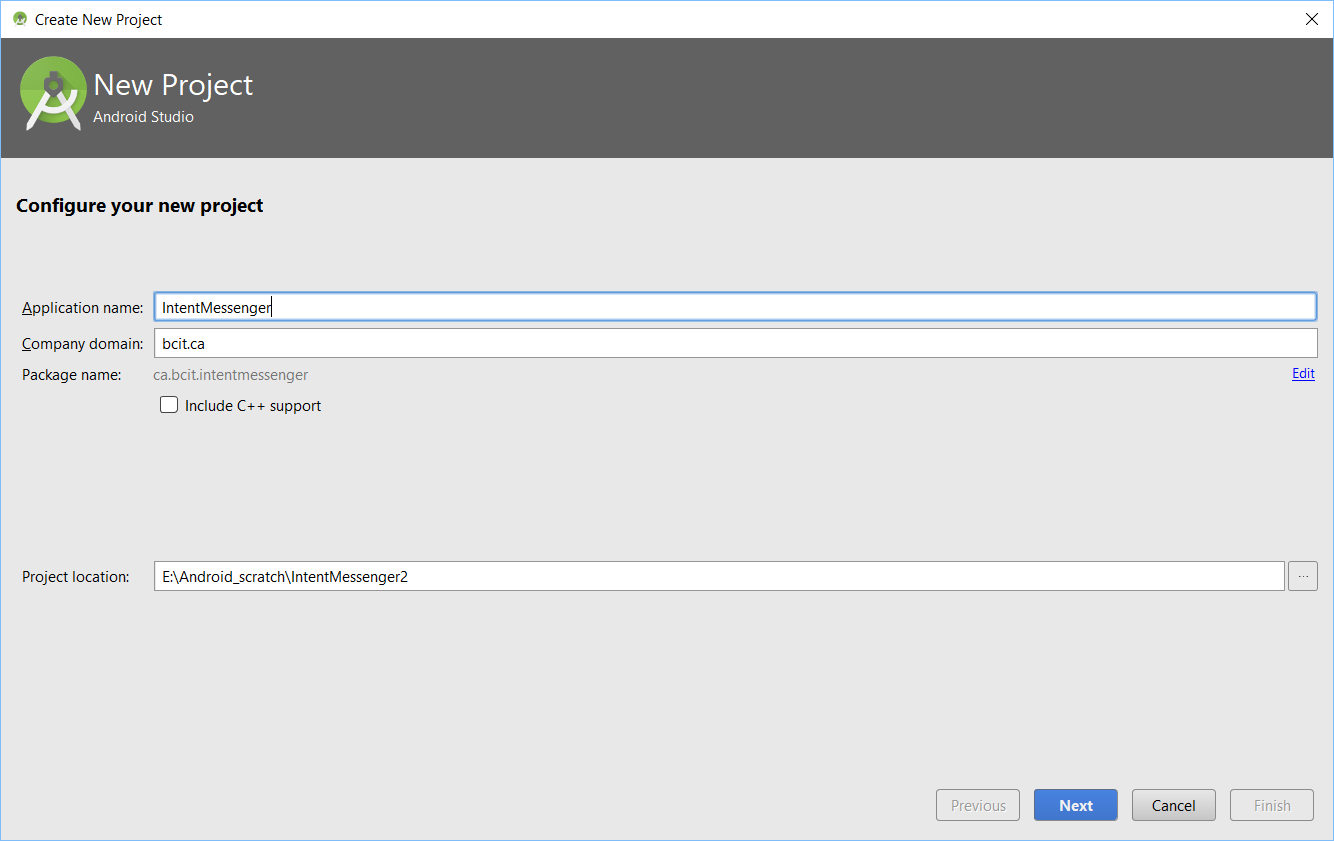
