Chapter 6

Android Telephony

Telephone Manager

- **android.telephony.TelephonyManager** class provides information about the telephony services such as subscriber id, sim serial number, phone network type etc.

```
//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 phoneID=tm.getLine1Number();
    String SIMSerialNumber=tm.getSimSerialNumber();
    String networkCountryISO=tm.getNetworkCountryIso();
    String SIMCountryISO=tm.getSimCountryIso();
    String softwareVersion=tm.getDeviceSoftwareVersion();
    String voiceMailNumber=tm.getVoiceMailNumber();
```

- Give READ_PHONE_STATE permission in android manifest file

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

Make a phone call

- making a phone call from our android applications is done easily by invoking built-in phone calls app using Intents action (**ACTION_CALL**).

```
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:" + txtPhone.getText().toString()));
startActivity(callIntent);
```

- we need to add a **CALL_PHONE** permission in our android manifest.

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

Sending SMS

- we can send SMS from our android application in two ways either by
  - using **SMSManager** api or

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

  - Intents **ACTION_VIEW** action

```
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");
```

- Make sure to allow **SEND_SMS** permission android manifest file

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

Sending Email

- we can easily send an email from our android application using **existing email clients** such as **GMAIL**, **Outlook**, etc
- The **Intent** 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

```
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");
```

- **it** - Our local implicit intent
- **ACTION_SEND** - It's an activity action which specifies that we are sending some data.
- **putExtra** - add extra information to our Intent. Example: recipient, subject and message
  - 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
- **setType** - set the MIME type of data to be sent. Example"**message/rfc822**" and other MIME types are "**text/plain**" and "**image/jpg**".

- We need to add **MIME type** and **Permission** in our android manifest file 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>
```

- **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 can send.