# Chapter 6

# Android Telephony

## Telephony Manager

The **android.telephony.TelephonyManager** class provides information about the telephony services such as subscriber id, sim serial number, phone network type etc. Moreover, you can determine the phone state etc.

**Example**: the following example shows how to display information of your telephony services using TelephoneManager class.

**activity\_main.xml**

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

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

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    android:paddingBottom="@dimen/activity\_vertical\_margin"

    android:paddingLeft="@dimen/activity\_horizontal\_margin"

    android:paddingRight="@dimen/activity\_horizontal\_margin"

    android:paddingTop="@dimen/activity\_vertical\_margin"

    tools:context=".MainActivity" >

    <TextView

        android:id="@+id/textView1"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_alignParentLeft="true"

        android:layout\_alignParentTop="true"

        android:layout\_marginLeft="38dp"

        android:layout\_marginTop="30dp"

        android:text="Phone Details:" />

</RelativeLayout>

**MainActivity.java**

import android.os.Bundle;

import android.app.Activity;

import android.content.Context;

import android.telephony.TelephonyManager;

import android.view.Menu;

import android.widget.TextView;

public class MainActivity extends Activity {

   TextView textView1;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

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

        //Get the instance of TelephonyManager

TelephonyManager  tm=(TelephonyManager)getSystemService(Context.TELEPHONY\_SERVICE);

         //Calling the methods of TelephonyManager the returns the information

        String IMEINumber=tm.getIMEINumber();

        String subscriberID=tm.getDeviceId();

        String SIMSerialNumber=tm.getSimSerialNumber();

        String networkCountryISO=tm.getNetworkCountryIso();

        String SIMCountryISO=tm.getSimCountryIso();

        String softwareVersion=tm.getDeviceSoftwareVersion();

        String voiceMailNumber=tm.getVoiceMailNumber();

        //Get the phone type

        String strphoneType="";

        int phoneType=tm.getPhoneType();

        switch (phoneType)

        {

                case (TelephonyManager.PHONE\_TYPE\_CDMA):

                           strphoneType="CDMA";

                               break;

                case (TelephonyManager.PHONE\_TYPE\_GSM):

                           strphoneType="GSM";

                               break;

                case (TelephonyManager.PHONE\_TYPE\_NONE):

                            strphoneType="NONE";

                                break;

         }

        //getting information if phone is in roaming

        boolean isRoaming=tm.isNetworkRoaming();

        String info="Phone Details:\n";

        info+="\n IMEI Number:"+IMEINumber;

        info+="\n SubscriberID:"+subscriberID;

        info+="\n Sim Serial Number:"+SIMSerialNumber;

        info+="\n Network Country ISO:"+networkCountryISO;

        info+="\n SIM Country ISO:"+SIMCountryISO;

        info+="\n Software Version:"+softwareVersion;

        info+="\n Voice Mail Number:"+voiceMailNumber;

        info+="\n Phone Network Type:"+strphoneType;

        info+="\n In Roaming? :"+isRoaming;

        textView1.setText(info);//displaying the information in the textView

    }

}

**AndroidManifest.xml**

You need to provide **READ\_PHONE\_STATE** permission in the AndroidManifest.xml file.

*AndroidManifest.xml*

<manifest ...>

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

          …

</manifest>

## Phone Call

In android, we can easily make a phone call from our android applications by invoking built-in phone calls app using [Intents](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) action (**ACTION\_CALL**). To make a phone call using [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object in android application, we need to write the code like as shown below.

Intent callIntent = new Intent(Intent.ACTION\_CALL);  
callIntent.setData(Uri.parse("tel:" + txtPhone.getText().toString()));  
startActivity(callIntent);

If you observe above code, we are using Intent object **ACTION\_CALL** action to make a phone call based on our requirements.

**Example**

Following is the example of making a phone call by invoking default phone calls app using [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object in android application. Create a new android application using android studio and give names as **PhoneCallExample**.

**activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:orientation="vertical" android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent">  
    <TextView  
        android:id="@+id/fstTxt"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:layout\_marginTop="150dp"  
        android:text="Mobile No"  
        />  
    <EditText  
        android:id="@+id/mblTxt"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:ems="10">  
    </EditText>  
    <Button  
        android:id="@+id/btnCall"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:text="Call" />  
</LinearLayout>

Now open our main activity file **MainActivity.java** from **\src\main\java\com.example.phonecallexample** path and write the code like as shown below

**MainActivity.java**

import android.Manifest;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.net.Uri;  
import android.os.Build;  
import android.support.v4.app.ActivityCompat;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
  
public class MainActivity extends AppCompatActivity {  
    private EditText txtPhone;  
    private Button btn;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity\_main);  
        txtPhone = (EditText)findViewById(R.id.mblTxt);  
        btn = (Button)findViewById(R.id.btnCall);  
        btn.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                callPhoneNumber();  
            }  
        });  
    }  
    @Override  
    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults)  
    {  
        if(requestCode == 101)  
        {  
            if(grantResults[0] == PackageManager.PERMISSION\_GRANTED)  
            {                callPhoneNumber();            }  
        }  
    }  
  
    public void callPhoneNumber()  
    {  
        try  
        {  
            if(Build.VERSION.SDK\_INT > 22)  
            {  
                if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CALL\_PHONE) != PackageManager.PERMISSION\_GRANTED) {  
                    ActivityCompat.requestPermissions(MainActivity.this, new String[]{Manifest.permission.CALL\_PHONE}, 101);  
                    return;  
                }  
  
                Intent callIntent = new Intent(Intent.ACTION\_CALL);  
                callIntent.setData(Uri.parse("tel:" + txtPhone.getText().toString()));  
                startActivity(callIntent);  
  
            }  
            else {  
                Intent callIntent = new Intent(Intent.ACTION\_CALL);  
                callIntent.setData(Uri.parse("tel:" + txtPhone.getText().toString()));  
                startActivity(callIntent);  
            }  
        }  
        catch (Exception ex)  
        {  
            ex.printStackTrace();  
        }  
    }  
}

If you observe above code we are adding **Runtime** permissions to make sure our application work in both old / latest android OS versions and we used [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) action (**ACTION\_CALL**) to make phone call on button click using default phone calls app. As discussed, we need to add a **CALL\_PHONE** permission in our android manifest.

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

</manifest>

## Send SMS

In android, we can send SMS from our android application in two ways either by using **SMSManager** api or [Intents](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) based on our requirements. If we use **SMSManager** api, it will directly send SMS from our application. In case if we use [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) with proper action (**ACTION\_VIEW**), it will invoke built-in SMS app to send SMS from our application.

**Sending SMS using SMSManager API**

In android, to send SMS using SMSManager API we need to write the code like as shown below.

SmsManager smgr = SmsManager.getDefault();  
smgr.sendTextMessage(MobileNumber,null,Message,null,null);

**SMSManager** API required **SEND\_SMS** permission in our android manifest to send SMS. Following is the code snippet to set **SEND\_SMS** permissions in manifest file.

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

**Android Send SMS using Intent**

To send SMS using [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object, we need to write the code like as shown below.

Intent sInt = new Intent(Intent.ACTION\_VIEW);  
sInt.putExtra("address", new String[]{txtMobile.getText().toString()});  
sInt.putExtra("sms\_body",txtMessage.getText().toString());  
sInt.setType("vnd.android-dir/mms-sms");

Even for [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html), it required a **SEND\_SMS** permission in our android manifest to send SMS. Following is the code snippet to set **SEND\_SMS** permissions in manifest file.

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

Now we will see how to send SMS in android application using **SMSManager** API with examples.

**Example: The f**ollowing example shows how to send sms from our android app using **SMSManager** api.

 Create a new android application using android studio and give names as **SendSMSExample**.

**activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:orientation="vertical" android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent">  
    <TextView  
        android:id="@+id/fstTxt"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:layout\_marginTop="150dp"  
        android:text="Mobile No" />  
    <EditText  
        android:id="@+id/mblTxt"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:ems="10"/>

    <TextView  
        android:id="@+id/secTxt"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:text="Message"  
        android:layout\_marginLeft="100dp" />  
    <EditText  
        android:id="@+id/msgTxt"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:ems="10" />  
    <Button  
        android:id="@+id/btnSend"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:layout\_marginLeft="100dp"  
        android:text="Send SMS" />  
</LinearLayout>

Now open our main activity file **MainActivity.java** from **\src\main\java\com.example.sendsmsexample** path and write the code like as shown below

**MainActivity.java**

import android.content.Intent;  
import android.net.Uri;  
import android.provider.Telephony;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.telephony.SmsManager;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Toast;  
  
public class MainActivity extends AppCompatActivity {  
  
    private EditText txtMobile;  
    private EditText txtMessage;  
    private Button btnSms;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity\_main);  
        txtMobile = (EditText)findViewById(R.id.mblTxt);  
        txtMessage = (EditText)findViewById(R.id.msgTxt);  
        btnSms = (Button)findViewById(R.id.btnSend);  
        btnSms.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                try{  
                    SmsManager smgr = SmsManager.getDefault();  
                    smgr.sendTextMessage(txtMobile.getText().toString(),null,txtMessage.getText().toString(),null,null);  
                    Toast.makeText(MainActivity.this, "SMS Sent Successfully", Toast.LENGTH\_SHORT).show();  
                }  
                catch (Exception e){

Toast.makeText(MainActivity.this, "SMS Failed to Send, Please try again", Toast.LENGTH\_SHORT).show();  
                }  
            }  
        });  
    }  
}

Do not forget to add a **SEND\_SMS** permission in our android manifest.

<manifest …>  
    <**uses-permission android:name="android.permission.SEND\_SMS"**/>  
<**uses-permission android:name="android.permission.RECEIVE\_SMS"**/>

…

</manifest>

if we want to use [Intents](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) to send SMS replace button click code like as shown below.

btnSms.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        try{  
            Intent i = new Intent(Intent.ACTION\_VIEW);  
            i.setData(Uri.parse("smsto:"));  
            i.setType("vnd.android-dir/mms-sms");  
            i.putExtra("address", new String(txtMobile.getText().toString()));  
            i.putExtra("sms\_body",txtMessage.getText().toString());  
            startActivity(Intent.createChooser(i, "Send sms via:"));  
        }  
        catch(Exception e){  
Toast.makeText(MainActivity.this, "SMS Failed to Send, Please try again", Toast.LENGTH\_SHORT).show();  
        }  
    }  
});

## Send Email

In android, we can easily send an email from our android application using existing email clients such as **GMAIL**, **Outlook**, etc. instead of building an email client from scratch. The [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object in android with proper action (**ACTION\_SEND**) and data will help us to launch the available email clients to send an email in our application. To send an email using [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object in android application, we need to write the code like as shown below.

Intent it = new Intent(Intent.ACTION\_SEND);  
it.putExtra(Intent.EXTRA\_EMAIL, new String[]{"abc@gmail.com "});  
it.putExtra(Intent.EXTRA\_SUBJECT, "test");  
it.putExtra(Intent.EXTRA\_TEXT, "Hello this is to inform you that….");  
it.setType("message/rfc822");

If you observe above code we used multiple components to send email, those are

* **it** - Our local [implicit intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-implicit-intents-with-examples.html)
* **ACTION\_SEND** - It’s an [activity](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-activity-lifecycle.html) action which specifies that we are sending some data.
* **putExtra** - we use this **putExtra()** method to add extra information to our Intent. Here we can add following things.
* EXTRA\_EMAIL - It’s an array of email addresses
* EXTRA\_SUBJECT - The subject of the email that we want to send
* EXTRA\_TEXT - The body of the email

The android [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object is having different options such as EXTRA\_CC, EXTRA\_BCC, EXTRA\_HTML\_TEXT, EXTRA\_STREAM, etc. to add different options for email client.

