How to use the JavaMail API in your android applications

What is JavaMail API?

The JavaMail API is a built in feature in JAVA SE which is used to compose, write and read emails. In simple words it provides a simple platform and protocol independent framework for email communication. The core of this framework is originally found and integrated in *javax.mail* and *javax.mail.activation* packages.

Why would I use it in my applications?

Android documentation provides simple ways to send and receive emails.
https://developers.google.com/gmail/api/guides/sending However, this method essentially redirects the user to email clients present on the device and let user has the complete control over seding “what” content to “whom”. What if you want to send email to a particular email ID formatted to your, as a developer’s preferences. That’s where JavaMail slides in. To achieve the following task from scratch you would require to configure a crap load of stuff including connection adapters, networking protocols, application interfaces. JavaMail API is specifically written for android for this purpose. Someone has done all the hard work for us and we need just to follow through.

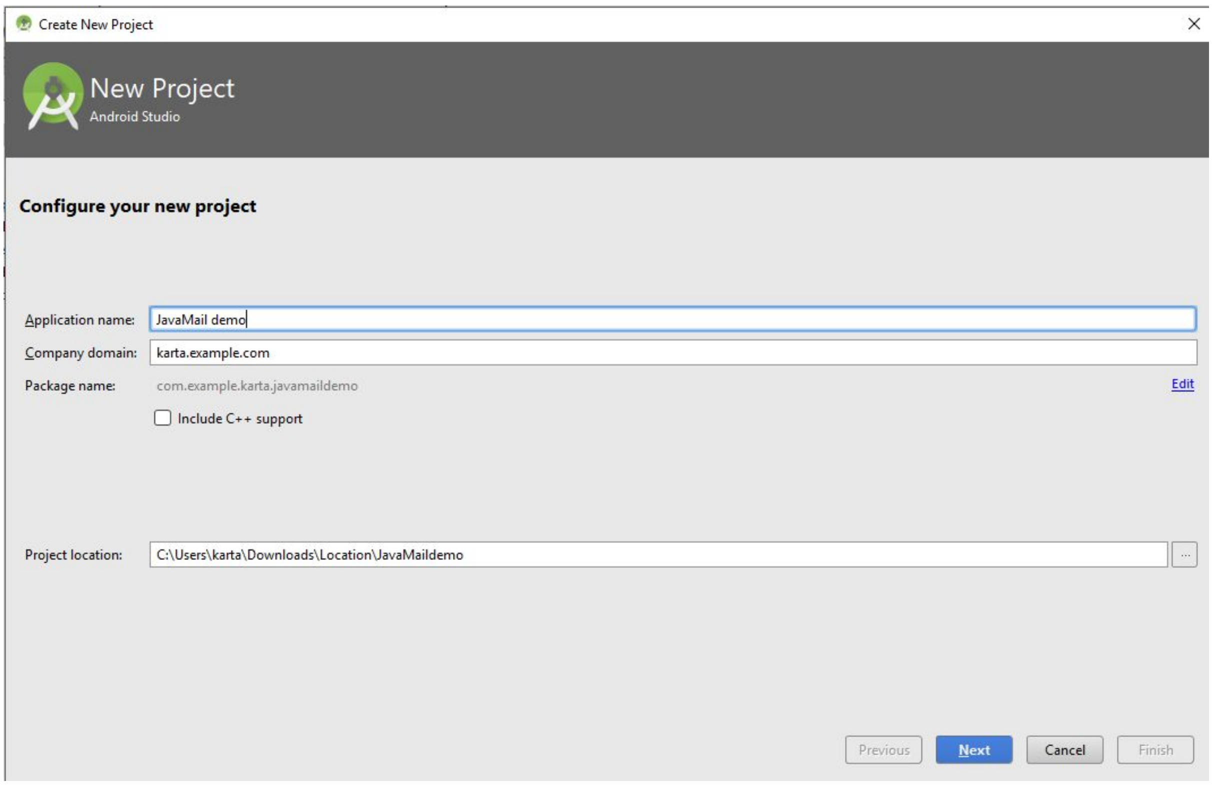
Where would I find the need to use this facility?

Say for example you are developing a car service application. When an user requires emergency vehicle services, they open the application and click on a button “GET CAR AID”. The application sends an email to you about the user’s location and contact details. This is one place where you would require an automated mailing service. Another important example could be accepting enquiries or feedback about a particular service. The list goes on and on. So what are we waiting for, let’s get started!

In this tutorial we will build an android application from scratch implementing JavaMail API. The application will accept text from user and send a mail automatically to a predefined email address.

