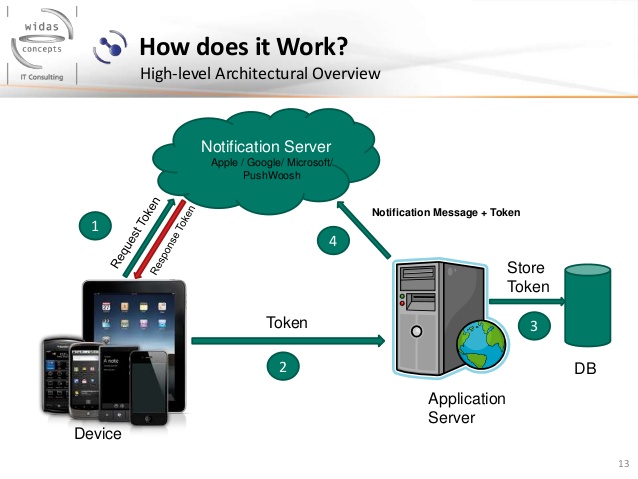
}

}

**Push Notification**

A notification is a message you can display to the user outside of your application's normal UI. When you tell the system to issue a notification, it first appears as an icon in the **notification area**. To see the details of the notification, the user opens the **notification drawer**. Both the notification area and the notification drawer are system-controlled areas that the user can view at any time.

Push Notification Architecture



For implementing Notification using **FCM (Firebase Cloud Messaging)** follow these steps:

Add following lines in build.gradle

classpath **'com.google.gms:google-services:3.1.0'**

**build.gradle(Module:app)**

apply **plugin**: **'com.android.application'**  
  
android {  
compileSdkVersion 25  
buildToolsVersion **"25.0.2"**  
defaultConfig {  
applicationId **"cs.app.nic.com.notificationexample"**  
minSdkVersion 15  
targetSdkVersion 25  
versionCode 1  
versionName **"1.0"**  
testInstrumentationRunner **"android.support.test.runner.AndroidJUnitRunner"**  
}  
buildTypes {  
release {  
minifyEnabled **false**  
proguardFiles getDefaultProguardFile(**'proguard-android.txt'**), **'proguard-rules.pro'**  
}  
}  
}  
  
dependencies {  
compile fileTree(**dir**: **'libs'**, **include**: [**'\*.jar'**])  
androidTestCompile(**'com.android.support.test.espresso:espresso-core:2.2.2'**, {  
exclude **group**: **'com.android.support'**, **module**: **'support-annotations'**  
})  
compile **'com.android.support:appcompat-v7:25.3.1'**  
compile **'com.android.support.constraint:constraint-layout:1.0.2'**  
testCompile **'junit:junit:4.12'**  
  
compile **'com.google.firebase:firebase-messaging:9.2.1'**  
  
}  
apply **plugin**: **'com.google.gms.google-services**

Step1:

1. ***Add below code in "activity\_main.xml" file.***

<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="cs.app.nic.com.notificationexample.MainActivity">

<TextView

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

android:text="your message"

/>

<TextView

android:id="@+id/txt"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

/>

</LinearLayout>

Step2:

Create a class to get the firebaseid

1. ***Add below code in "FirebaseId.java" file.***

**package** cs.app.nic.com.seminarclass;  
package cs.app.nic.com.notificationexample;

import android.util.Log;

import com.google.firebase.iid.FirebaseInstanceId;

import com.google.firebase.iid.FirebaseInstanceIdService;

/\*\*

\* Created by asg4.dev13 on 6/9/2017.

\*/

public class FirebaseId extends FirebaseInstanceIdService{

@Override

public void onTokenRefresh() {

String gettoken= FirebaseInstanceId.getInstance().getToken();

Log.d("id",gettoken);

}

}

**Step 3:**

Create a class to receive the message

c. Add below code in "FirebaseService.java" file.

package cs.app.nic.com.notificationexample;

import android.app.NotificationManager;

import android.app.PendingIntent;

import android.content.Context;

import android.content.Intent;

import android.support.v4.app.NotificationCompat;

import android.util.Log;

import com.google.firebase.messaging.FirebaseMessagingService;

import com.google.firebase.messaging.RemoteMessage;

/\*\*

\* Created by asg4.dev13 on 6/9/2017.