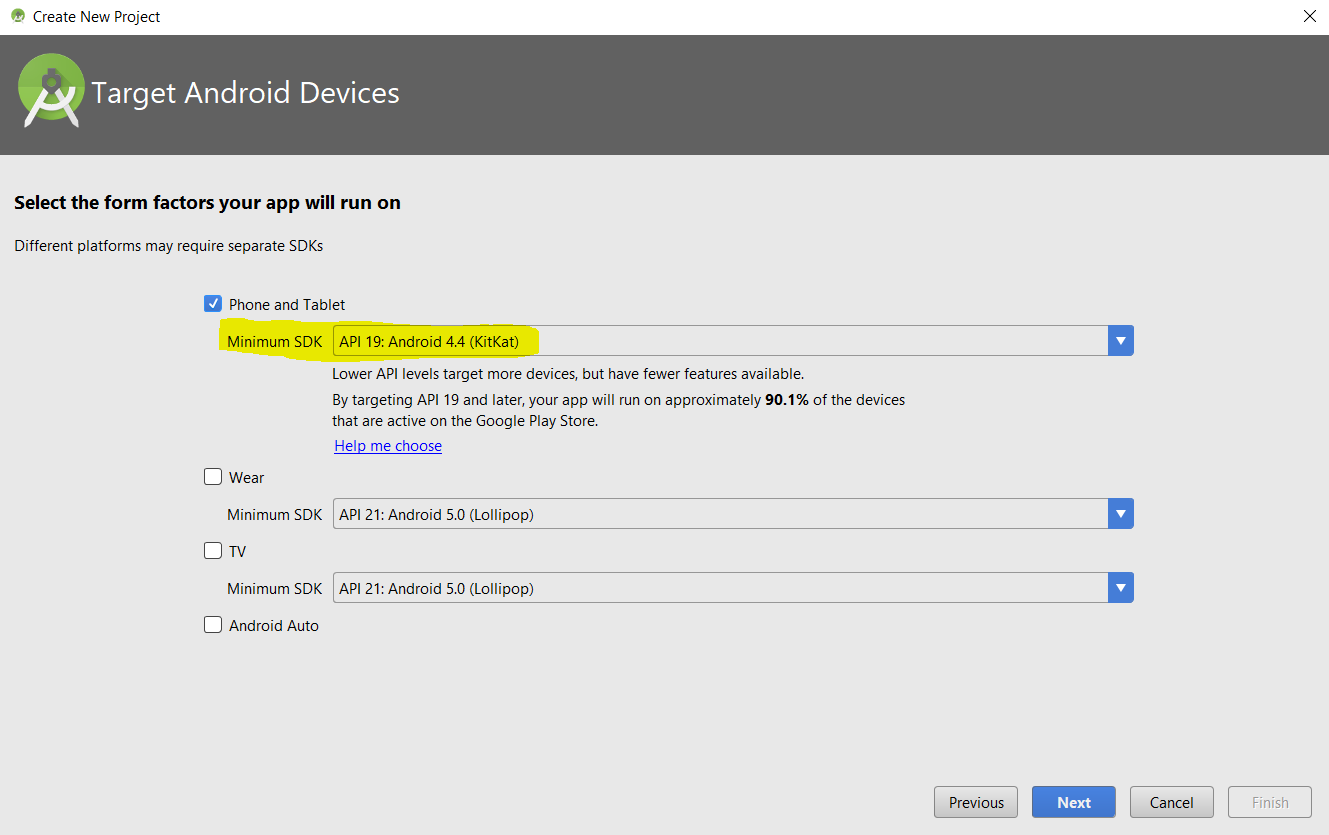
Intro to Android Intents