Download the required addtional.jar, mail.jar, activation.jar files from https://code.google.com/archive/p/javamail-android/downloads we will use these later in our project.

Start a new android project:



Configure your new project

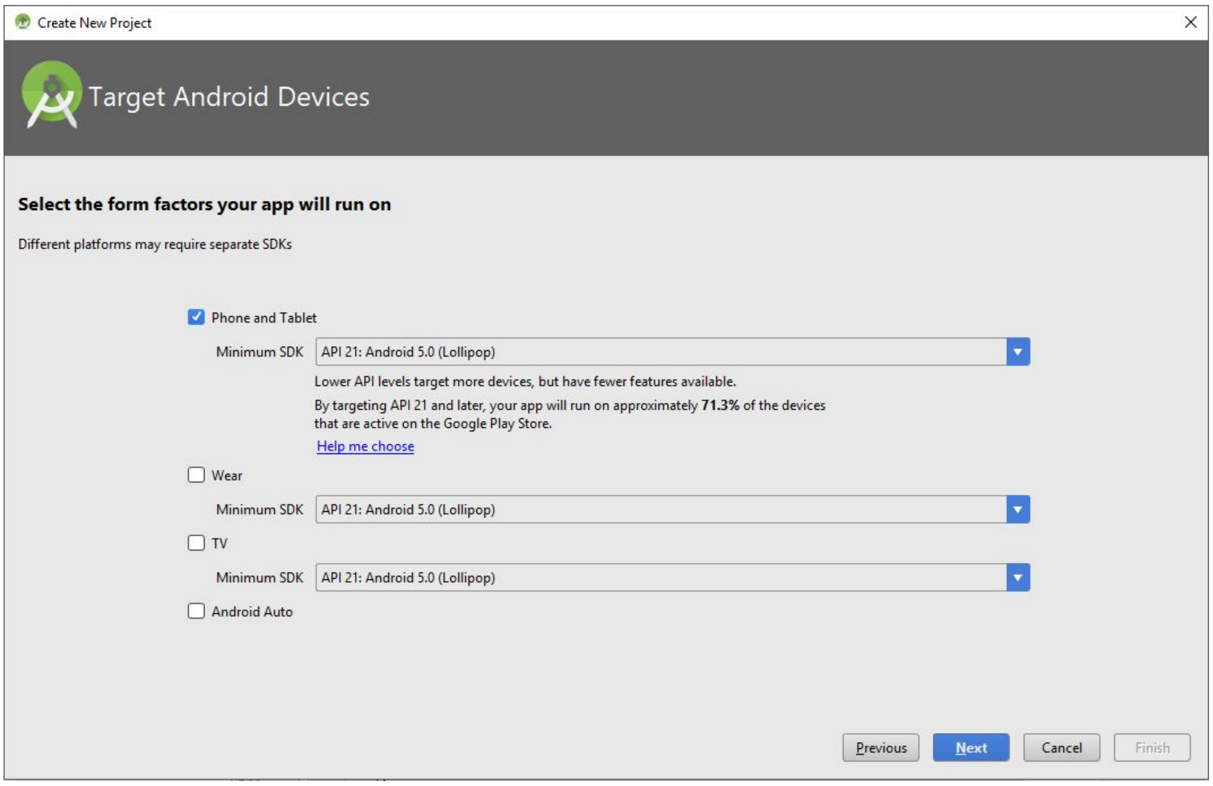
Application name: JavaMail demo

Company domain: karta.example.com

Package name: com.example.karta.javamaildemo

☐ Include C++ support

Project location: C:\Users\karta\Downloads\Location\JavaMaildemo



Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet

Minimum SDK: API 21: Android 5.0 (Lollipop)

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

By targeting API 21 and later, your app will run on approximately **71.3%** of the devices that are active on the Google Play Store.