\*/

public class FirebaseService extends FirebaseMessagingService{

private static final String TAG = "FCM Service";

@Override

public void onMessageReceived(RemoteMessage remoteMessage) {

Log.d(TAG, "From: " + remoteMessage.getFrom());

Log.d(TAG, "Notification Message Body: " + remoteMessage.getNotification().getBody());

Intent intent=new Intent(this,MainActivity.class);

System.out.println("adda"+remoteMessage.getNotification().getBody().toString());

intent.putExtra("message", remoteMessage.getFrom().toString());

intent.setFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

PendingIntent pendingintent=PendingIntent.getActivity(this,0,intent,PendingIntent.FLAG\_ONE\_SHOT);

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

notificationbuilder.setContentTitle("From Notification");

notificationbuilder.setContentText(remoteMessage.getNotification().getBody());

notificationbuilder.setAutoCancel(true);

notificationbuilder.setContentIntent(pendingintent);

NotificationManager notificationManager=(NotificationManager)getSystemService(Context.NOTIFICATION\_SERVICE);

notificationManager.notify(0,notificationbuilder.build());

}

}

d. Add below code in "Main Activity.java" file.

package cs.app.nic.com.notificationexample;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.widget.TextView;

import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

TextView txt;

private BroadcastReceiver mRegistrationBroadcastReceiver;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

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

Bundle extras = getIntent().getExtras();

if(extras!=null) {

String messages = extras.getString("message");

System.out.println("asdasd:"+messages);

txt.setText(messages);

}

//mRegistrationBroadcastReceiver=new BroadcastReceiver() {

// @Override

// public void onReceive(Context context, Intent intent) {

// if (intent.getAction().equals("push")) {

// // new push notification is received

//

// String message = intent.getStringExtra("message");

//

// Toast.makeText(getApplicationContext(), "Push notification: " + message, Toast.LENGTH\_LONG).show();

//

// txt.setText(message);

// }

// }

//};

}

}

In manifest add two service classes'

*<?***xml version="1.0"encoding="utf-8"***?>*  
<**manifest xmlns:android="http://schemas.android.com/apk/res/android"**  
**package="cs.app.nic.com.notificationexample"**>  
  
<**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**>  
  
  
<**service android:name=".FirebaseId"**>  
<**intent-filter**>  
<**action android:name="com.google.firebase.INSTANCE\_ID\_EVENT"**></**action**>  
</**intent-filter**>  
</**service**>  
  
<**service android:name=".FirebaseService"**>  
<**intent-filter**>  
<**action android:name="com.google.firebase.MESSAGING\_EVENT"**></**action**>  
</**intent-filter**>  
</**service**>  
</**application**>  
  
</**manifest**>

Here we are implelementing notification using firebase cloud messaging (FCM). Now go to firebase console

https://console.firebase.google.com/?pli=1

Step1:

Create a project with this package name. Add google-services.json that we will get during creating a project in firebase console to the project.

Step2:

Select Notification in side menu and select new message. Enter firebase id generated while running the app and type the message in the space provided. Then send the message. On completion of sending message, we will recieve a Notification in phone. This is an example of implementing push notification through FCM.

Sending message from firebase console is not always an optimal solution for lot of apps. Huge number of apps requires an automatic notification to user whenever there is an event, which is not possible using firebase console. To overcome this problem you need to integrate the firebase API from your backend server.

# [How to Install Android Apps on Bluestacks](http://www.wikihow.com/Install-Android-Apps-on-Bluestacks)

BlueStacks App Player is an Android emulator and freely downloadable. BlueStacks App Player can be downloaded from the BlueStacks website, and it is available for both PC and Mac.

To install the .apk file in BlueStacks App Player

* + - right click on the .apk file
    - select “Open with Blue Stack APK Installer” option from the popup menu.
    - The .apk will get installed in the BlueStack App player.
    - Once it is installed, by clicking on the corresponding launch icon, you may view the app.

**Webview(HTML 5 & CSS3)**

WebView is a view that display web pages inside your android application. We can also specify HTML string and can show it inside our application using WebView. WebView turns our application to a web application.

Only Responsive Web Pages will be rendered properly in a webview.

It allows us to integrate a webpage as a part of the app. WebView comes with all the features that of a desktop browser like managing history, cookies, HTML5 support and lot more

**Responsive Web Design**

Responsive Web design is the approach that suggests that design and development should respond to the user’s behaviour and environment based on screen size, platform and orientation.