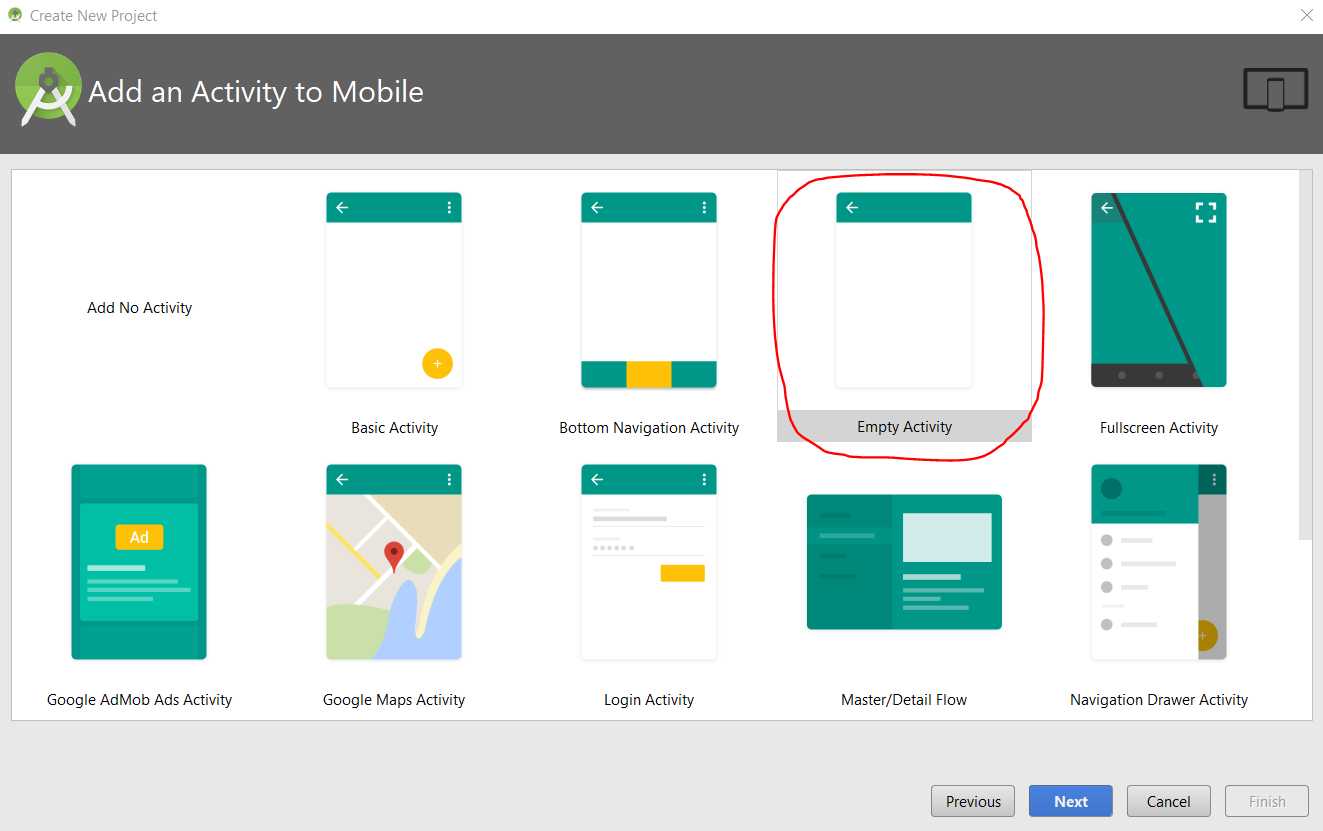
Create an application named “*IntentMessenger*”: *File >> New >> New Project*