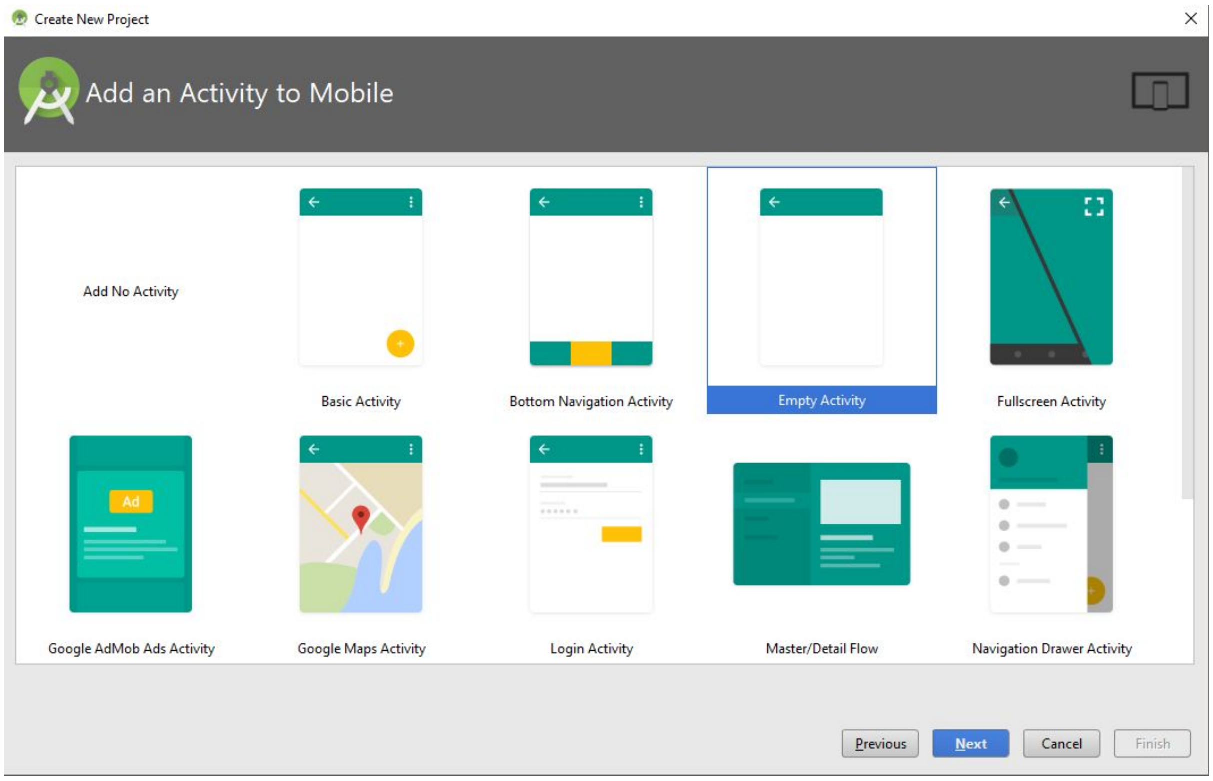
☐ Wear

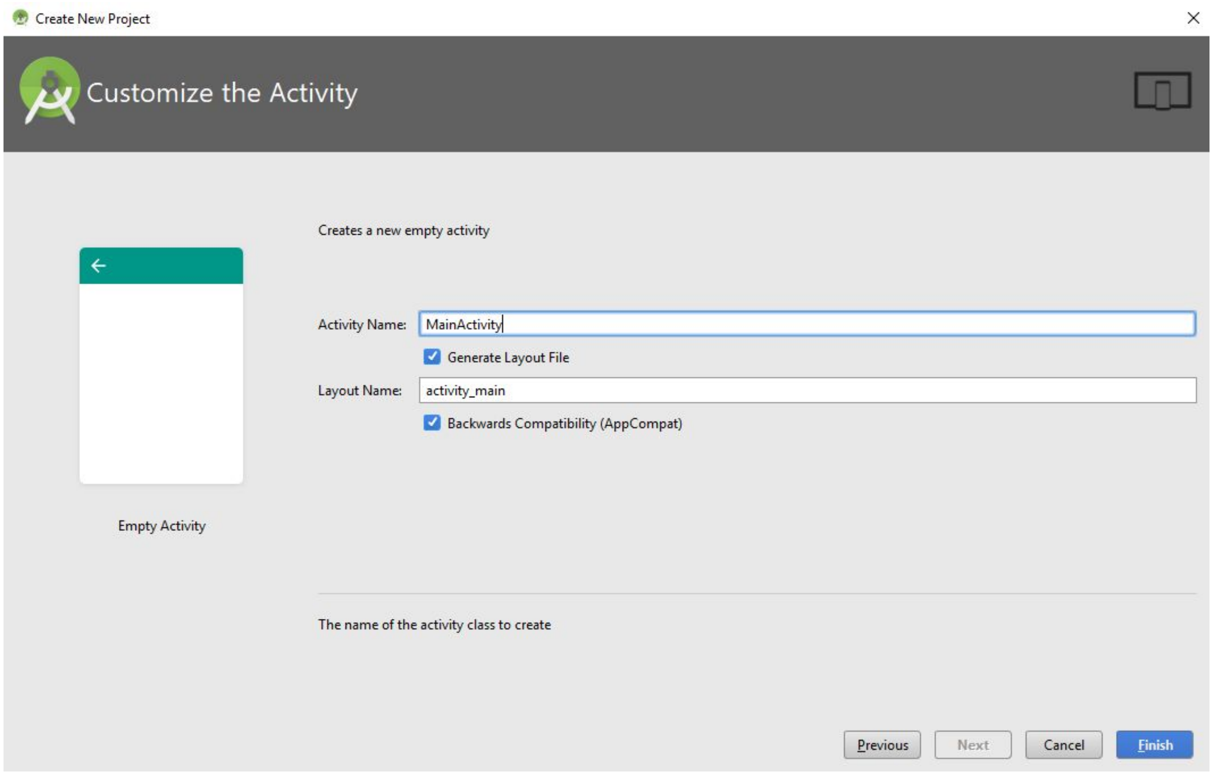
Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ Android Auto