The practice consists of a mix of flexible grids and layouts, images and an intelligent use of CSS media queries. As the user switches from their laptop to iPad, the website should automatically switch to accommodate for resolution, image size and scripting abilities. In other words, the website should have the technology to automatically respond to the user’s preferences. This would eliminate the need for a different design and development phase for each new gadget on the market.

Popular Responsive Front End Templates:

* Bootstrap
* Foundation 3
* YAML
* Google Material Design

Apps developed Using WebView are termed as Hybrid Apps .Some of the pros and cons in using webview are:

Pros

* Easy development with simple business logic into single coding
* Re-usable coding for different mobile platforms
* Plenty of resources are available
* Based on HTML, Hybrid application is highly extensible with rich features, animations and functionalities
* compatible with different devices and multiple platforms
* we can change the UI, Logic everything from the website instead of releasing new APK's

Cons

* User-experience is relatively poor than Native Papplications
* Hybrid apps requires to implement distinct web view as per platforms
* Limited access to advanced features such as gyroscopes, geolocations, cameras, and much more
* Because of high resource consumption of browser’s Java Virtual Machine Hybrid apps perform with certain limitations

An example to show how you place a webviw wiget

<WebView android:id="@+id/webView“ android :layout\_width="match\_parent" android: layout\_height ="wrap\_content"/>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**An Introduction to Mobile Application Development using Cordova**

**Hybrid Application Frameworks**

* Xamarin- Build native apps for multiple platforms on a shared C# codebase. Microsoft-owned.
* Phonegap -Hybrid application build with HTML, CSS, and JavaScript. Adobe owned. Free
* Intel XDK
* Ionic Framework **-** Built on top of Angular.js and Apache Cordova.
* Framework7 **-** is a free and open source mobile HTML framework to develop hybrid mobile apps
* Appcelerator Titanium

**Why Cordova?**

* Providing a way to develop mobile applications using standard web technologies - HTML5, CSS3 and JavaScript.
* Build cross-platform mobile applications for iOS/Android, and other mobile platforms.

**Cordova or PhoneGap?**

* This first choice might be a confusing one but it’s really very simple. [PhoneGap](http://phonegap.com/) is a product owned by Adobe which currently includes additional build services, and it may or may not eventually offer additional services and/or charge payments for use in the future. Cordova is owned and maintained by Apache, and will always be maintained as an open source project. Currently they both have a very similar API

**Architecture**



* What you get with Cordova is simply a JavaScript API, which serves as a wrapper for native code and is consistent across devices. You can consider Cordova to be an application container with a web view, which covers the entire screen of the device. The web view used by Cordova is the same web view used by the native operating system. On iOS, this is the Objective-C UIWebView class; on Android, this is android.webkit.WebView.
* When the app is compiled, your code actually stays intact. The compiler just takes your code and makes it available to the web view for rendering.
* Building your code for iOS will produce an IPA file, for Android an APK file, and building for Windows Phone produces an APPX file.

**Pros**

* **Easy to Learn**
* **Access to Native Functionality** With Cordova, you have access to native device capabilities, such as the camera, contacts, geolocation, media, SMS, and many others.
* **Free** You don’t have to pay anything to use Cordova.
* **Open Source** Anyone can contribute to Cordova’s source code to make it better. Plugins are also open source and anyone can build custom plugins. This means that developers like yourself can easily install and use these plugins. Or you can build your own plugin and share it with the community.
* **Big Community**

**Write Once, Deploy Everywhere** Cordova compiles your app into a package file, which is required by most app stores. This means that apps created with Cordova can easily be deployed to the app store of your choosing. If you’re deploying to Android, Cordova creates an **APK** (Android Application Package) file. If you’re deploying to iOS, Cordova compiles to **IPA**. For Windows, it’s **APPX**

**Cons**

* Poor Documentation It’s hard to find information about really specific things, such as what packages you need to install with the Android SDK Manager. And when you look something up, the results point to information specific to different versions of Cordova. This is sometimes confusing for beginners as they might have a different version of Cordova installed and they’re looking at documentation for another version of Cordova.
* Slower Than Native Since apps built with Cordova are basically web apps that are contained in a web view, they don’t perform as well as their native counterparts. This means that there is a limit to what kind of apps you can build. For example, a video editing application is better built natively since it will heavily rely on the CPU and GPU to do its work.
* Frameworks Because Cordova is just a wrapper for a web application, it doesn’t come with the user interface components, animations, and other goodies that you find in most native applications. This means that you have to implement all of these on your own. That’s why many developer rely on frameworks like [Ionic](http://ionicframework.com/) or [Onsen UI](https://onsen.io/) for building the user interface of their applications.
* Bugs in Plugins Not every plugin is created equal. There are those that have bugs or don’t work as expected
* Not Every Device Is the Same Native device functionality is accessed through the use of plugins. Cordova exposes an API so that these plugins can be used in the web view, but not every device is the same. There are quirks on every device. To put it simply, not every option that you can set on a plugin will work on every device.