Set minimum SDK to API 19:

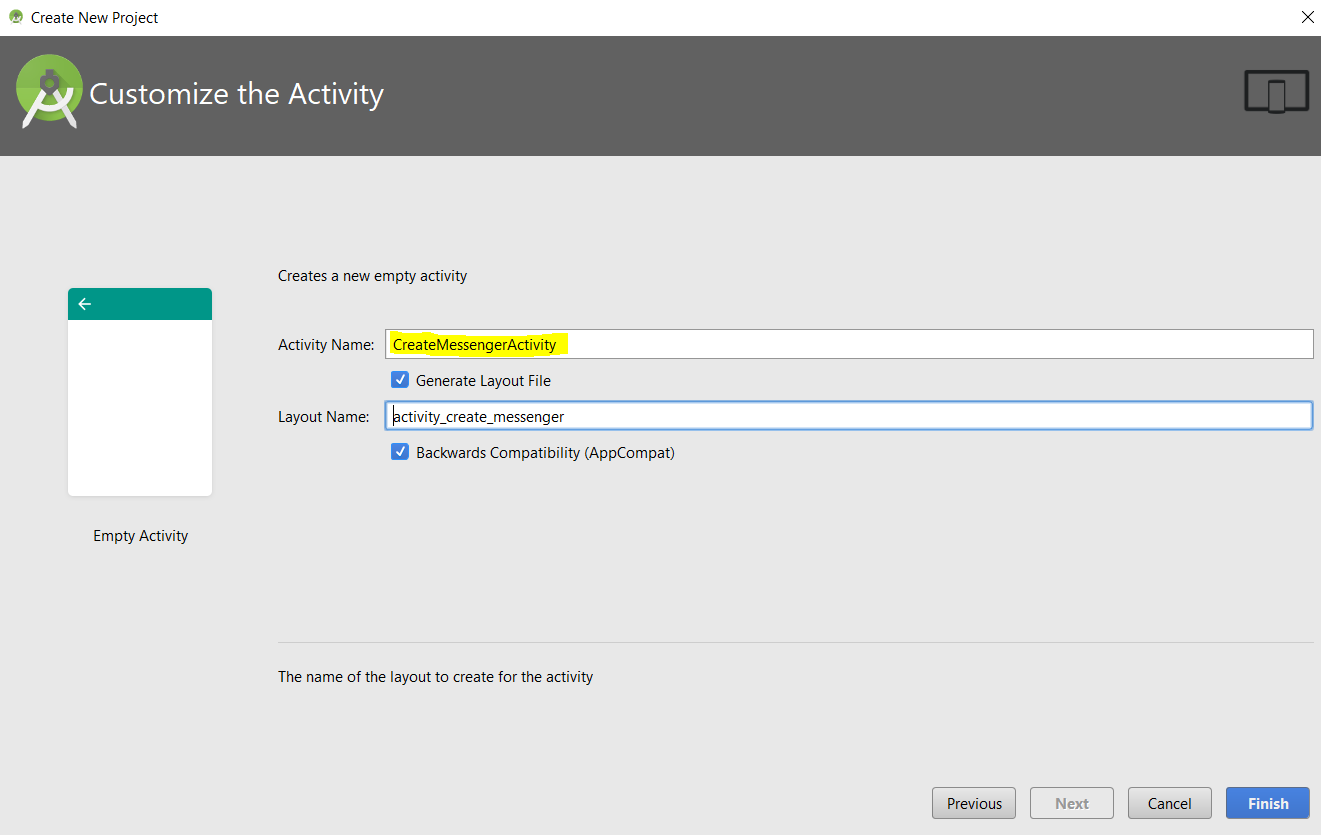


Choose *Empty Activity*:



## Explicit Intent

Change Activity Name to *CreateMessengerActivity*:



Change *activity\_create\_message.xml* to use *RelativeLayout*.

Replace contents of *activity\_create\_message.xml* with:

<Button

android:id="@+id/send"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

01. Button EditText..txt

android:layout\_alignParentLeft="true"

android:layout\_alignParentTop="true"

android:layout\_marginTop="21dp"

android:layout\_marginLeft="36dp"

android:onClick="onSendMessage"

android:text="@string/send" />

<EditText

android:id="@+id/message"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/send"

android:layout\_below="@+id/send"

android:layout\_marginTop="18dp"

android:ems="10"/>

Add the following to *strings.xml*:

<string name="send">Send Message</string>

02. onSendMessage 1.txt

Add following *onSendMessage()* method code to *CreateMessengerActivity.java*:

// call onSendMessage() when button is clicked

public void onSendMessage(View view) {

EditText messageView = (EditText) findViewById(R.id.message);