Creates a new empty activity

Activity Name: MainActivity

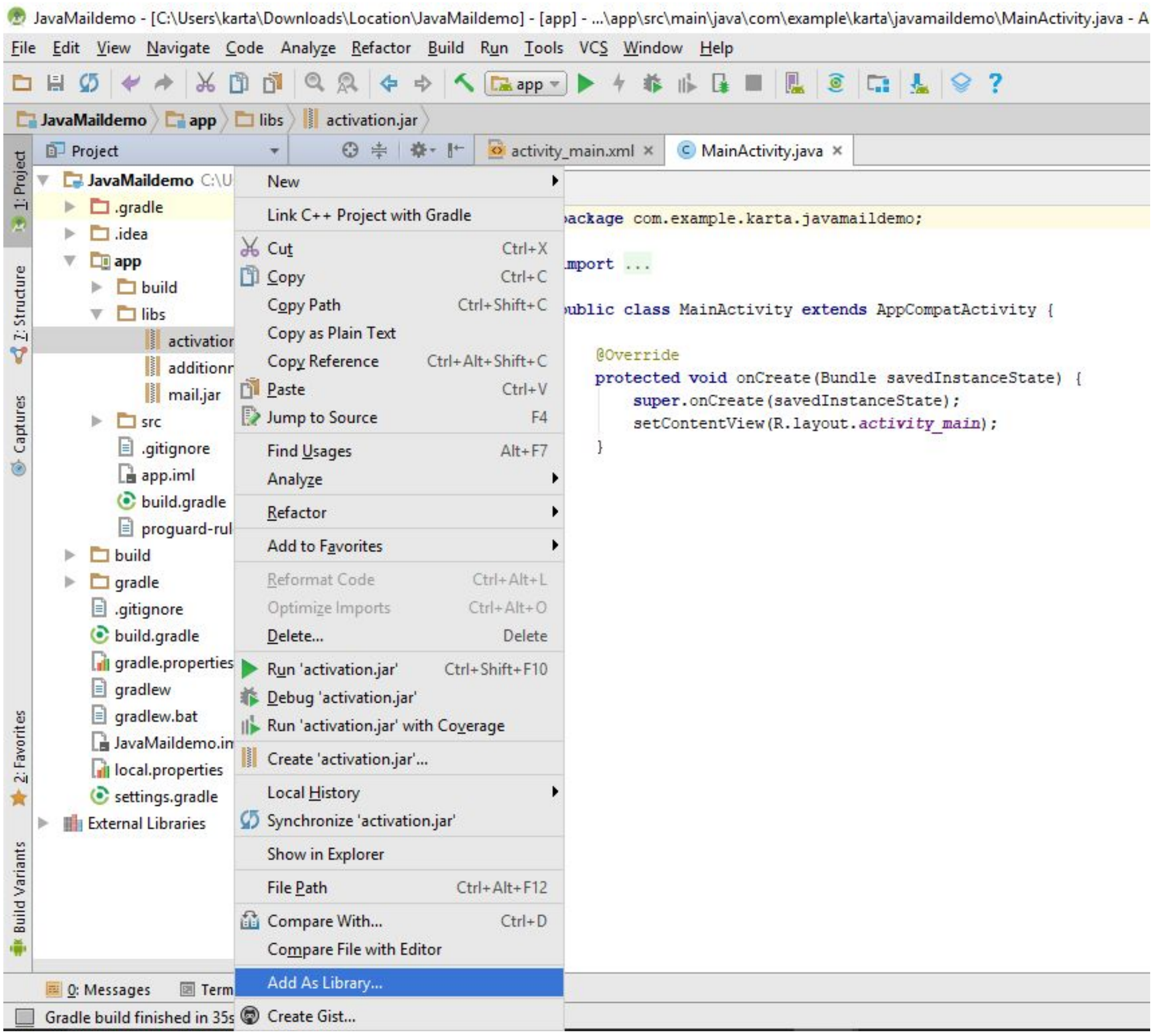
☑ Generate Layout File

Layout Name: activity_main

☑ Backwards Compatibility (AppCompat)

The name of the activity class to create

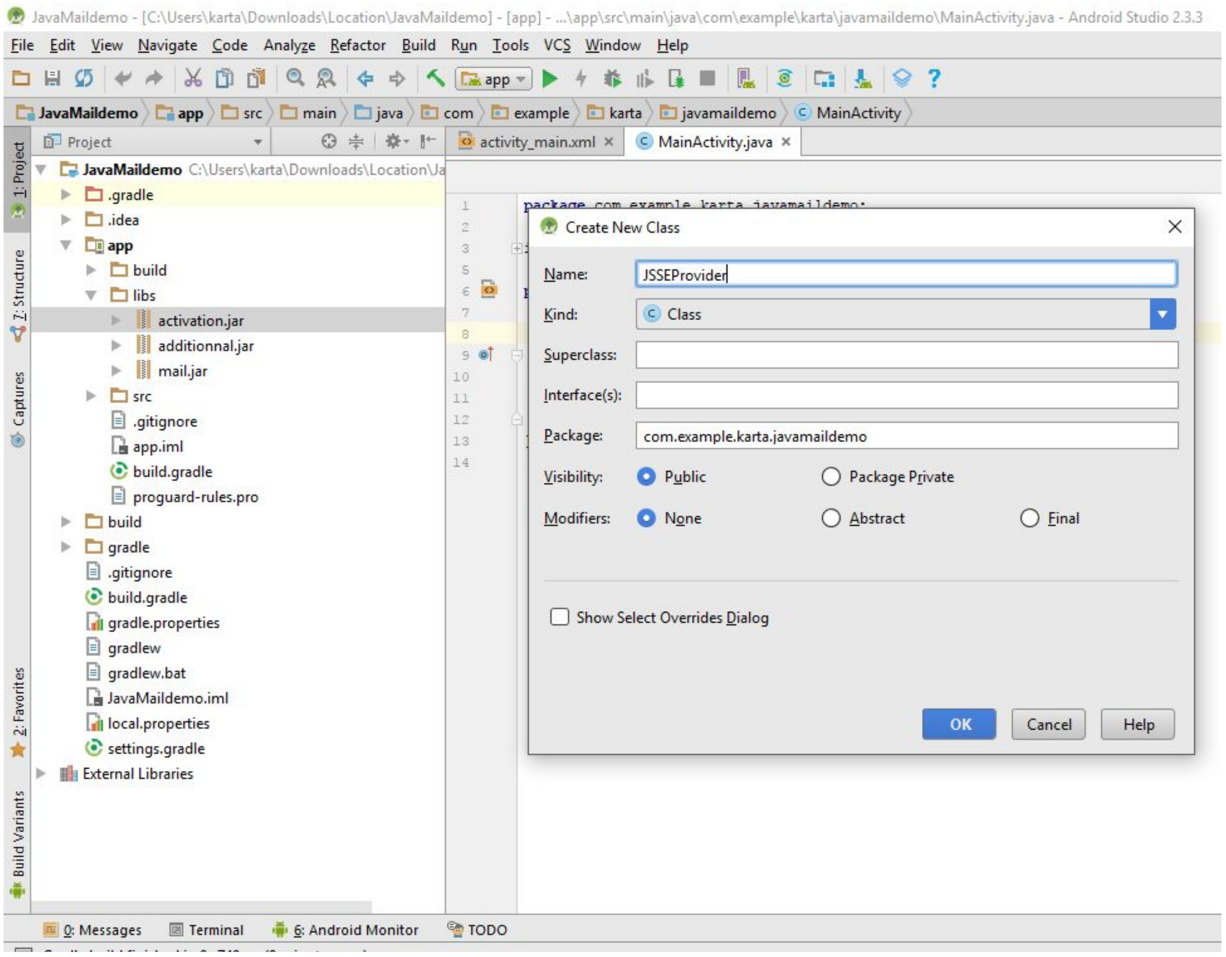
# In the app directory, find the lib folder and add the three .jar files we downloaded
Right click any .jar file and click add to library:



```
ackage com.example.karta.javamaildemo;

mport ...

ublic class MainActivity extends AppCompatActivity {

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

Now let’s add the framework files which will take care of the framework configurations

- *Make a new class JSSEProvider with the following code (code for JSSEProvider.java)*


JavaMaildemo - [C:\Users\karta\Downloads\Location\JavaMaildemo] - [app] - ...\app\src\main\java\com\example\karta\javamaildemo\MainActivity.java - Android Studio 2.3.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
JavaMaildemo app src main java com example karta javamaildemo MainActivity
Project
activity_main.xml
MainActivity.java
JavaMaildemo C:\Users\karta\Downloads\Location\Ja
.gradle
.idea
app
build
libs
activation.jar
additionnal.jar
mail.jar
src
.gitignore
app.iml
build.gradle
proguard-rules.pro
build
gradle
.gitignore
build.gradle
gradle.properties
gradlew
gradlew.bat
JavaMaildemo.iml
local.properties
settings.gradle
External Libraries
0: Messages
Terminal
6: Android Monitor
TODO

Create New Class

Name: JSSEProvider
Kind: Class
Superclass:
Interface(s):
Package: com.example.karta.javamaildemo
Visibility: [✓ Public] Package Private
Modifiers: [✓ None] Abstract Final
Show Select Overrides Dialog

OK Cancel Help