**Installing Cordova**

* **Java Development Kit** (JDK) version 1.7 or later
* **Android SDK Tools** - install Android Studio or optionally you can install Android SDK Tools (sdkmanager) only.
* **Install Node.js** -Cordova runs on the Node.js platform, which needs to be installed as the first step. [http://nodejs.org](http://nodejs.org/)
* **Install Git** - Git is a version control system, which is used by Cordova behind-the-scenes. [http://git-scm.com](http://git-scm.com/).
* **Add in Environment Variables** (As per installation path)–
  + - C:\Program Files\Java\jdk1.8.0\_11\bin
    - C:\Development\android-sdk\platform-tools
    - C:\Development\android-sdk\tools
    - C:\Program Files (x86)\nodejs\
    - C:\Program Files (x86)\Git\cmd
* **Step 1** : Install the cordova module using npm utility of Node.js. The cordova module will automatically be downloaded by the npm utility.
  + C:\>npm install -g cordova
* **Step 2** : Go to the directory where you maintain your source code, and create a cordova project:
  + C:\>cordova create hello com.example.hello HelloWorld
* **Step 3** : Add the platforms that you want to target your app. We will add the 'ios' and 'android' platform and ensure they get saved to config.xml:
  + C:\>cordova platform add android --save
* The **www** folder is where you will code your HTML / JavaScript application. Open the index.html file in a browser to see the default application created by the Cordova CLI.
* The **platforms** folder is where Cordova will build your application for different platforms (iOS, Android, etc). The contents of this folder will be automatically generated by the Cordova CLI, and you should never edit code in that directory.
* Plugins are installed in the **plugins** directory.
* Application parameters (name, author, etc) are stored in **config.xml**.

**Cordova plugins**

* A plugin is a bit of add-on code that provides JavaScript interface to native components. They allow your app to use native device capabilities beyond what is available to pure web apps.
* <https://cordova.apache.org/plugins/>
* https://www.npmjs.com/

Example:

* cordova plugin add cordova-plugin-camera
* cordova plugin add cordova-plugin-splashscreen

**Apk generation and signing**

* **Step 1:**

D:\projects\Phonegap\Example> cordova plugin rm org.apache.cordova.console –save add the --save so that it removes the plugin from the config.xml file.

* **Step 2:**

**(Not needed in Cordova 6.2.0 and above)**

To generate a release build for Android, we first need to make a small change to the AndroidManifest.xml file found in platforms/android. Edit the file and change the line:

<application android:debuggable="true" android:hardwareAccelerated="true" android:icon="@drawable/icon" android:label="@string/app\_name">and

change android:debuggable to false:

<application android:debuggable="false" android:hardwareAccelerated="true" android:icon="@drawable/icon" android:label="@string/app\_name">

* **Step 3:**

Now we can tell cordova to generate our release build:

D:\projects\Phonegap\Example> cordova build --release android

Then, we can find our unsigned APK file in platforms\android\build\outputs\apk.

* **Step 4**

Key Generation:

keytool -genkey -v -keystore <keystoreName>.keystore -alias <Keystore AliasName> -keyalg <Key algorithm> -keysize <Key size> -validity <Key Validity in Days>

Example: keytool -genkey -v -keystore NAME-mobileapps.keystore -alias NAMEmobileapps -keyalg RSA -keysize 2048 -validity 10000

Then the Key store has been generated with name as NAME-mobileapps.keystore

* **Step 5:**

Place the generated keystore in unsigned apk folder.