* **setType** - We use this property to set the MIME type of data that we want to send. Here we used “**message/rfc822**” and other MIME types are “**text/plain**” and “**image/jpg**”.

 Now we will see how to send an email in android application using [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) object with examples.

**Example:** Following is the example of sending an email with existing email clients using [Intent](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intents-implicit-explicit.html) in android application. Create a new android application using android studio and give names as **SendMailExample**. In case if you are not aware of creating an app in android studio check this article [Android Hello World App](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-hello-world-app-example.html).

**activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"  
    android:paddingLeft="20dp"  
    android:paddingRight="20dp"  
    android:orientation="vertical" >  
    <EditText  
        android:id="@+id/txtTo"  
        android:layout\_width="match\_parent"  
        android:layout\_height="wrap\_content"  
        android:hint="To"/>  
    <EditText  
        android:id="@+id/txtSub"  
        android:layout\_width="match\_parent"  
        android:layout\_height="wrap\_content"  
        android:hint="Subject"/>  
    <EditText  
        android:id="@+id/txtMsg"  
        android:layout\_width="match\_parent"  
        android:layout\_height="0dp"  
        android:layout\_weight="1"  
        android:gravity="top"  
        android:hint="Message"/>  
    <Button  
        android:layout\_width="100dp"  
        android:layout\_height="wrap\_content"  
        android:layout\_gravity="right"  
        android:text="Send"  
        android:id="@+id/btnSend"/>  
</LinearLayout>

Now open our main [activity](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-activity-lifecycle.html) file **MainActivity.java** from **\src\main\java\com.example.sendmailexample** path and write the code like as shown below

**MainActivity.java**

import android.content.Intent;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
  
public class MainActivity extends AppCompatActivity {  
  
    private EditText eTo;  
    private EditText eSubject;  
    private EditText eMsg;  
    private Button btn;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity\_main);  
        eTo = (EditText)findViewById(R.id.txtTo);  
        eSubject = (EditText)findViewById(R.id.txtSub);  
        eMsg = (EditText)findViewById(R.id.txtMsg);  
        btn = (Button)findViewById(R.id.btnSend);  
        btn.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                Intent it = new Intent(Intent.ACTION\_SEND);  
                it.putExtra(Intent.EXTRA\_EMAIL, new String[]{eTo.getText().toString()});  
                it.putExtra(Intent.EXTRA\_SUBJECT,eSubject.getText().toString());  
                it.putExtra(Intent.EXTRA\_TEXT,eMsg.getText());  
                it.setType("message/rfc822");  
                startActivity(Intent.createChooser(it,"Choose Mail App"));  
            }  
        });  
    }  
}

We need to add **MIME type** in our android manifest file for that open android manifest file (**AndroidManifest.xml**) and write the code like as shown below

<manifest …>  
    <application  
        …

<activity android:name=".MainActivity">  
            <intent-filter>  
                <action android:name="android.intent.action.MAIN" />  
                <category android:name="android.intent.category.LAUNCHER" />  
                <action android:name="android.intent.action.SEND"/>  
                <category android:name="android.intent.category.DEFAULT"/>  
                <data android:mimeType="message/rfc822"/>  
            </intent-filter>  
        </activity>  
    </application>  
</manifest>

If you observe above **AndroidManifest.xml** file we added following extra fields of [Intent filters](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-intent-filters-with-examples.html).

* **action** - we use this property to define that the activity can perform **SEND** action.
* **category** - we included the **DEFAULT** category for this activity to be able to receive implicit intents.
* **data** - the type of data the [activity](file:///D:\DMU\2011\MobileProgramming\References\meti%20tutlane%20android\www.tutlane.com\tutorial\android\android-activity-lifecycle.html) can send.