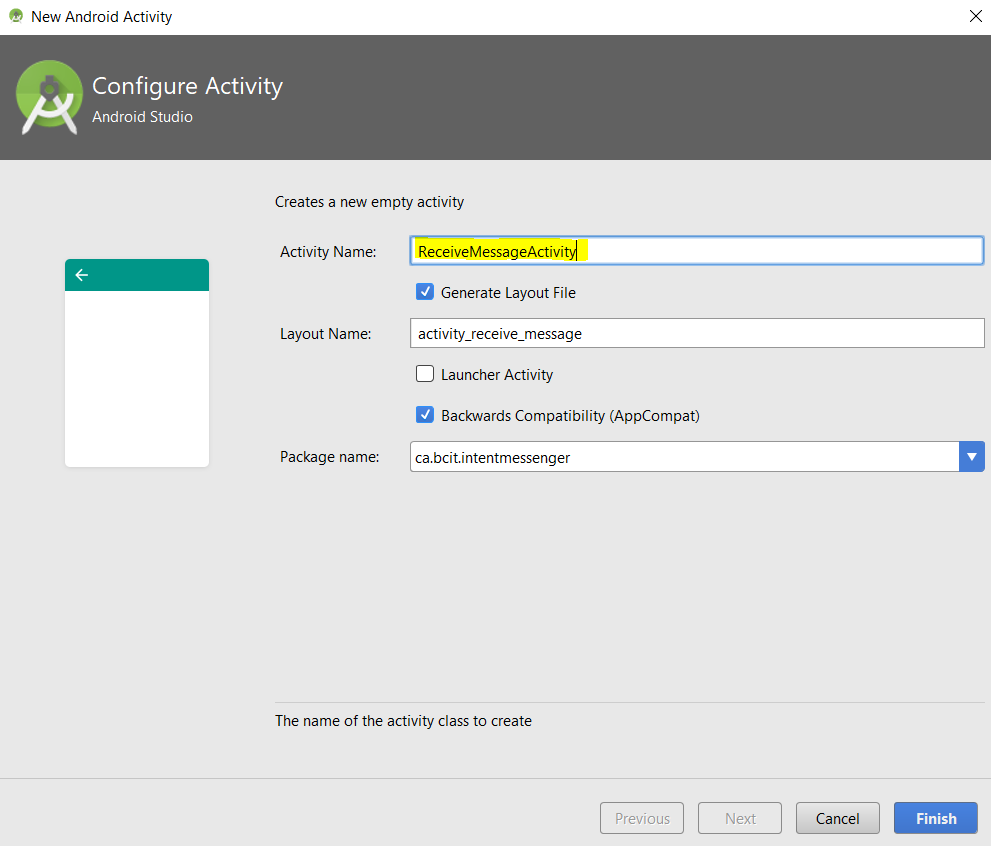
String messageText = messageView.getText().toString();

}

Create a second Activity to receive message:

*File >> New >> Activity >> Empty Activity*

Name the new Activity “*ReceiveMessageActivity*”.



Have a peek at *AndroidManifest.xml*. You will realize that an entry exists for every Activity:

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

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

package="ca.bcit.intentmessenger">

<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=".CreateMessengerActivity">

<intent-filter>

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

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

</intent-filter>

</activity>

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

</application>

</manifest>

Back in CreateMessengerActivity.java, add the following code to onSendMessage() method:

03. onSendMessage 2.txt

// Explicit intent

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

intent.putExtra("msg", messageText);

startActivity(intent);

The new Activity does not have any UI. Add the following TextView to activity\_receive\_message.xml:

<TextView

04. TextView.txt

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

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

Add the following instance variable to ReceiveMessageActivity.java:

05. EXTRA\_MESSAGE.txt

public static final String EXTRA\_MESSAGE = "msg";

Add the following code to the onCreate() method in ReceiveMessageActivity.java:

Intent intent = getIntent();

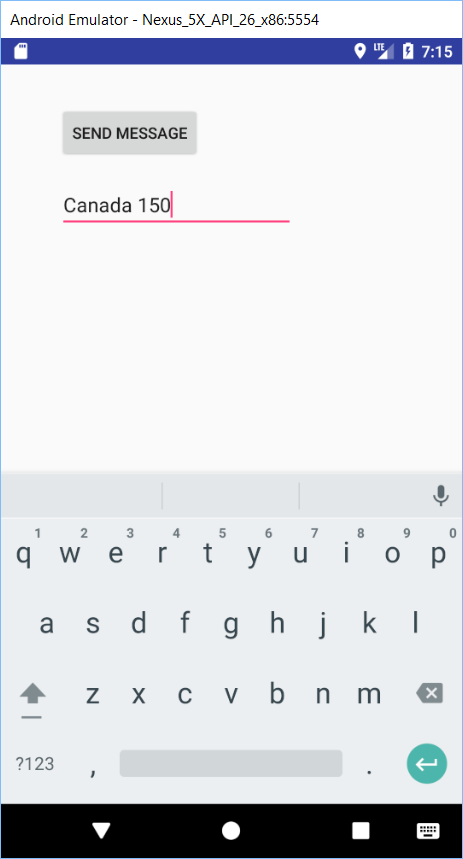
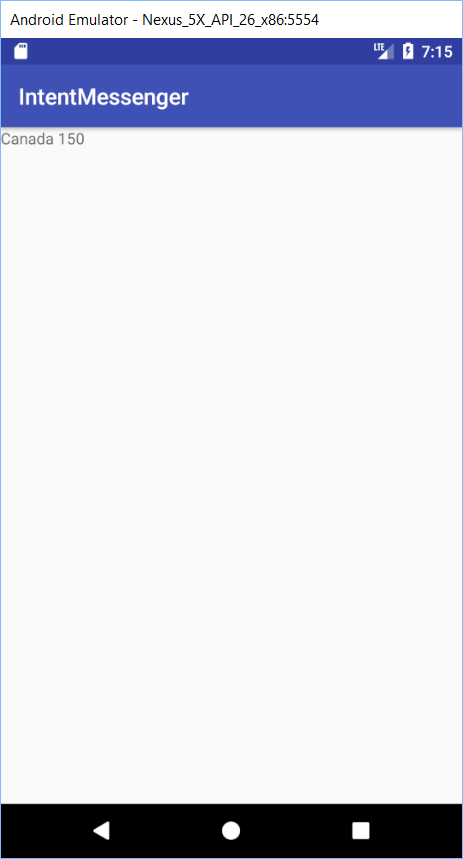
06. Intent.txt