To sign the unsigned APK, run the jarsigner tool which is also included in the JDK:

jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore <keystorename <Unsigned APK file><Keystore Alias name>

Example: jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore NAME-mobileapps.keystore Example-release-unsigned.apk xxxxxmobileapps

Enter KeyPhrase as 'xxxxxxxx'

* **Step 6:**

Finally, we need to run the zip align tool to optimize the APK:

zipalign -v 4 Example-release-unsigned.apk Example.apk

Example:

D:\projects\Phonegap\Example\platforms\android\build\outputs\apk> C:\Phonegap\adt-bundle-windows-x86\_64-20140624\sdk\build-tools\android-4.4W\zipalign -v 4 Example-release-unsigned.apk Example.apk

**Mobile App Security Best Practices & App Deployment in Play store**

Mobile devices had a boom in 2016, with usage of iOS and Android handhelds growing steadily throughout the year. By contrast, desktop operating systems other than Windows 10 and OS X generally showed a decline in growth. The global mobile internet usage was higher than that of desktop systems as of October, and predicted that nearly 80% of internet usage will be mobile by 2018. As with any element of technology, more widespread usage leads to greater and more widespread threats, and mobility is no exception. We shall discuss in this session, the top ten vulnerabilities in the mobile devices/ applications.

**1. Unsecured devices**

This is a significant issue and, it has to be taken care by every mobile user. As per 2016 statistics, only about one-third of Android users lock their screens with a passcode.

The unlocked devices fall into the hands of malicious people can do harm to the owner if they keep confidential data in their mobiles. Someone could do havoc with a found or stolen phone since many people store account IDs and passwords with device apps such as email and Facebook.

Using complex passcodes (avoid trivial passwords such as your name, date of birth or a combination of such data) and encryption on the device can greatly help reduce the risk of stolen data or malicious mischief. Encryption on removable micro-SD cards which contain private data is especially a must.

**2. Lack of Mobile Device Management solutions in place**

Businesses can greatly benefit from the use of mobile device management (or MDM) solutions to provide enhanced security for mobile users, especially if a Bring Your Own Device (BYOD) policy is in place.

Microsoft Exchange Server, for instance, has options to block unauthorized devices from connecting to user mailboxes, can require passwords, and can remote wipe lost or stolen phones.

Other third party solutions can implement detailed and granular controls such as white listing and blacklisting applications, employing anti-malware protection, enforcing encryption, pushing out updates, enabling and disabling various device functions such as the camera, and in general providing a good standardization of device settings.

Consumer devices do have some measure of available management options; Apple and Android each provide a "find my device" feature for lost phones as well as the option to ring or erase said devices.

**3. Apps that request too many permissions**

Your devices should alert you when a newly-installed app wants access to various components of your phone, such as your contacts, or functions such as your location service. If you were installing a simple flashlight app, we normally get suspicious if it began making such requests when you launched it.

Whenever an app requests for permission, make sure the request is genuine and as required for the speficied app functionality.

**4. Outdated apps**

Outdated apps can invite security risks if attackers find exploitable code in them. Reputable app stores like Google Play and Apple's iTunes and App store have options so apps will update on their own.

To check and confirm this is working on Android, open the Google Play Store app, tap the Menu icon, choose Settings, and select Auto-update apps. You can choose to have apps update any time or exclusively over Wi-Fi.

For Apple's iOS 7, go to Settings and scroll down to iTunes and App Store. Tap this icon and scroll down towards the bottom until you see Automatic Downloads. You can activate automatic downloads for Music, Apps, Books and Updates and specify the update intervals (you should pick the most feasible option which will result in the most frequent updates).

It is better to avoid installing apps from unknown sources.

**5. Bogus app stores**

App stores besides iTunes and Google Play are many in the cyberworld.

Apple and Google vet the apps which they permit to be distributed via their stores, but still the people manage to fake apps which scammers and hackers try to slip past them. We should be judicious even with apps found in the Apple and Google Play stores.

Third party app stores can contain much more dangerous content which can be completely unscreened.

**6. Decoy apps**

There was a Pokeman Go guide actually contained a trojan which had the potential to steal confidential data which could then be sold. This app was made available from a third-party app store. Beware of similar apps which can contain Trojan to harvest your data.

As you can imagine, decoy apps will closely match current trends so as to attract as many victims as possible.

If you try searching the popular ‘Bhim’ App in play store, you will land in finding many similar app from other sources deployed in play store.

**7. Mobile app malware**

There are mobile app malware which can be installed deliberately or via exploited vulnerabilities. Legitimate and bogus apps alike may be subject to malware which can operate in many different ways and comes in several types of forms.

Exploitable vulnerabilities are bad enough, but other vulnerabilities can lead to the installation of high risk ransomware or device management tools.