```
package com.example.karta.javamaildemo;

import java.security.AccessController;
import java.security.Provider;

public final class JSSEProvider extends Provider {

   public JSSEProvider() {
       super("HarmonyJSSE", 1.0, "Harmony JSSE Provider");

       AccessController.doPrivileged(new java.security.PrivilegedAction<Void>() {
           public Void run() {

put("SSLContext.TLS","org.apache.harmony.xnet.provider.jsse.SSLContextImpl");
               put("Alg.Alias.SSLContext.TLSv1", "TLS");

put("KeyManagerFactory.X509","org.apache.harmony.xnet.provider.jsse.KeyManagerFacto
ryImpl");
put("TrustManagerFactory.X509","org.apache.harmony.xnet.provider.jsse.TrustManagerF
actoryImpl");
               return null;
           }
       });
   }
}
```

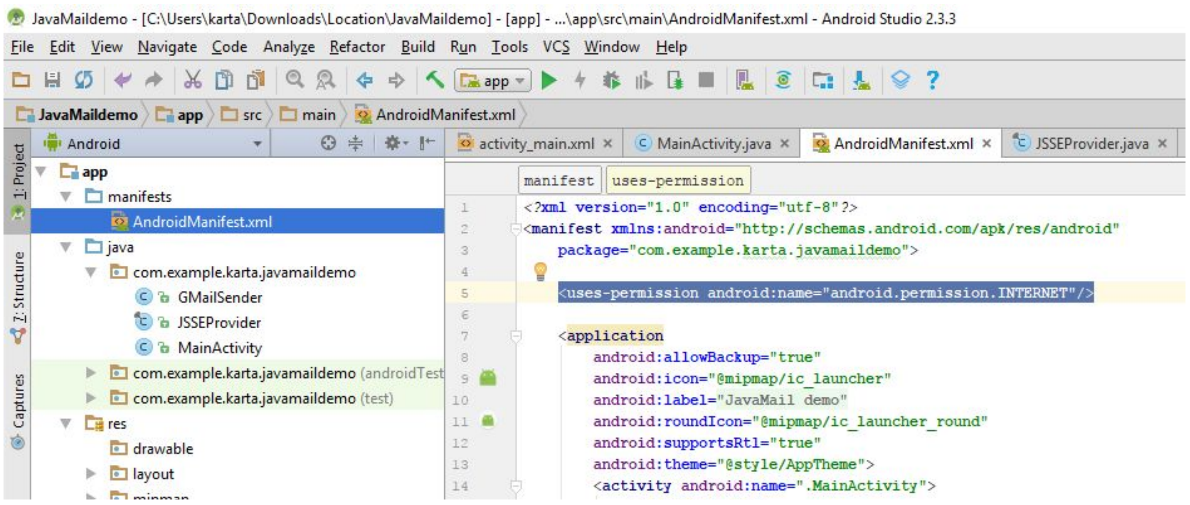
- *Make a new class GMailSender with the following code(code for GMailSender.java)*

```
package com.example.karta.javamaildemo;

import javax.activation.DataHandler;
import javax.activation.DataSource;
import javax.activation.FileDataSource;
import javax.mail.BodyPart;
import javax.mail.Message;
import javax.mail.Multipart;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.security.Security;
import java.util.Properties;


public class GMailSender extends javax.mail.Authenticator {

   static {
       Security.addProvider(new com.example.karta.javamaildemo.JSSEProvider());
   }

   private String mailhost = "smtp.gmail.com";
   private String user;
   private String password;
   private Session session;
   private Multipart _multipart = new MimeMultipart();


   public GMailSender(String user, String password) {

       this.user = user;
       this.password = password;

       Properties props = new Properties();
```

```
        props.setProperty("mail.transport.protocol", "smtp");
        props.setProperty("mail.host", mailhost);
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "465");
        props.put("mail.smtp.socketFactory.port", "465");
        props.put("mail.smtp.socketFactory.class","javax.net.ssl.SSLSocketFactory");
        props.put("mail.smtp.socketFactory.fallback", "false");
        props.setProperty("mail.smtp.quitwait", "false");

        session = Session.getDefaultInstance(props, this);
    }

    protected PasswordAuthentication getPasswordAuthentication() {
        return new PasswordAuthentication(user, password);
    }

    public synchronized void sendMail(String subject, String body, String sender,
String recipients) throws Exception {
        try {
            MimeMessage message = new MimeMessage(session);
            DataHandler handler = new DataHandler(new
ByteArrayDataSource(body.getBytes(), "text/plain"));
            message.setSender(new InternetAddress(sender));
            message.setSubject(subject);
            message.setDataHandler(handler);
            BodyPart messageBodyPart = new MimeBodyPart();
            messageBodyPart.setText(body);
            _multipart.addBodyPart(messageBodyPart);

            // Put parts in message
            message.setContent(_multipart);
            if (recipients.indexOf(',') > 0)
                message.setRecipients(Message.RecipientType.TO,
                        InternetAddress.parse(recipients));
            else message.setRecipient(Message.RecipientType.TO,new
InternetAddress(recipients));

            Transport.send(message);
        } catch (Exception e) {
            throw e;
        }
    }

    public void addAttachment(String filename) throws Exception {
        BodyPart messageBodyPart = new MimeBodyPart();
        DataSource source = new FileDataSource(filename);
        messageBodyPart.setDataHandler(new DataHandler(source));
        messageBodyPart.setFileName("download image");
        _multipart.addBodyPart(messageBodyPart);
    }

    public class ByteArrayDataSource implements DataSource {
        private byte[] data;
        private String type;
        public ByteArrayDataSource(byte[] data, String type) {
            super();
            this.data = data;
            this.type = type;
        }

        public ByteArrayDataSource(byte[] data) {
            super();
            this.data = data;
        }

        public void setType(String type) {
            this.type = type;
        }

        public String getContentType() {
            if (type == null)
                return "application/octet-stream";
            else
                return type;
        }

        public InputStream getInputStream() throws IOException {
            return new ByteArrayInputStream(data);
        }

        public String getName() {
            return "ByteArrayDataSource";
        }

        public OutputStream getOutputStream() throws IOException {
            throw new IOException("Not Supported");
        }
    }
}
```

# Let’s now define required permissions in the manifest file (manifest.xml):

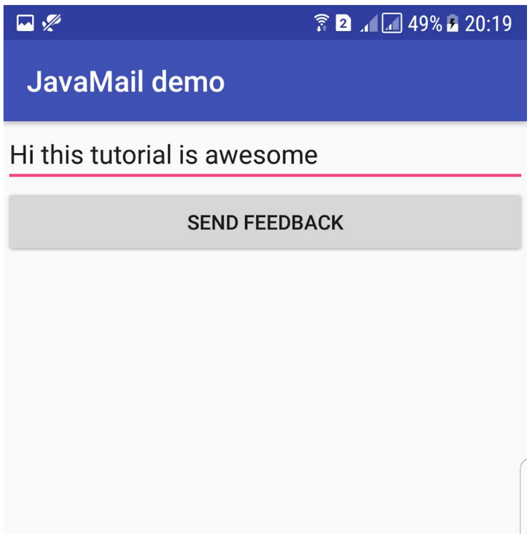
```
<uses-permission android:name="android.permission.INTERNET"/>
```


JavaMaildemo - [C:\Users\karta\Downloads\Location\JavaMaildemo] - [app] - ...\app\src\main\AndroidManifest.xml - Android Studio 2.3.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
JavaMaildemo app src main AndroidManifest.xml
Android
activity_main.xml MainActivity.java AndroidManifest.xml JSSEProvider.java
manifest uses-permission
app
manifests
AndroidManifest.xml
java
com.example.karta.javamaildemo
GMailSender
JSSEProvider
MainActivity
com.example.karta.javamaildemo (androidTest
com.example.karta.javamaildemo (test)
res
drawable
layout
1: Project
Z: Structure
Captures

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

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="JavaMail demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
```

WE NOW HAVE OUR BASIC FILES READY!

Let us now create a text box area and a button in the layout (Code for activity_layout.xml):



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

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

       <EditText
           android:id="@+id/feedbackEditText"
           android:layout_width="match_parent"
           android:layout_height="wrap_content"
           android:ems="10"
           android:hint="Enter Feedback"
           android:inputType="textPersonName" />

       <Button
           android:id="@+id/submitButton"
           android:layout_width="match_parent"
           android:layout_height="wrap_content"
           android:text="Send feedback" />
   </LinearLayout>
</android.widget.RelativeLayout>
```

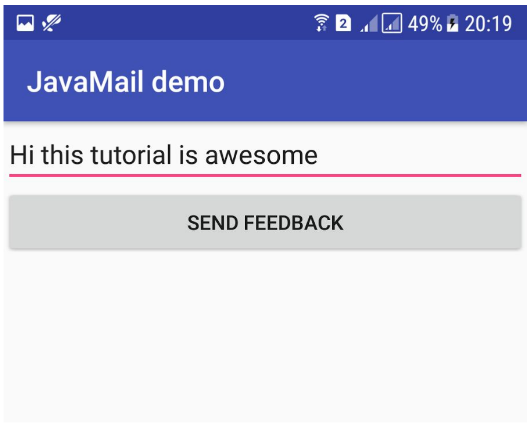
In the MainActivity class put in the following code:

Make sure to update the string values (Email id, Password) commented out

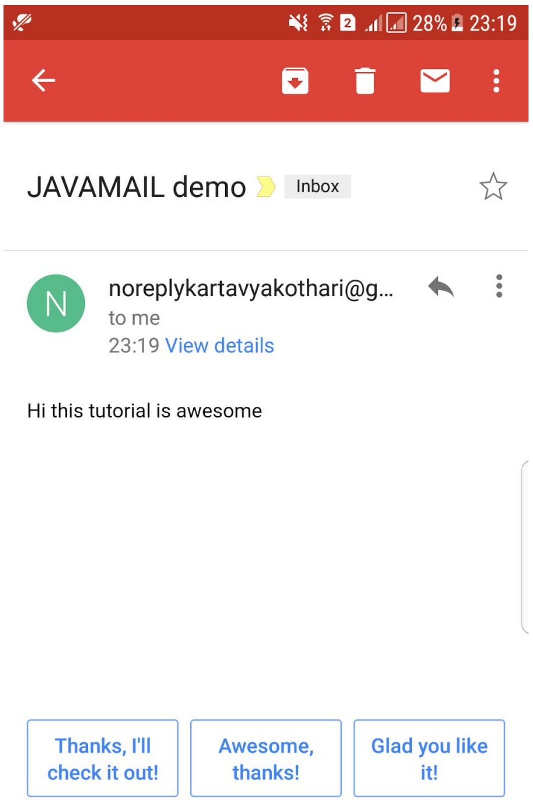
```
package com.example.karta.javamaildemo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

   Button submitButton;
   EditText feedbackEditText;
   String message;
   String SENDING_MAIL_TO;
   String SENDING_MAIL_FROM;
   String PASSWORD;

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

       //Update with your string values
       SENDING_MAIL_FROM = "YOUR GMAIL ID";
       SENDING_MAIL_TO = "WHERE DO YOU WANNA SEND THE MAIL";
       PASSWORD= "YOUR GMAIL PASSWORD";

       feedbackEditText = (EditText) findViewById(R.id.feedbackEditText);

       submitButton = (Button) findViewById(R.id.submitButton);
       submitButton.setOnClickListener(new View.OnClickListener() {
           @Override
           public void onClick(View view) {
               message = feedbackEditText.getText().toString();

               new Thread(new Runnable() {
                   public void run() {
                       try {
                           GMailSender sender = new
GMailSender(SENDING_MAIL_FROM,PASSWORD);

//sender.addAttachment(Environment.getExternalStorageDirectory().getPath()+"/image.
jpg");
                           sender.sendMail("This is a demo app",message,
                                   SENDING_MAIL_FROM,SENDING_MAIL_TO);
                       } catch (Exception e) {

Toast.makeText(getApplicationContext(),"Error",Toast.LENGTH_LONG).show();
                       }
```

```
                        }
                    }).start();
                }
            });
        }
}
```

# The application is now ready!



# Click on send feedback and open the mail you have set it to be sent to:



Hi this tutorial is awesome