String messageText = intent.getStringExtra(EXTRA\_MESSAGE);

TextView messageView = (TextView) findViewById(R.id.message);

messageView.setText(messageText);

Run your app and see what it does.

To return to the calling activity, simply call the *finish()* method.

Add the following Button control to *activity\_receive\_message.xml*:

<Button

android:layout\_width="wrap\_content"

07. Button1.txt

android:layout\_height="wrap\_content"

android:text="@string/back"

android:onClick="onBackClick"/>

Add the following to strings.xml:

<string name="back">Back</string>

Then, add the following *onBackClick()* event handler method to *ReceiveMessageActivity.java*:

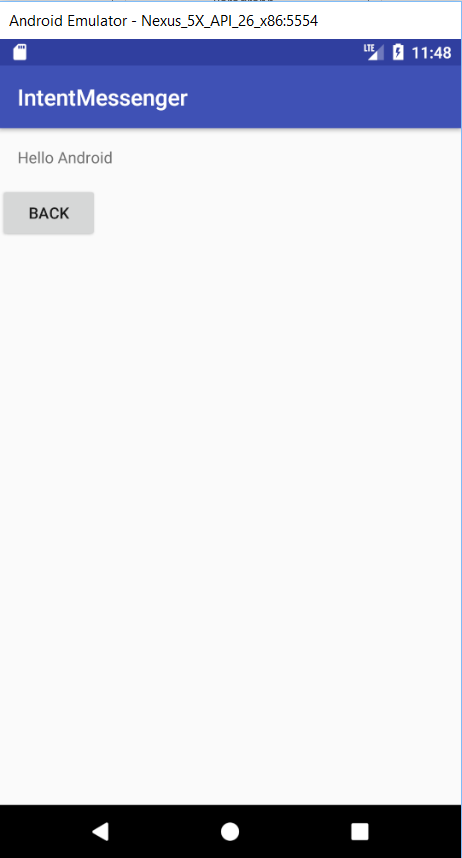
08. onBackClick.txt

public void onBackClick(View v) {

finish();

}

You will have a Back button that returns you back to CreateMessengerActicity.



## Implicit Activities

Instead of sending a message to our own activity, we can send our message to another app like Email, Messenger, Whatsapp, etc. This is called an Implicit Activity.

Add a button to *activity\_create\_messenger.xml* below the *EditText* control as follows:

<Button

09. Button2.txt

android:id="@+id/sendToApp"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="21dp"

android:onClick="onSendMessageToOtherApp"

android:layout\_alignLeft="@+id/message"

android:layout\_below="@+id/message"

android:text="@string/sendToApp" />

Add the following to *strings.xml*:

<string name="sendToApp">Send message to another app</string>

Code the event handler method for our button by adding this method to *CreateMessengerActivity.java*:

public void onSendMessageToOtherApp(View v) {

EditText messageView = (EditText) findViewById(R.id.message);

String messageText = messageView.getText().toString();

10. onSendMessageToOtherApp.txt

Intent i = new Intent(Intent.ACTION\_SEND);