A report discussed on TechRepublic in August listed five threats that had either emerged, or gotten worse, over the last few months:

Apps and operating systems should be kept up to date religiously to protect mobile devices from vulnerabilities or at the very least to maximize their protection.

Running anti-malware on your device makes sense along with keeping up on the latest mobile security news via sites like Security Week, ComputerWorld, DarkReading and, of course, TechRepublic.

It's also important to note that security experts often advise against rooting or jailbreaking phones, since this can render them more susceptible to risk. I realize technical people thrive on experimenting with and customizing their devices, but it's fair to point out the danger involved.

**8. Botnets**

Botnets are technological entities made up of many compromised mobile devices which can then be collectively harnessed like flying monkeys into performing nefarious tasks. For example, they can launch distributed denial of service (DDOS) attacks against websites in an attempt to extort money from the website owner.

Botnets can utilize a diverse array of other tricks; one such example discovered this August operated by checking a specific Twitter account periodically to receive commands, which are critical for impacted devices to operate in sync with one another. Other online elements such as blog pages or messaging system can also be used to control botnet participants. Botnets are even being sold online to bidders willing to pay for their utilization, a troubling sign of what lies to come.

Suspicious phone behavior such as a quickly draining battery can be a sign that your phone is part of a botnet. Also be on the lookout for network connectivity problems, sluggish performance, the presence of unknown applications or the absence of known ones, and text or email communications sent without your knowledge and consent.There is also virus scan engines which can be used to check for malware on your phone.

**9. Exploding Notes**

Not all mobile risks are security-related. The infamous Samsung Note 7 phone was recalled in October after several dozen devices caught fire. In fact, the Federal Aviation Administration (FAA) actually banned the Note 7 from being taken on board all aircraft flights.

Samsung began offering extra money to Note7 owners for turning their phones in and recently confirmed that it will shortly release a software update which will completely disable these phones, knocking them out of service for good.

It is always better to wait a bit before buying a new device model, to see if design or operational issues such as these begin rearing their heads.

**10. Open Wi-Fi networks**

The proverbial Wi-Fi in the coffee shop was just the beginning. Wi-Fi connectivity is everywhere now. Restaurants, airports, stadiums hotels and many other places where groups of people congregate offer free open Wi-Fi. It's like that old saying about how you can never have too much money or be too thin. There's no such thing as too many Wi-Fi networks, especially for devices without global data access.

However, Wi-Fi availability isn't necessarily a good thing at times. Open, unsecured Wi-Fi networks can pose a significant security risk, especially if you're using credentials to log into other sites or accessing confidential data. Wi-Fi traffic can be "sniffed" and unencrypted data

may end up stolen — including passwords, credit card data or other sensitive details.

If you're logging into your bank account of official sites using open wi-fi from unknow sources, you may be at risk revealing your confidential information and assets to unauthorized persons. If you're doing something which might lead data or credentials to fall into the wrong hands, make sure to do it over a virtual private network (VPN) or use your own Wi-Fi hotspot or some other secure means for access

**Best Practises with Android App Development**

**Some Tips For Developers**

* Use a different package name other than “com.example” as it is default one. The build used with this type package name will be rejected by Google Play at review time.
* Keep safely the keystore file used for app signing as it is required for future version updating of the same app.
* Always keep track of version code and version name
* While generating signed APK remove all commands used for debug purpose like log.e, print.trace, etc.
* Clean your code before generating the signed APK. Keep only the codes needed
* Keep a version control table in server to manage update

**Keep a watch on Apk size**

* + APK size should be small (keep it under 10 MB)
  + Optimize graphic resources bundled in the APK. (Use vector images or WebP instead of png/jpg). Avoid large background images
  + Keep multiple builds for download targeting the low end and high end devices
  + ProGaurd or similar tools can reduce the size of APK
  + Take care while using external libraies; often such libraries will make your APK fatty
  + Allow app to be installed in external storage by setting appropriae flags in manifest.
  + Always store the database in external storage

**App Presentation for in different devices**

* + Use density independent pixel (dp) units instead of pixel (px) units while defining app layouts.
  + Use material design guidelines to ensure scaling of layout w.r.t. screen densities.
  + Ensure that the app layouts work on small and medium size screens. Focus on showing the essential information first
  + Ensure the graphics and text work properly with varying screen densities.
  + Test the graphics and layouts on screens with m/ldpi and small/medium screen sizes.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***