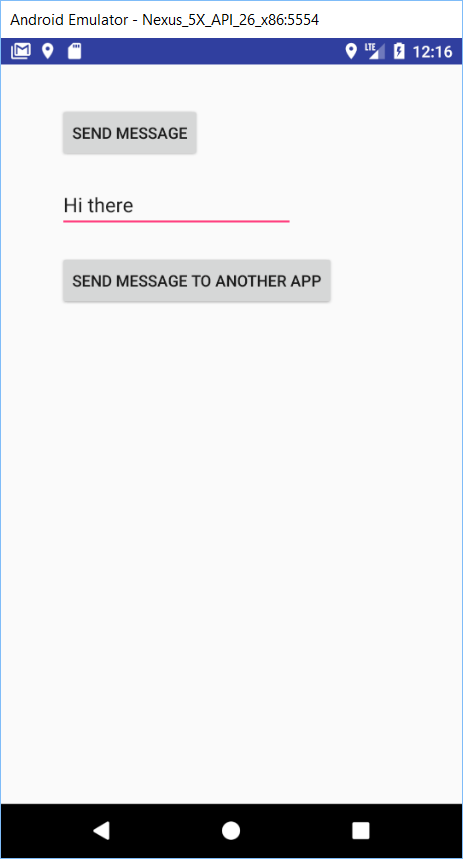
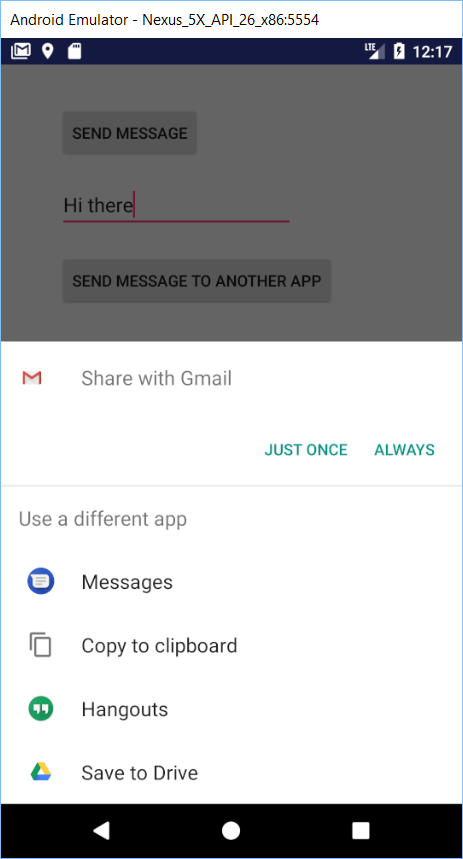
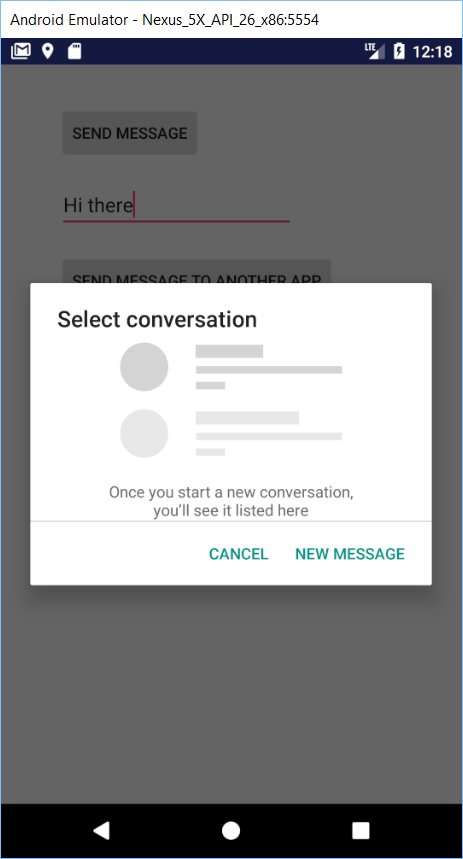
i.setType("text/plain");

i.putExtra(Intent.EXTRA\_TEXT, messageText);

startActivity(i);

}

Run the app and click on the “SEND MESSAGE TO OTHER APP” button. On the next screen select Messages.

Click “NEW MESSAGE”. The messages app is open. After you enter the phone number you will see the message from the main activity:

