12:13 PM 73

Android Share App Data (ACTION_SEND)

Android uses **ACTION_SEND** event of **android.content.Intent** class to send data from one activity to another and from current activity to outside the application. Intent class needs to specify the data and its type which is to be share.

Most commonly, ACTION_SEND action sends URL of build-in Browser app. While sharing the data, Intent call *createChooser()* method which takes Intent object and specify the title of the chooser dialog. **Intent.createChooser()** method allows to display the chooser.

Example of ACTION_SEND

In this example, we are going to share plain text which is a URL of browser.

activity_main.xml

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.c
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

12:13 PM 73

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.cc
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_marg
    android:paddingLeft="@dimen/activity_horizontal_margi
    android:paddingRight="@dimen/activity_horizontal_marg
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.test.shareapp.MainActivity"

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        android:id="@+id/textView" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Share App"
        android:layout_height="wrap_content"

        android:text="Hello World!"

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


    <Button

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:text="Share App"

        android:id="@+id/button"

        android:layout_marginBottom="95dp"

        android:layout_alignParentBottom="true"

        android:layout_centerHorizontal="true" />


</RelativeLayout>
```

Activity class

12:14 PM

https://www.javatpoint.com/andrc

File: MainActivity.java

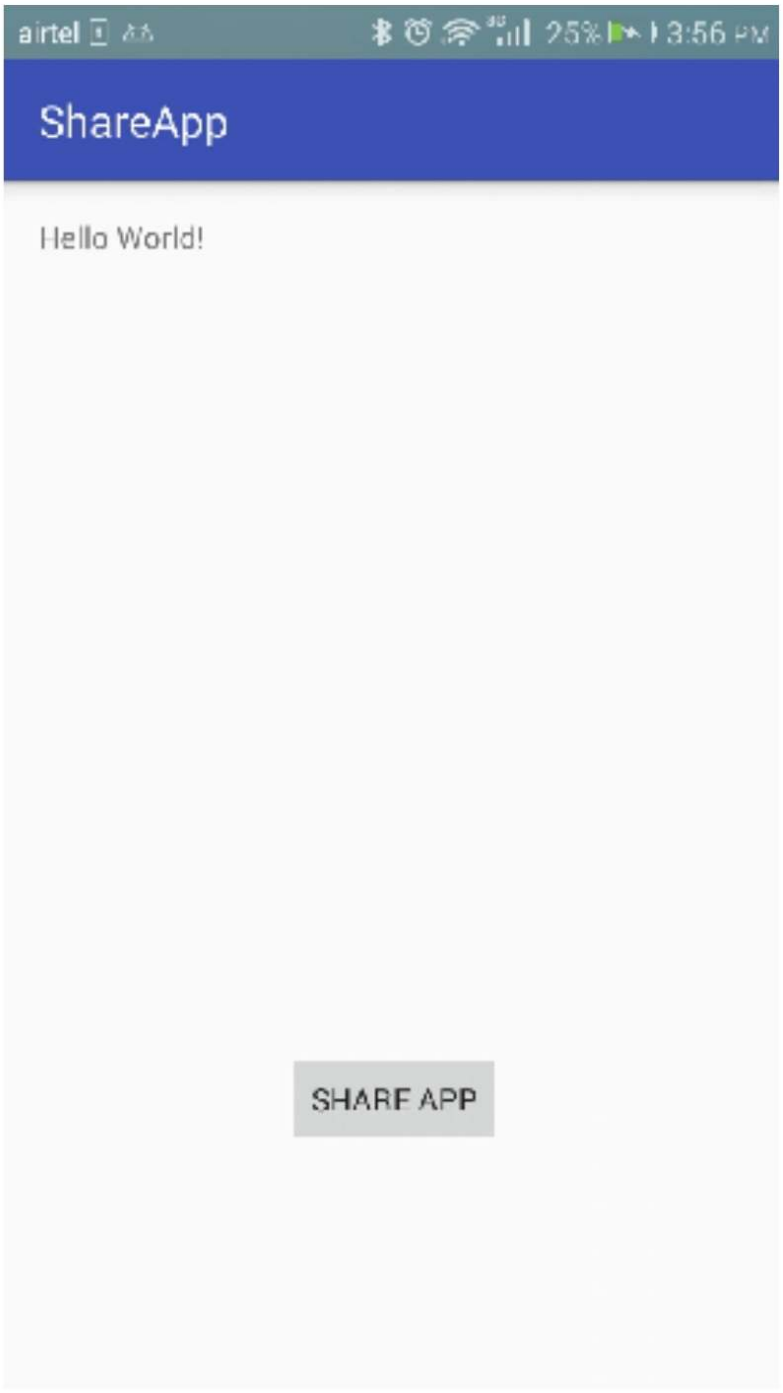
```
package com.example.test.shareapp;


import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;


public class MainActivity extends AppCompatActivity {

Button sharebutton;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);


        sharebutton=(Button)findViewById(R.id.button);

        sharebutton.setOnClickListener(new View.OnClickListe

            @Override

            public void onClick(View v) {
```

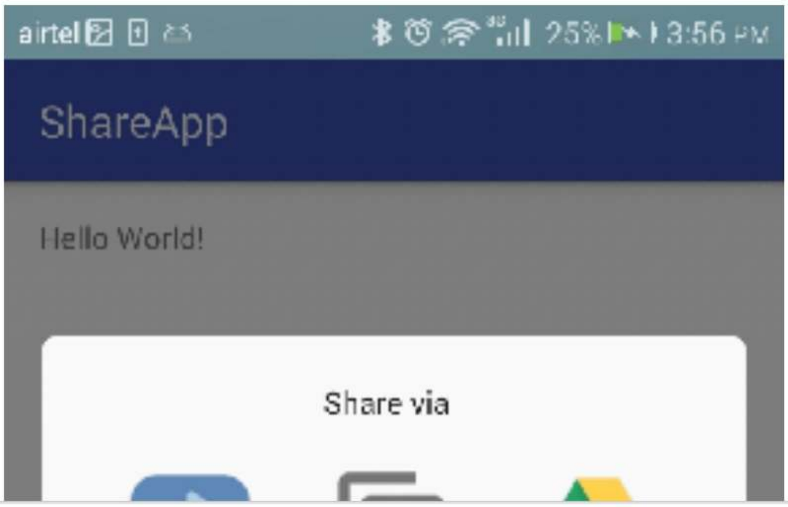
12:14 PM

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


        sharebutton=(Button)findViewById(R.id.button);
        sharebutton.setOnClickListener(new View.OnClickListe
            @Override
            public void onClick(View v) {
                Intent shareIntent =   new Intent(android.content.Ir
                shareIntent.setType("text/plain");
                shareIntent.putExtra(Intent.EXTRA_SUBJECT,"Inse
                String app_url = " https://play.google.com/store/ap
id=my.example.javatpoint";
                shareIntent.putExtra(android.content.Intent.EXTRA
                startActivity(Intent.createChooser(shareIntent, "Sh
            }
        });
    }
}
```

Output

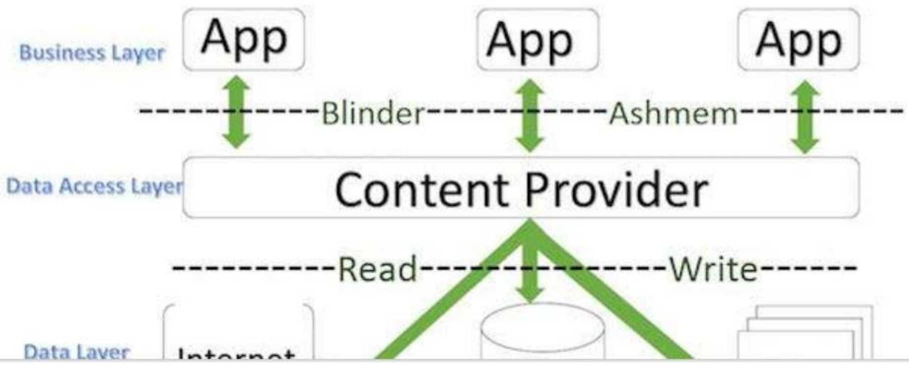
ADVERTISEMENT


airtel
25% 3:56 PM
ShareApp
Hello World!
SHARE APP


airtel
25% 3:56 PM
ShareApp
Hello World!
Share via

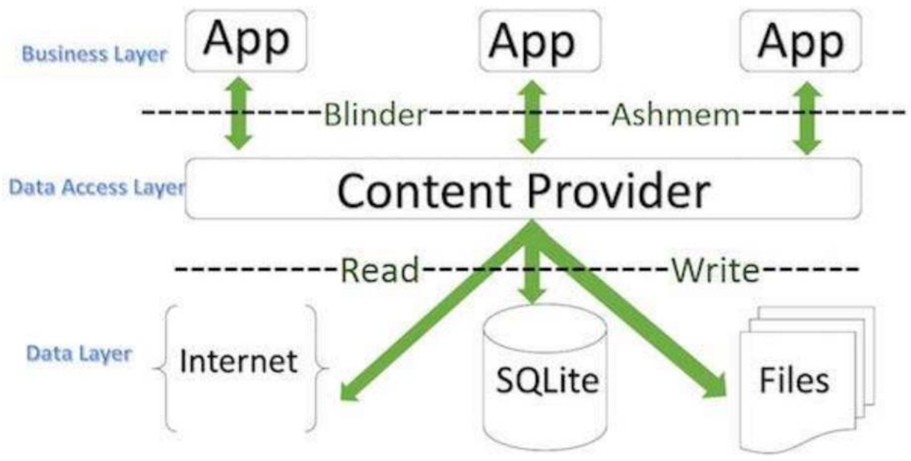
Android - Content Providers

A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolver class. A content provider can use different ways to store its data and the data can be stored in a database, in files, or even over a network.



12:16 PM

Search tutorials, courses ar



ContentProvider

sometimes it is required to share data across applications. This is where content providers become very useful.

Content providers let you centralize content in one place and have many different applications access it as needed. A content provider behaves very much like a database where you can query it, edit its content, as well as add or delete content using insert(), update(), delete(), and query() methods. In most cases this data is stored in an **SQlite** database.

A content provider is implemented as a subclass of **ContentProvider** class and must implement a standard set of APIs that enable

12:16 PM

Search tutorials, courses ar

Content providers let you centralize content in one place and have many different applications access it as needed. A content provider behaves very much like a database where you can query it, edit its content, as well as add or delete content using insert(), update(), delete(), and query() methods. In most cases this data is stored in an **SQlite** database.

A content provider is implemented as a subclass of **ContentProvider** class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class My Application exten
}
```

Content URIs

To query a content provider, you specify the query string in the form of a URI which has following format –

```
<prefix>://<authority>/<data_type
```

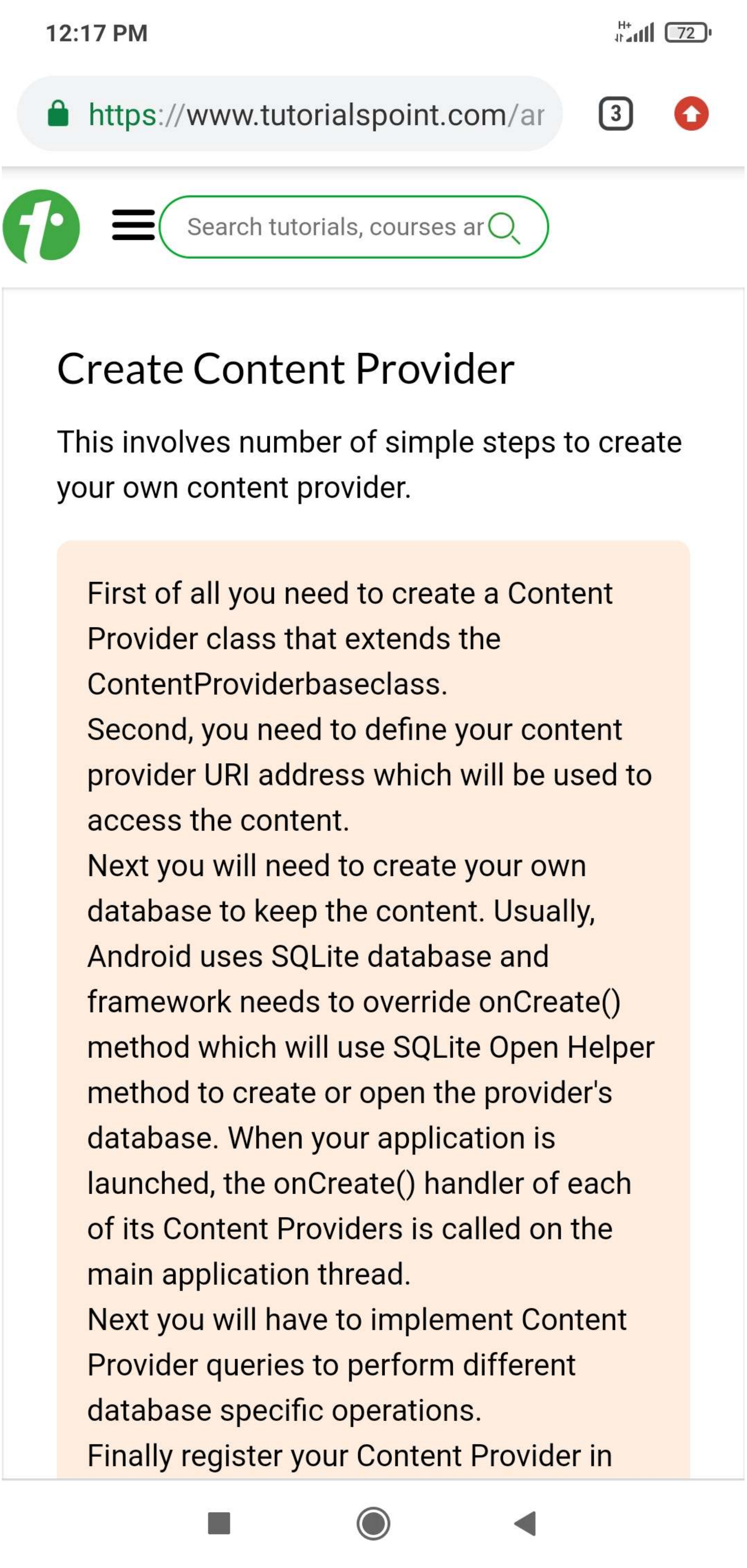
Here is the detail of various parts of the URI –

Sr.No	Part & Description
1	**prefix** This is always set to content://
2	**authority** This specifies the name of the content provider, for example contacts, browser etc. For third-party content providers, this could be the fully qualified name, such as com.tutorialspoint.statusprovider
3	**data_type** This indicates the type of data that this particular provider provides. For example, if you are getting all the contacts from the Contacts content provider, then the data path would be people and URI would look like thiscontent://contacts/people
	id

12:17 PM

Search tutorials, courses ar

3	**data_type** This indicates the type of data that this particular provider provides. For example, if you are getting all the contacts from the Contacts content provider, then the data path would be people and URI would look like thiscontent://contacts/people
4	**id** This specifies the specific record requested. For example, if you are looking for contact number 5 in the Contacts content provider then URI would look like this content://contacts/people/5.



12:17 PM

https://www.tutorialspoint.com/ar

Search tutorials, courses ar

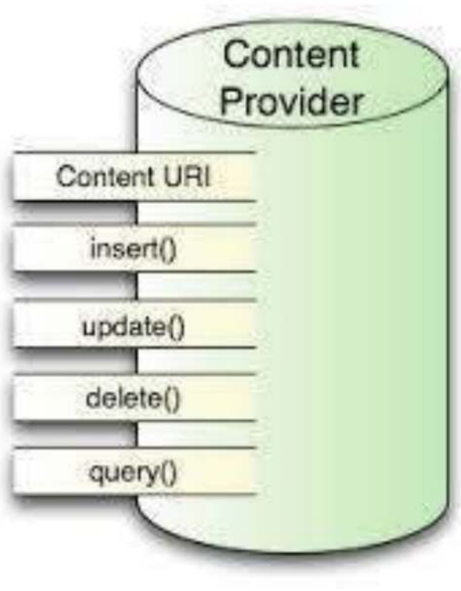
Create Content Provider

This involves number of simple steps to create your own content provider.

First of all you need to create a Content Provider class that extends the ContentProviderbaseclass.
Second, you need to define your content provider URI address which will be used to access the content.
Next you will need to create your own database to keep the content. Usually, Android uses SQLite database and framework needs to override onCreate() method which will use SQLite Open Helper method to create or open the provider's database. When your application is launched, the onCreate() handler of each of its Content Providers is called on the main application thread.
Next you will have to implement Content Provider queries to perform different database specific operations.
Finally register your Content Provider in

launched, the onCreate() handler of each of its Content Providers is called on the main application thread.
Next you will have to implement Content Provider queries to perform different database specific operations.
Finally register your Content Provider in your activity file using <provider> tag.

Here is the list of methods which you need to override in Content Provider class to have your Content Provider working –



ContentProvider

onCreate() This method is called when the provider is started.

ContentProvider

onCreate() This method is called when the provider is started.
query() This method receives a request from a client. The result is returned as a Cursor object.
insert() This method inserts a new record into the content provider.
delete() This method deletes an existing record from the content provider.
update() This method updates an existing record from the content provider.
getType() This method returns the MIME type of the data at the given URI.

Example

This example will explain you how to create your own ContentProvider. So let's follow the following steps to similar to what we followed while creating Hello World Example−

Step	Description
1	You will use Android StudioIDE to create an Android application and name it as My Application under a package com.example.MyApplication, with blank Activity.
2	Modify main activity file MainActivity.java to add two new methods onClickAddName() and onClickRetrieveStudents().
3	Create a new java file called StudentsProvider.java under the package com.example.MyApplication to define your actual provider and associated methods.
4	Register your content provider in your AndroidManifest.xml file using <provider.../> tag
5	Modify the default content of res/layout/activity_main.xml file to include a small GUI to add students records.

3	Create a new java file called StudentsProvider.java under the package com.example.MyApplication to define your actual provider and associated methods.
4	Register your content provider in your AndroidManifest.xml file using <provider.../> tag
5	Modify the default content of res/layout/activity_main.xml file to include a small GUI to add students records.
6	No need to change string.xml.Android studio take care of string.xml file.
7	Run the application to launch Android emulator and verify the result of the changes done in the application.

Following is the content of the modified main activity file **src/com.example.MyApplication/MainActivity.java** This file can include each of the fundamental life cycle methods. We have added two new methods onClickAddName() and onClickRetrieveStudents() to handle user interaction with the application.

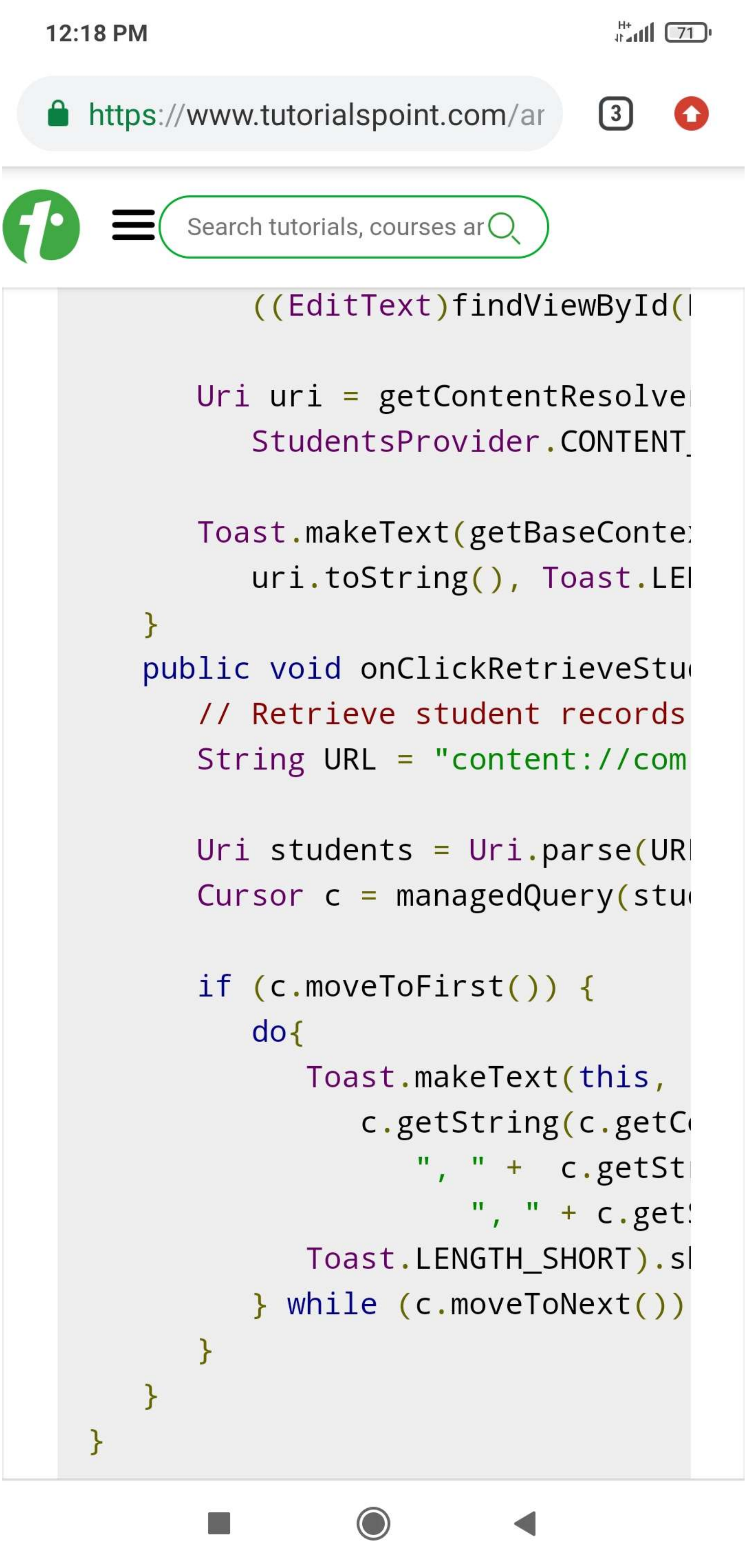
```
package com.example.MyApplication

import android.net.Uri;
```

12:18 PM

Search tutorials, courses ar

interaction with the application.

```
package com.example.MyApplication

import android.net.Uri;
import android.os.Bundle;
import android.app.Activity;

import android.content.ContentVal
import android.content.CursorLoad

import android.database.Cursor;

import android.view.Menu;
import android.view.View;

import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends

   @Override
   protected void onCreate(Bundle
      super.onCreate(savedInstanc
      setContentView(R.layout.act
   }
   public void onClickAddName(Vie
      // Add a new student record
      ContentValues values = new
```



12:18 PM

https://www.tutorialspoint.com/ar

Search tutorials, courses ar

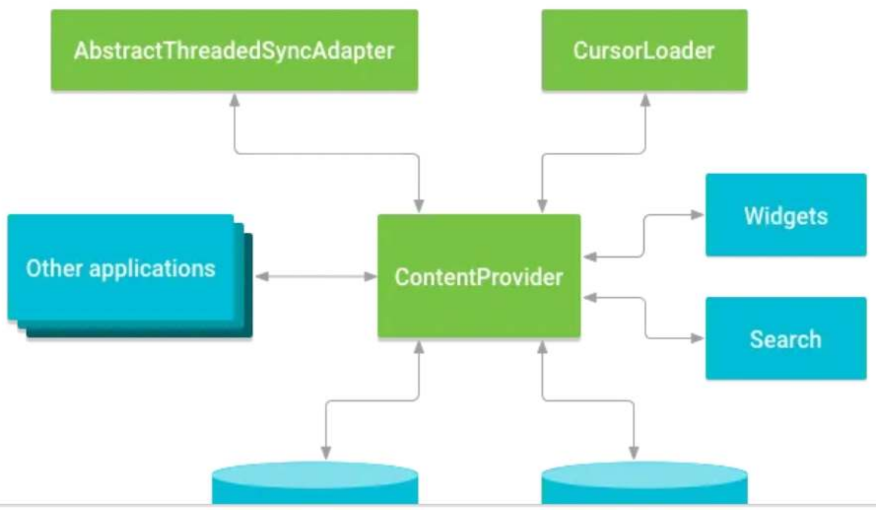
```
        ((EditText)findViewById(

      Uri uri = getContentResolve
        StudentsProvider.CONTENT_

      Toast.makeText(getBaseConte
        uri.toString(), Toast.LE
   }
   public void onClickRetrieveStu
      // Retrieve student records
      String URL = "content://com

      Uri students = Uri.parse(UR
      Cursor c = managedQuery(stu

      if (c.moveToFirst()) {
         do{
            Toast.makeText(this,
               c.getString(c.getC
                  ", " +  c.getSt
                     ", " + c.get
            Toast.LENGTH_SHORT).s
         } while (c.moveToNext())
      }
   }
}
```

data of other applications and secondly, we can create a content provider that gives access or share our application data to other application.

Ways to Use Content Provider

a. You want to expose your application data to widget.
b. You want to implement custom search suggestion in your application
c. You want to copy and paste a complex data or file from your application to other application.

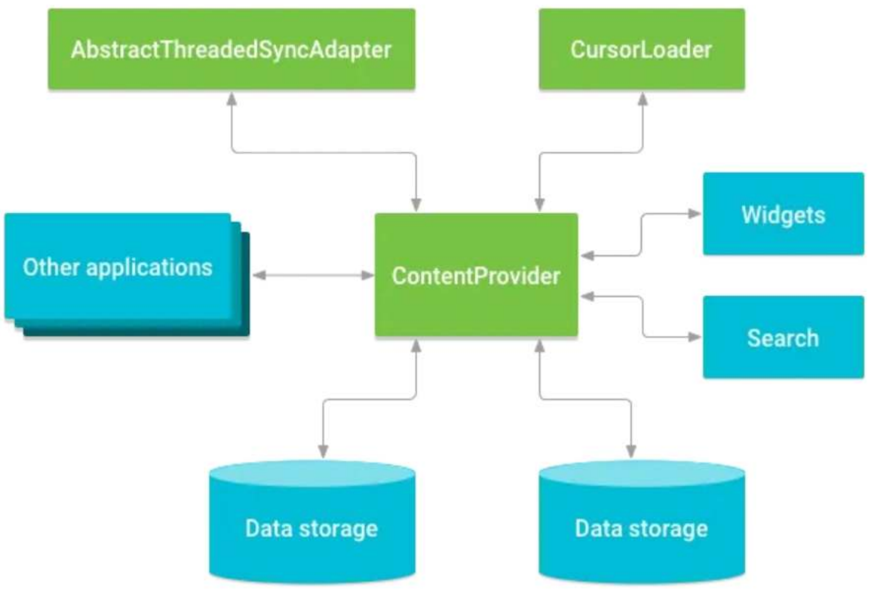
Content provider provides a secured access to structured data like SQLite relational databases and unstructured data like audio, video, images and personal contacts.



Ways to Use Content Provider

a. You want to expose your application data to widget.
b. You want to implement custom search suggestion in your application
c. You want to copy and paste a complex data or file from your application to other application.

Content provider provides a secured access to structured data like SQLite relational databases and unstructured data like audio, video, images and personal contacts.



Relationship between content provider and other components.

Accessing a Provider

When we need to access a content provider, we make use of **ContentResolver** object within your application context. The **contentResolver** communicates with the provider, an instance of the class that implements **contentProvider**. The provider object receives data request from the client and perform the request action on behalf of the client (which is the ContentResolver) and deliver the results back to the client.

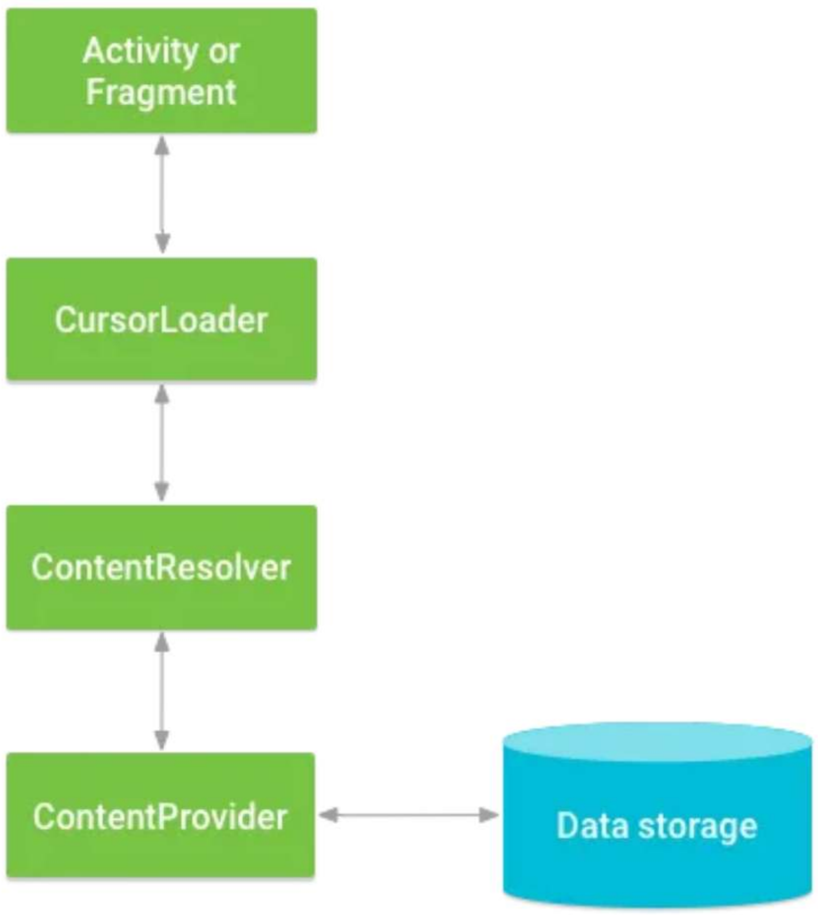
This object has methods that call identically-named methods in the provider object, an instance of one of the concrete subclasses of `ContentProvider`. The `ContentResolver` methods provide the basic "CRUD" (create, retrieve, update, and delete) functions of persistent storage.

Pattern for Accessing Content Provider

Let's use **cursorLoader** to generate a pattern in accessing a **contentProvider** from UI. The UI (Fragment or the Activity) calls the **cursorLoader** which in turns calls the **ContentProvider** through the **contentResolver** and the result is returned delete) functions of persistent storage.

Pattern for Accessing Content Provider

Let's use **cursorLoader** to generate a pattern in accessing a **contentProvider** from UI. The UI (Fragment or the Activity) calls the **cursorLoader** which in turns calls the **ContentProvider** through the **contentResolver** and the result is returned to the UI. This pattern allows the UI to continue to be available to the user while the query is running.



Interaction between ContentProvider, other classes, and storage.

How is Content Provider Used?

In our quest to understand the content providers, let's start by using a content provider to retrieve data from **User Dictionary** tables. From the above pattern of how content providers work, we intend to follow the pattern and use the content provider likewise to retrieve data from the UserDictionary tables.

Before we can retrieve data from the UserDictionary table like **word** table, we need the **content URI** that points to the **word** table. The ContentURI follows a pattern, the first part is **content://**, which is the **scheme**, and the second part is the **authority**, which uniquely identifies the content provider, and the last part is the **path** to the table.

The **ContentResolver** which is the client of the **contentProvider** uses the **authority** to uniquely identify the a particular content provider to dispatch the request to. Using this content URI to access the first record of Word table, we could simply do this by provider to dispatch the request to. Using this content URI to access the first record of Word table, we could simply do this by appending the particular ID to the last part of the content uri.

```
//the content uri of the words
table in USERDICTIONARY.
content://user_dictionary/words

//Accessing a record of the
words table with ID 1
Uri singleUri =
ContentUris.withAppendedId(UserD
ictionary.Words.CONTENT_URI,1);
```

Before we start reading records from the table, we need a permission to do that and this can be done in the **androidManifest.xml** as done below.

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

Querying for a data

Now after the permission added in the android manifest file, we can read a record with the *contentResolver#query()* method

Querying for a data

Now after the permission added in the android manifest file, we can read a record with the *contentResolver#query()* method which accepts some arguments.

```
private void queryWordData() {
        // A "projection" defines the colu
        String[] projections = {
                UserDictionary.Words._ID,
                UserDictionary.Words.WORD,
                UserDictionary.Words.APP_I
                UserDictionary.Words.LOCAL
        };

        //word to query from the UI
        String searchItem = searchWord.get

        // Defines a string to contain the
        String selectionClause = null;

        // Initializes an array to contain
        String[] selectionArgs = {""};

        //pre-process the input to check i
        if (searchItem ==  null){
            selectionClause = null;
            selectionArgs[0] = "";
        }else{
            selectionClause = UserDictiona
            selectionArgs[0] = searchItem;
        }
```

```java
                        String locale = cursor
                        cursor.moveToNext();
                    }
                }
            }
        }
```

queryWord.java hosted with ❤ by GitHub   view raw

Inserting a data

To insert data into a provider, you call *contentResolver#insert()* to add a new row to the provider and returns a content URI for that row.

```java
private void insertWordData(){
        // Defines an object to contain th
        ContentValues contentValues = new
        contentValues.put(UserDictionary.W
        contentValues.put(UserDictionary.W
        contentValues.put(UserDictionary.W
        contentValues.put(UserDictionary.W

        Uri newUri = getContentResolver().
                UserDictionary.Words.CONTE
                contentValues    // the val
        );

        // this gives the last id inserted
        long id = ContentUris.parseId(User

    }
```

insertWordData.java hosted with ❤ by GitHub   view raw

12:21 PM

Updating a data

To update a data, we use ***contentValues*** with the updated values just as you did with a insertion data and the selection criteria like we did with a query. We use *contentResolver#update() in order to update a contentValues.*

```
private void updateWordData() {
          // Defines an object to contain th
          ContentValues updateValues = new (

          // Defines selection criteria for
          String selectionClause = UserDicti
          String[] selectionArgs = {"en_%"};

          // Defines a variable to contain t
          int rowsUpdated = 0;

          /*
           * Sets the updated value and upda
           */
          updateValues.putNull(UserDictionar

          rowsUpdated = getContentResolver()
                  UserDictionary.Words.CONTE
                  updateValues,
                  selectionClause,
                  selectionArgs
```

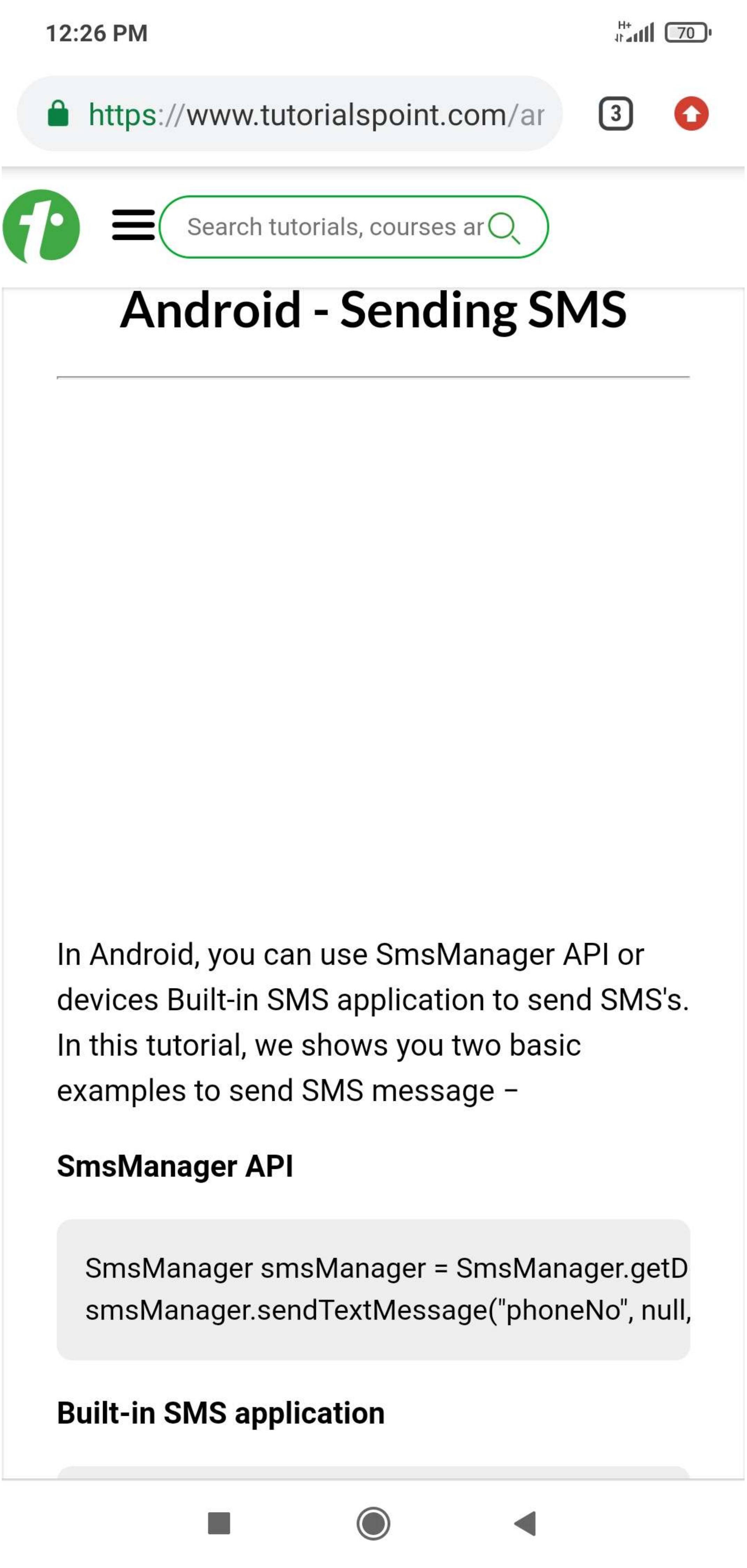
59

Creating A Content Provider

Now, we already know that content provider manages the access to data repository. We create a class that implements *ContentProvider* which is an interface between your provider and other applications. Content providers are meant to share data to other applications but you can also make your activities and fragments query and modify data managed by your provider.

Before we start with the creation of the content provider, we need to create our data store or repository from which the content provider will manage the data access. Let's go ahead and create our database, and you can create database using **android Room database, Sugar DB, or normal SQLite** database which we will use in our example.

Creating the SQLite Database helper and this is going to help us create, upgrade and manage our database. Now we create **DBHelper.java** class.


12:26 PM
https://www.tutorialspoint.com/ar
Search tutorials, courses ar

Android - Sending SMS

In Android, you can use SmsManager API or devices Built-in SMS application to send SMS's. In this tutorial, we shows you two basic examples to send SMS message –

SmsManager API

```
SmsManager smsManager = SmsManager.getD
smsManager.sendTextMessage("phoneNo", null,
```

Built-in SMS application

Built-in SMS application

```
Intent sendIntent = new Intent(Intent.ACTION_VI
sendIntent.putExtra("sms_body", "default conten
sendIntent.setType("vnd.android-dir/mms-sms")
startActivity(sendIntent);
```

Of course, both need **SEND_SMS permission**.

```
<uses-permission android:name="android.permi
```

Apart from the above method, there are few other important functions available in SmsManager class. These methods are listed below –

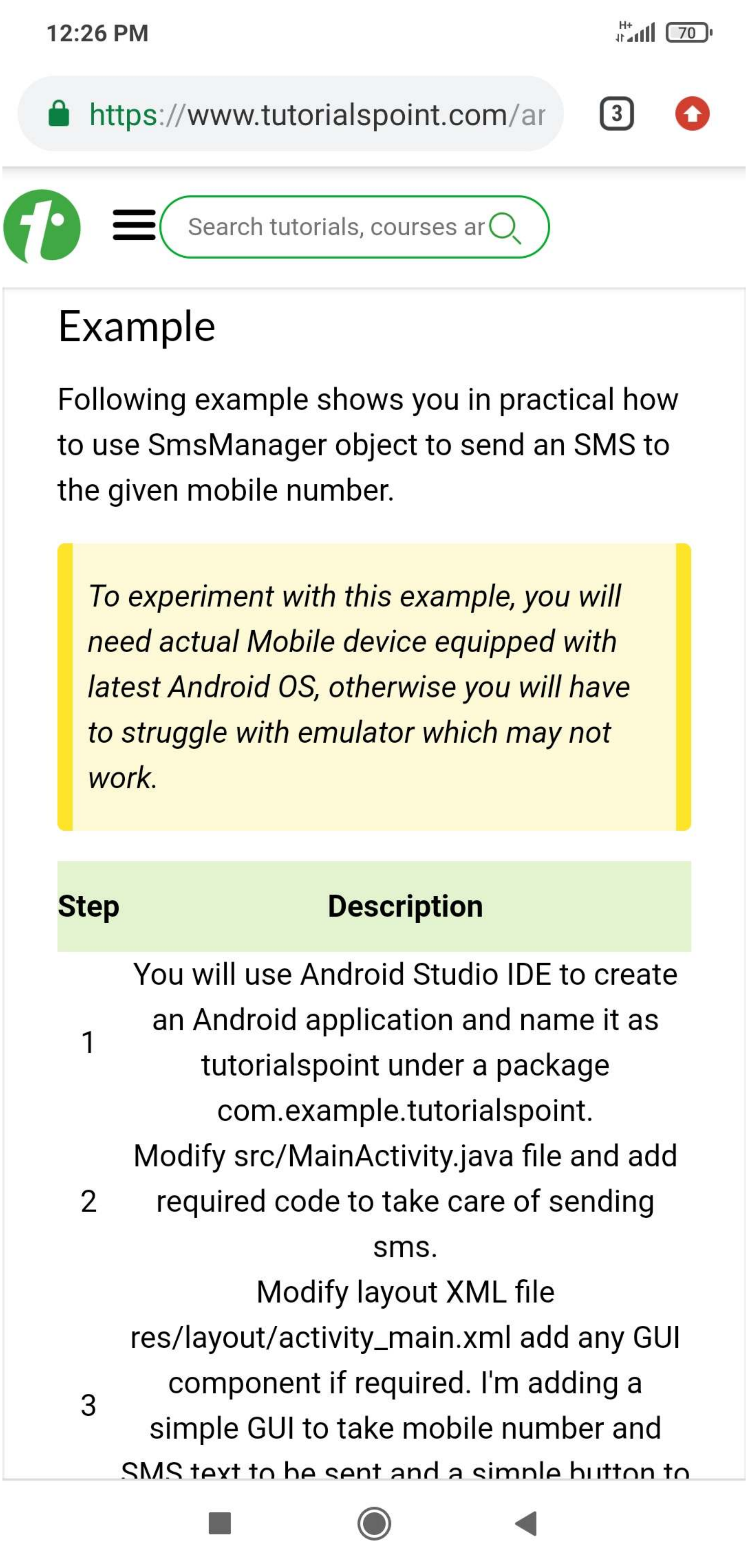
Sr.No.	Method & Description
1	**ArrayList<String> divideMessage(String text)** This method divides a message text into several fragments, none bigger than the maximum SMS message size.
2	**static SmsManager getDefault()** This method is used to get the default instance of the SmsManager
	void sendDataMessage(String

Sr.No.	Method & Description
1	**ArrayList<String> divideMessage(String text)** This method divides a message text into several fragments, none bigger than the maximum SMS message size.
2	**static SmsManager getDefault()** This method is used to get the default instance of the SmsManager
3	**void sendDataMessage(String destinationAddress, String scAddress, short destinationPort, byte[] data, PendingIntent sentIntent, PendingIntent deliveryIntent)** This method is used to send a data based SMS to a specific application port.
4	**void sendMultipartTextMessage(String destinationAddress, String scAddress, ArrayList<String> parts, ArrayList<PendingIntent> sentIntents, ArrayList<PendingIntent> deliveryIntents)** Send a multi-part text based SMS.
5	**void sendTextMessage(String destinationAddress, String scAddress, String text, PendingIntent sentIntent, PendingIntent deliveryIntent)** Send a text based SMS.

	ArrayList<String> parts,
4	**ArrayList<PendingIntent> sentIntents, ArrayList<PendingIntent> deliveryIntents)** Send a multi-part text based SMS.
5	**void sendTextMessage(String destinationAddress, String scAddress, String text, PendingIntent sentIntent, PendingIntent deliveryIntent)** Send a text based SMS.

Example

Following example shows you in practical how to use SmsManager object to send an SMS to


12:26 PM
https://www.tutorialspoint.com/ar
Search tutorials, courses ar

Example

Following example shows you in practical how to use SmsManager object to send an SMS to the given mobile number.

To experiment with this example, you will need actual Mobile device equipped with latest Android OS, otherwise you will have to struggle with emulator which may not work.

Step	Description
1	You will use Android Studio IDE to create an Android application and name it as tutorialspoint under a package com.example.tutorialspoint.
2	Modify src/MainActivity.java file and add required code to take care of sending sms.
3	Modify layout XML file res/layout/activity_main.xml add any GUI component if required. I'm adding a simple GUI to take mobile number and SMS text to be sent and a simple button to

Step	Description
1	You will use Android Studio IDE to create an Android application and name it as tutorialspoint under a package com.example.tutorialspoint.
2	Modify src/MainActivity.java file and add required code to take care of sending sms.
3	Modify layout XML file res/layout/activity_main.xml add any GUI component if required. I'm adding a simple GUI to take mobile number and SMS text to be sent and a simple button to send SMS.
4	No need to define default string constants at res/values/strings.xml. Android studio takes care of default constants.
5	Modify AndroidManifest.xml as shown below
6	Run the application to launch Android emulator and verify the result of the changes done in the application.

Following is the content of the modified main activity file **src/com.example.tutorialspoint/MainActivity.java**.

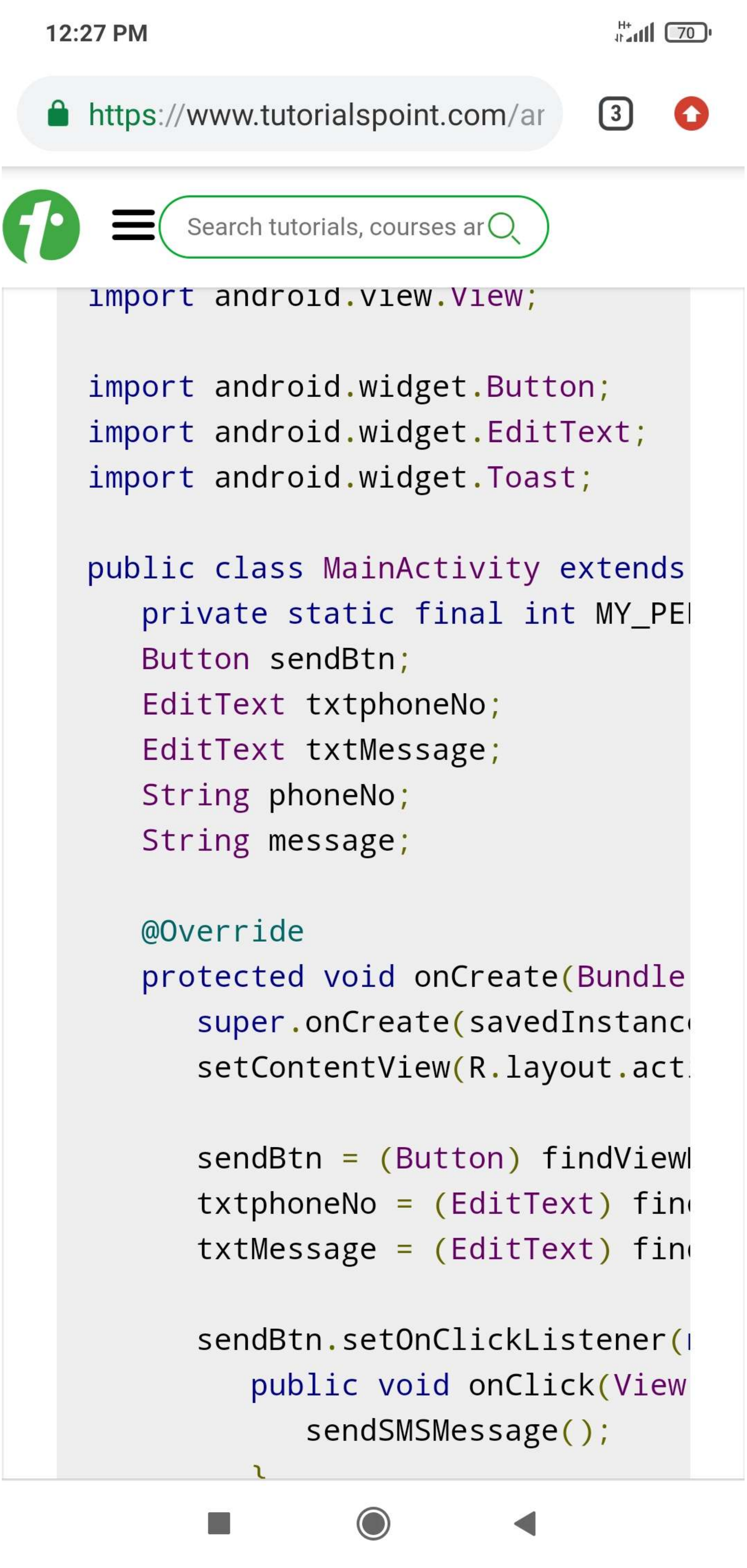
12:26 PM

Search tutorials, courses ar

changes done in the application.

Following is the content of the modified main activity file **src/com.example.tutorialspoint/MainActivity.java**.

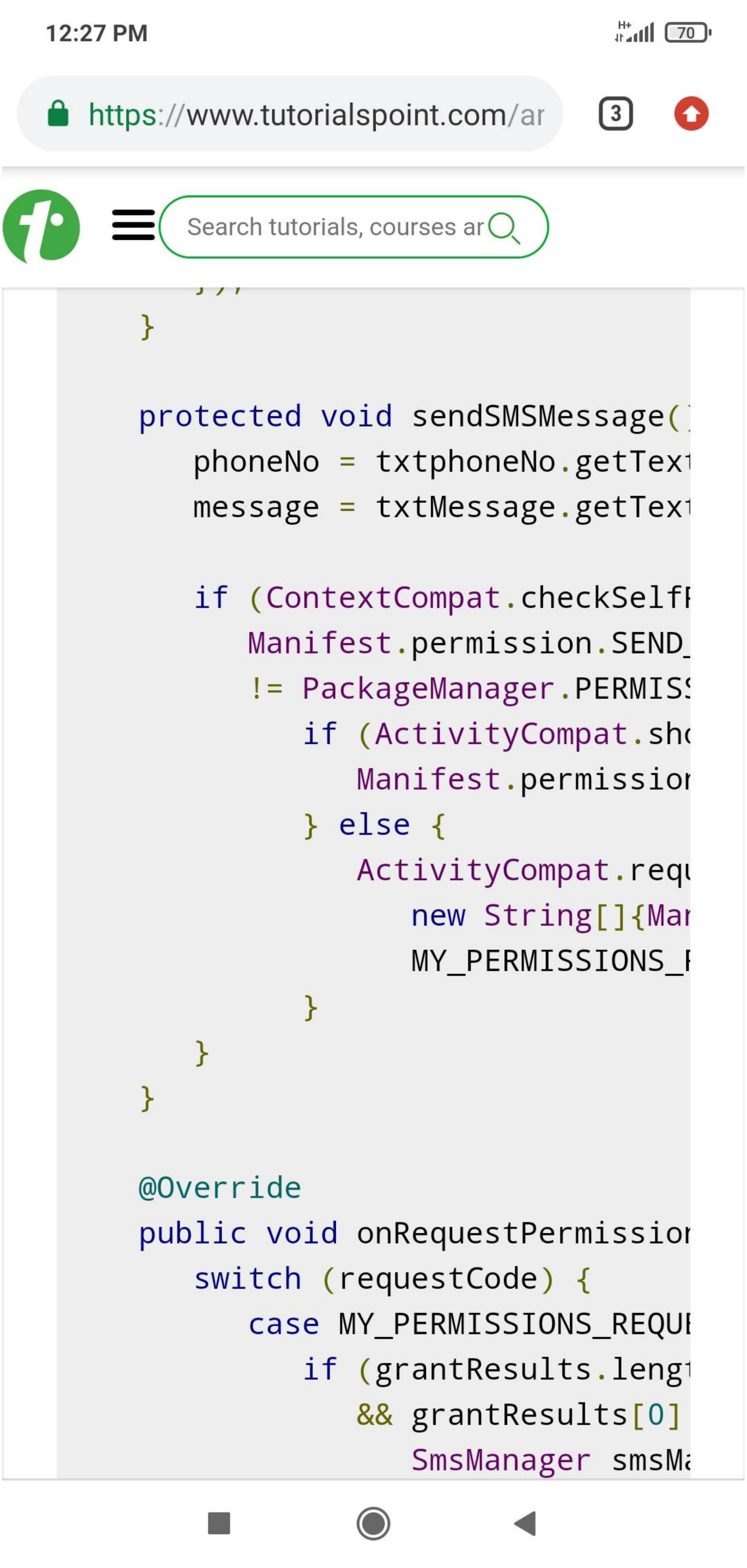
```
package com.example.tutorialspoin

import android.Manifest;
import android.content.pm.Package
import android.os.Bundle;
import android.app.Activity;

import android.support.v4.app.Act
import android.support.v4.content
import android.telephony.SmsManag

import android.util.Log;
import android.view.Menu;
import android.view.View;

import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends
   private static final int MY_PE
   Button sendBtn;
```


12:27 PM

https://www.tutorialspoint.com/ar

Search tutorials, courses ar

```
import android.view.View;

import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends
   private static final int MY_PE
   Button sendBtn;
   EditText txtphoneNo;
   EditText txtMessage;
   String phoneNo;
   String message;

   @Override
   protected void onCreate(Bundle
      super.onCreate(savedInstanc
      setContentView(R.layout.act

      sendBtn = (Button) findView
      txtphoneNo = (EditText) fin
      txtMessage = (EditText) fin

      sendBtn.setOnClickListener(
         public void onClick(View
            sendSMSMessage();
         }
```


12:27 PM

https://www.tutorialspoint.com/ar

Search tutorials, courses ar

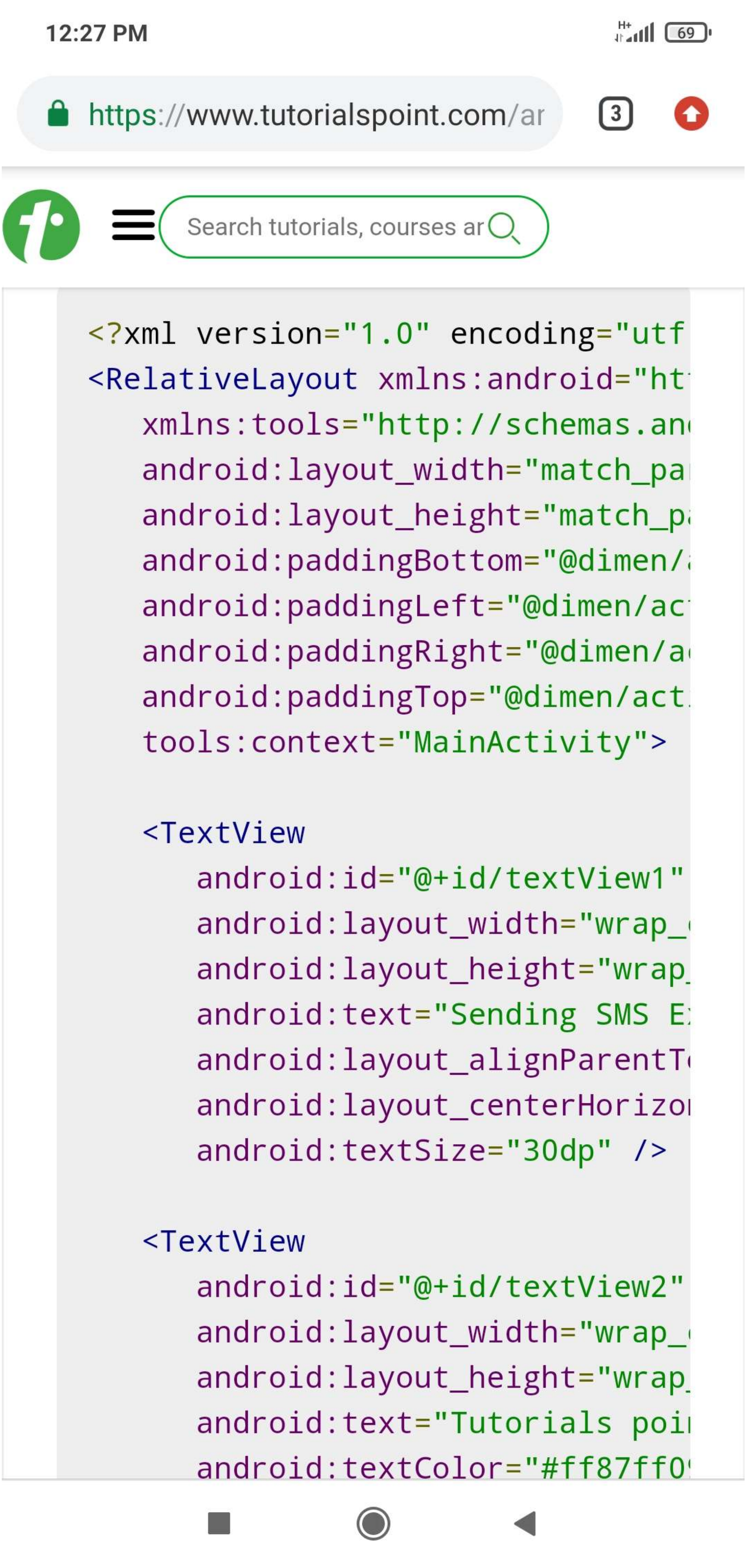
```
      });
   }

   protected void sendSMSMessage()
      phoneNo = txtphoneNo.getText
      message = txtMessage.getText

      if (ContextCompat.checkSelfP
         Manifest.permission.SEND_
         != PackageManager.PERMISS
            if (ActivityCompat.sho
               Manifest.permission
            } else {
               ActivityCompat.requ
                  new String[]{Man
                  MY_PERMISSIONS_R
            }
      }
   }

   @Override
   public void onRequestPermission
      switch (requestCode) {
         case MY_PERMISSIONS_REQUE
            if (grantResults.lengt
               && grantResults[0]
                  SmsManager smsMa
```

```
                case MY_PERMISSIONS_REQUI
                    if (grantResults.lengt
                        && grantResults[0]
                            SmsManager smsMa
                            smsManager.sendT
                            Toast.makeText(g
                                Toast.LENGTH_
                    } else {
                        Toast.makeText(getA
                            "SMS faild, plea
                        return;
                    }
                }
            }

        }
    }
```

Following will be the content of **res/layout/activity_main.xml** file −

> *Here abc indicates about tutorialspoint logo*

```
<?xml version="1.0" encoding="utf
<RelativeLayout xmlns:android="htt
    xmlns:tools="http://schemas.and
    android:layout_width="match_pa
```


12:27 PM

https://www.tutorialspoint.com/ar

Search tutorials, courses ar

```
<?xml version="1.0" encoding="utf
<RelativeLayout xmlns:android="ht
   xmlns:tools="http://schemas.an
   android:layout_width="match_pa
   android:layout_height="match_p
   android:paddingBottom="@dimen/
   android:paddingLeft="@dimen/ac
   android:paddingRight="@dimen/a
   android:paddingTop="@dimen/act
   tools:context="MainActivity">

   <TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_
      android:layout_height="wrap
      android:text="Sending SMS E
      android:layout_alignParentT
      android:layout_centerHorizo
      android:textSize="30dp" />

   <TextView
      android:id="@+id/textView2"
      android:layout_width="wrap_
      android:layout_height="wrap
      android:text="Tutorials poi
      android:textColor="#ff87ff0
```

```
    <ImageButton
        android:layout_width="wrap_
        android:layout_height="wrap_
        android:id="@+id/imageButto
        android:src="@drawable/abc"
        android:layout_below="@+id/
        android:layout_centerHorizo

    <EditText
        android:layout_width="wrap_
        android:layout_height="wrap_
        android:id="@+id/editText"
        android:hint="Enter Phone Nu
        android:phoneNumber="true"
        android:textColorHint="@col
        android:layout_below="@+id/
        android:layout_centerHorizo

    <EditText
        android:layout_width="wrap_
        android:layout_height="wrap_
        android:id="@+id/editText2"
        android:layout_below="@+id/
        android:layout_alignLeft="@
        android:layout_alignStart="
        android:textColorHint="@col
        android:layout_alignRight="
        android:layout_alignEnd="@+
        android:layout_height="wrap_
        android:text="Send Sms"
        android:id="@+id/btnSendSMS
        android:layout_below="@+id/
        android:layout_centerHorizo
        android:layout_marginTop="4

</RelativeLayout>
```

Following will be the content of **res/values/strings.xml** to define two new constants −

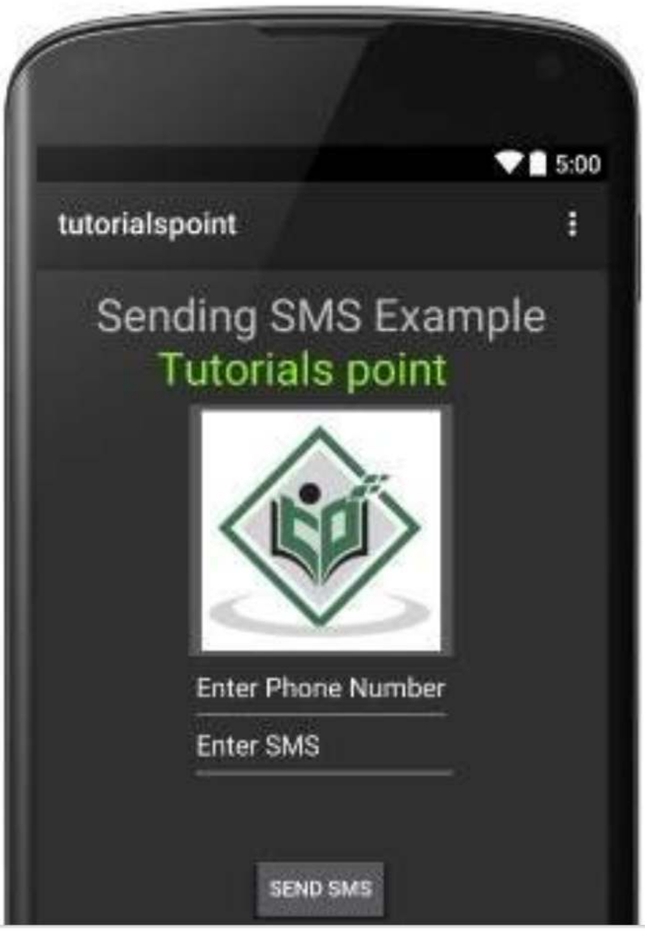
```
<?xml version="1.0" encoding="utf
<resources>
   <string name="app_name">tutori
</resources>
```

Following is the default content of **AndroidManifest.xml** −

```
<?xml version="1.0" encoding="utf
<manifest xmlns:android="http://s
   package="com.example.tutorials

   <uses-permission android:name=
</manifest>
```

Let's try to run your **tutorialspoint** application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run

icon from the toolbar. Before starting your application, Android studio installer will display following window to select an option where you want to run your Android application.



Now you can enter a desired mobile number and a text message to be sent on that number. Finally click on **Send SMS** button to send your SMS. Make sure your GSM/CDMA connection is working fine to deliver your SMS to its recipient.

You can take a number of SMS separated by comma and then inside your program you will have to parse them into an array string and finally you can use a loop to send message to all the given numbers. That's how you can write your own SMS client. Next section will show you how to use existing SMS client to send SMS.

Using Built-in Intent to send SMS

You can use Android Intent to send SMS by calling built-in SMS functionality of the Android. Following section explains different parts of our Intent object required to send an SMS.

Intent Object - Action to send SMS

You will use **ACTION_VIEW** action to launch an SMS client installed on your Android device. Following is simple syntax to create an intent with ACTION_VIEW action.

```
Intent smsIntent = new Intent(Intent.ACTION_VIE
```

Intent Object - Data/Type to send SMS

To send an SMS you need to specify **smsto:** as URI using setData() method and data type will be to **vnd.android-dir/mms-sms** using setType() method as follows –

```
smsIntent.setData(Uri.parse("smsto:"));
```

Android - Sending Email

> ***Email** is messages distributed by electronic means from one system user to one or more recipients via a network.*

Before starting Email Activity, You must know Email functionality with intent, Intent is carrying data from one component to another component with-in the application or outside the application.

Before starting Email Activity, You must know Email functionality with intent, Intent is carrying data from one component to another component with-in the application or outside the application.

To send an email from your application, you don't have to implement an email client from the beginning, but you can use an existing one like the default Email app provided from Android, Gmail, Outlook, K-9 Mail etc. For this purpose, we need to write an Activity that launches an email client, using an implicit Intent with the right action and data. In this example, we are going to send an email from our app by using an Intent object that launches existing email clients.

Following section explains different parts of our Intent object required to send an email.

Intent Object - Action to send Email

You will use **ACTION_SEND** action to launch an email client installed on your Android device. Following is simple syntax to create an intent with ACTION_SEND action.

12:29 PM

Search tutorials, courses ar

Intent Object - Action to send Email

You will use **ACTION_SEND** action to launch an email client installed on your Android device. Following is simple syntax to create an intent with ACTION_SEND action.

```
Intent emailIntent = new Intent(Intent.ACTION_S
```

Intent Object - Data/Type to send Email

To send an email you need to specify **mailto:**

Intent Object - Data/Type to send Email

To send an email you need to specify **mailto:** as URI using setData() method and data type will be to **text/plain** using setType() method as follows –

```
emailIntent.setData(Uri.parse("mailto:"));
emailIntent.setType("text/plain");
```

Intent Object - Extra to send Email

Android has built-in support to add TO, SUBJECT, CC, TEXT etc. fields which can be attached to the intent before sending the intent to a target email client. You can use following extra fields in your email –

Sr.No.	Extra Data & Description
1	**EXTRA_BCC** A String[] holding e-mail addresses that should be blind carbon copied.
2	**EXTRA_CC** A String[] holding e-mail addresses that should be carbon copied.

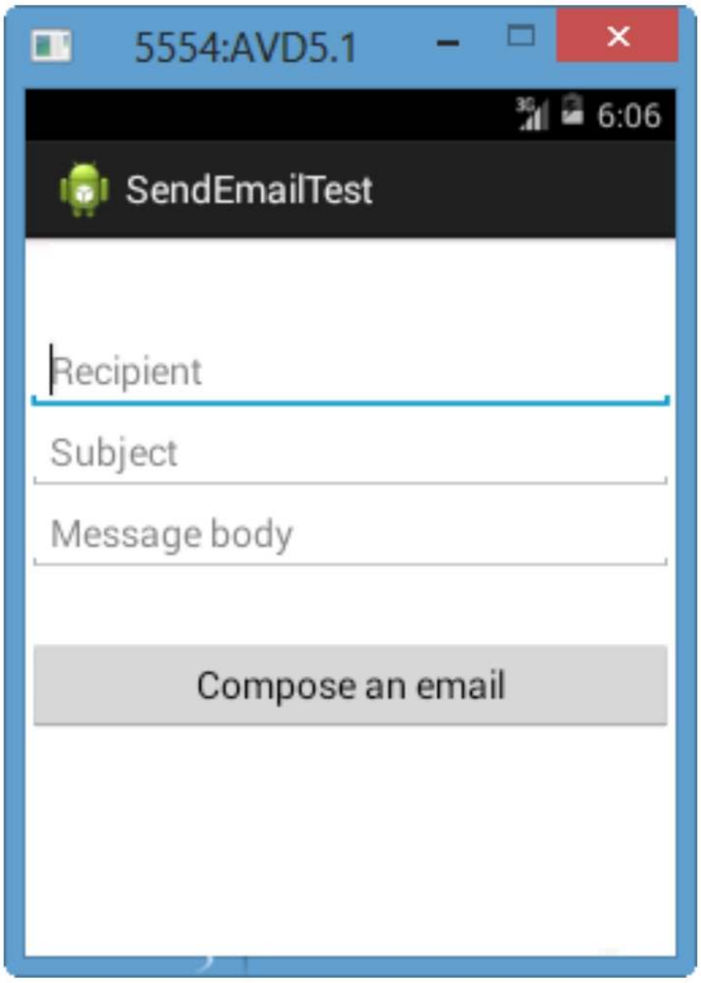
1	**EXTRA_BCC** A String[] holding e-mail addresses that should be blind carbon copied.
2	**EXTRA_CC** A String[] holding e-mail addresses that should be carbon copied.
3	**EXTRA_EMAIL** A String[] holding e-mail addresses that should be delivered to.
4	**EXTRA_HTML_TEXT** A constant String that is associated with the Intent, used with ACTION_SEND to supply an alternative to EXTRA_TEXT as HTML formatted text.
5	**EXTRA_SUBJECT** A constant string holding the desired subject line of a message.
6	**EXTRA_TEXT** A constant CharSequence that is associated with the Intent, used with ACTION_SEND to supply the literal data to be sent.
7	**EXTRA_TITLE** A CharSequence dialog title to provide to the user when used with a ACTION_CHOOSER.

Here is an example showing you how to assign extra data to your intent −

extra data to your intent –

```
emailIntent.putExtra(Intent.EXTRA_EMAIL  , new
emailIntent.putExtra(Intent.EXTRA_SUBJECT, "su
emailIntent.putExtra(Intent.EXTRA_TEXT   , "Mes
```

The out-put of above code is as below shown an image



Email Example

Email Example

Example

Following example shows you in practical how to use Intent object to launch Email client to send an Email to the given recipients.

To Email experiment with this example, you will need actual Mobile device equipped send an Email to the given recipients.

To Email experiment with this example, you will need actual Mobile device equipped with latest Android OS, otherwise you might get struggle with emulator which may not work properly. Second you will need to have an Email client like GMail(By default every android version having Gmail client App) or K9mail installed on your device.

Step	Description
1	You will use Android studio to create an Android application and name it as Tutorialspoint under a package com.example.tutorialspoint.
2	Modify src/MainActivity.java file and add required code to take care of sending email.
3	Modify layout XML file res/layout/activity_main.xml add any GUI component if required. I'm adding a simple button to launch Email Client.
4	Modify res/values/strings.xml to define required constant values

12:30 PM

Search tutorials, courses ar

2	required code to take care of sending email.
3	Modify layout XML file res/layout/activity_main.xml add any GUI component if required. I'm adding a simple button to launch Email Client.
4	Modify res/values/strings.xml to define required constant values
5	Modify AndroidManifest.xml as shown below
6	Run the application to launch Android emulator and verify the result of the changes done in the application.

Following is the content of the modified main activity file **src/com.example.Tutorialspoint/MainActivity.java**.

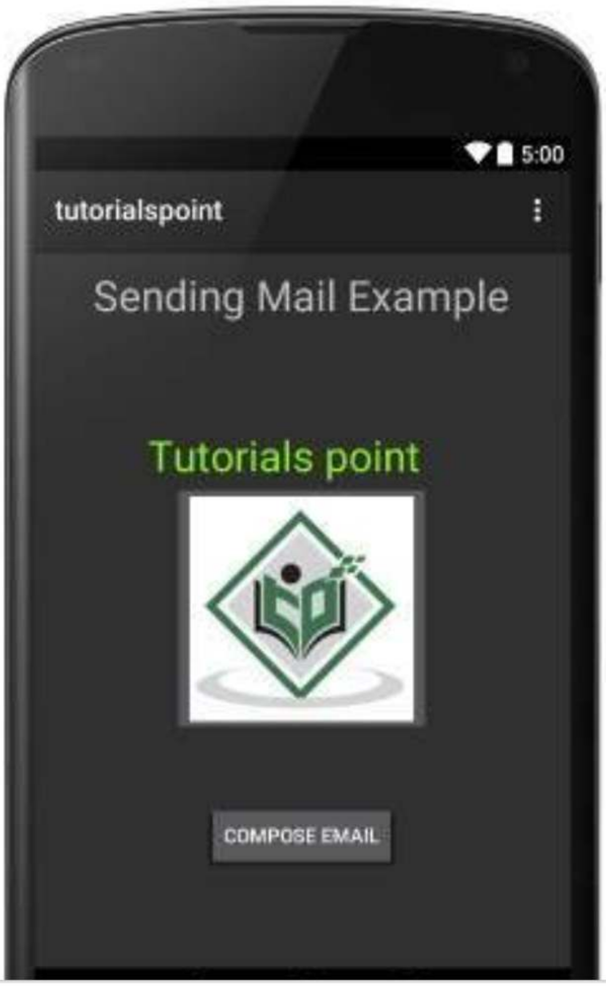
```
package com.example.tutorialspoin

import android.net.Uri;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends
   @Override
   protected void onCreate(Bundle
      super.onCreate(savedInstanc
      setContentView(R.layout.act

      Button startBtn = (Button)
      startBtn.setOnClickListener
         public void onClick(View
            sendEmail();
         }
      });
   }

   protected void sendEmail() {
      Log.i("Send email", "");
      String[] TO = {""};
      String[] CC = {""};
      Intent emailIntent = new In

      emailIntent.setData(Uri.par
    protected void sendEmail() {
        Log.i("Send email", "");
        String[] TO = {""};
        String[] CC = {""};
        Intent emailIntent = new In

        emailIntent.setData(Uri.par
        emailIntent.setType("text/p
        emailIntent.putExtra(Intent
        emailIntent.putExtra(Intent
        emailIntent.putExtra(Intent
        emailIntent.putExtra(Intent

        try {
            startActivity(Intent.cre
            finish();
            Log.i("Finished sending 
        } catch (android.content.Ac
            Toast.makeText(MainActiv
        }
    }
}
```

Following will be the content of **res/layout/activity_main.xml** file −

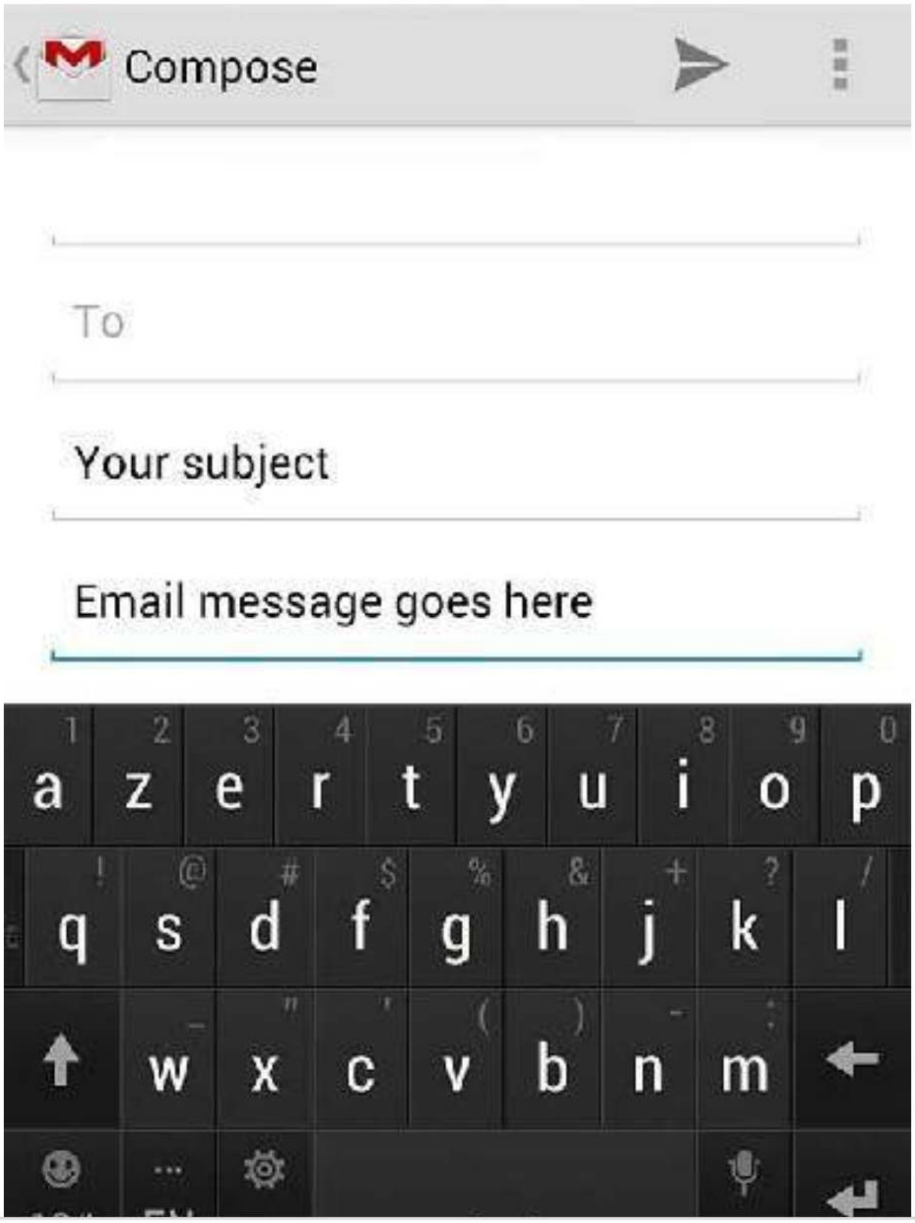
Here abc indicates about tutorialspoint logo

Let's try to run your **tutorialspoint** application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android Studio, open one of your project's activity files and click Run

icon from the toolbar. Before starting your application, Android studio installer will display following window to select an option where you want to run your Android application.Select your mobile device as an option and then check your mobile device which will display following screen –



Now use **Compose Email** button to list down all the installed email clients. From the list, you can choose one of email clients to send your email. I'm going to use Gmail client to send my email which will have all the provided defaults fields available as shown below. Here **From:** will be default email ID you have registered for your Android device.



Android Google Map

Android provides facility to integrate Google map in our application. Google map displays your current location, navigate location direction, search location etc. We can also customize Google map according to our requirement.

Types of Google Maps

There are four different types of Google maps, as well as an optional to no map at all. Each of them gives different view on map. These maps are as follow:

12:32 PM

Types of Google Maps

There are four different types of Google maps, as well as an optional to no map at all. Each of them gives different view on map. These maps are as follow:

1. **Normal:** This type of map displays typical road map, natural features like river and some features build by humans.
2. **Hybrid:** This type of map displays satellite photograph data with typical road maps. It also displays road and feature labels.
3. **Satellite:** Satellite type displays satellite photograph data, but doesn't display road and feature labels.
4. **Terrain:** This type displays photographic data. This includes colors, contour lines and labels and perspective shading.
5. **None:** This type displays an empty grid with no tiles loaded.

and feature labels.

4. **Terrain:** This type displays photographic data. This includes colors, contour lines and labels and perspective shading.

5. **None:** This type displays an empty grid with no tiles loaded.

Syntax of different types of map

```
googleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL
googleMap.setMapType(GoogleMap.MAP_TYPE_HYBRID)
googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLIT
googleMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN)
```

Syntax of different types of map

```
googleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL
googleMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLIT
googleMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN
```

Methods of Google map

Google map API provides several methods that help

12:32 PM 68

ADVERTISEMENT

Syntax of different types of map

```
ogleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
ogleMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
ogleMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
ogleMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
```

Methods of Google map

ADVERTISEMENT

ADVERTISEMENT

Google map API provides several methods that help

12:32 PM

Methods of Google map

ADVERTISEMENT

ADVERTISEMENT

Google map API provides several methods that help to customize Google map. These methods are as following:

Methods	De
addCircle(CircleOptions options)	Thi adc ma
addPolygon(PolygonOptions options)	Thi adc ma

12:32 PM 68

ADVERTISEMENT

Google map API provides several methods that help to customize Google map. These methods are as following:

Methods	Description
addCircle(CircleOptions options)	This method add circle to map.
addPolygon(PolygonOptions options)	This method add polygon to map.
addTileOverlay(TileOverlayOptions options)	This method add tile overlay to the map.
animateCamera(CameraUpdate update)	This method moves the map according to the update with an an ation.
clear()	This method removes everything

X

12:33 PM

	map.
addPolygon(PolygonOptions options)	This method add polygon to map.
addTileOverlay(TileOverlayOptions options)	This method add tile overlay to the map.
animateCamera(CameraUpdate update)	This method moves the map according to the update with an animation.
clear()	This method removes everything from the map.
getMyLocation()	This method returns the currently displayed user location.
moveCamera(CameraUpdate update)	This method reposition the camera according to the instructions

	currently displayed user location.
moveCamera(CameraUpdate update)	This method reposition the camera according to the instructions defined in the update.
setTrafficEnabled(boolean enabled)	This method set the traffic layer on or off.
snapshot(GoogleMap.SnapshotReadyCallback callback)	This method takes a snapshot of the map.
stopAnimation()	This method stops the camera animation if there is any progress.

Example of Google Map

12:33 PM

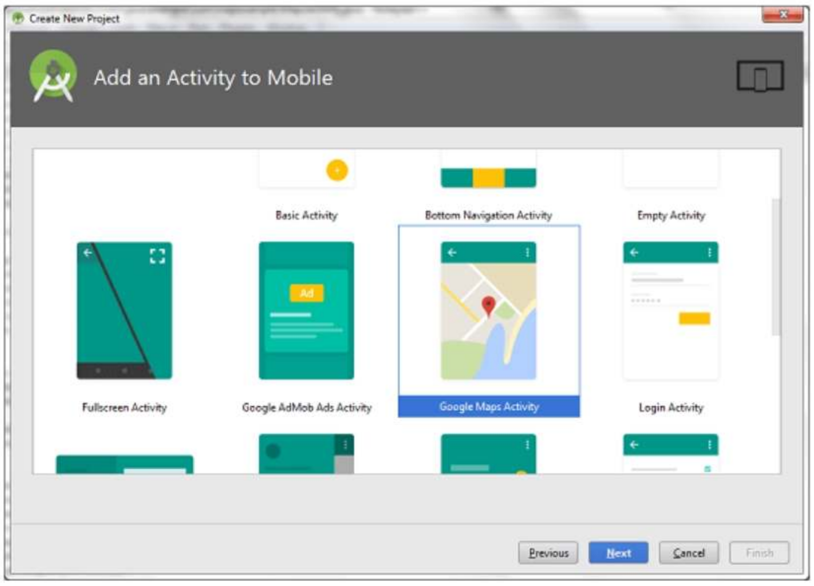
progress.

Example of Google Map

ADVERTISEMENT

ADVERTISEMENT

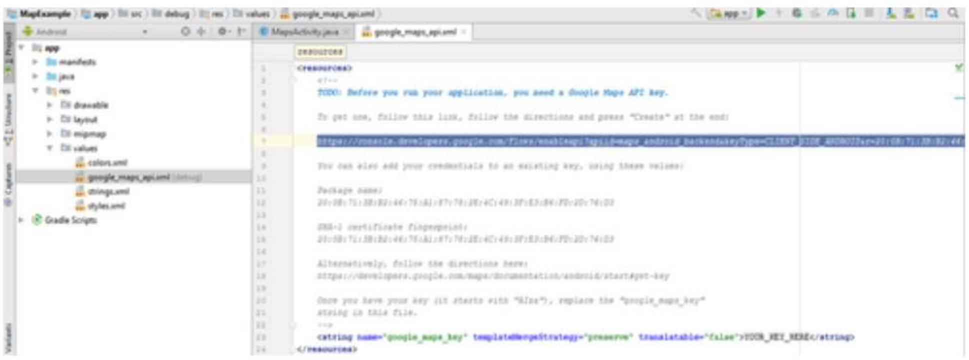
Let's create an example of Google map integrating within our app. For doing this we select Google Maps Activity.



Copy the URL from google_map_api.xml file to generate Google map key.

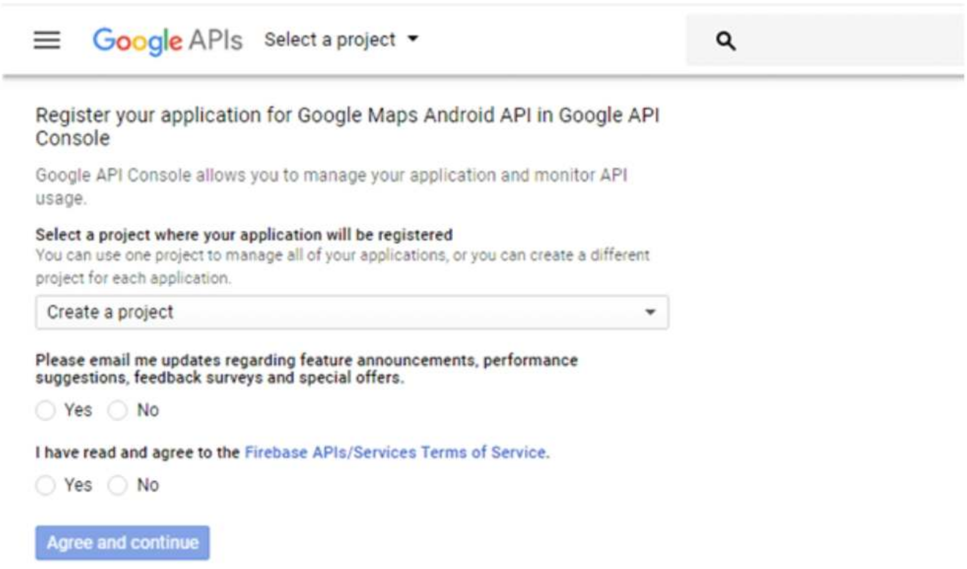
ADVERTISEMENT

ADVERTISEMENT



```
<resources>
    <!--
    TODO: Before you run your application, you need a Google Maps API key.

    To get one, follow this link, follow the directions and press "Create" at the end:

    https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID&r=20:0B:71:3B:B2:46

    You can also add your credentials to an existing key, using these values:

    Package name:
    20:0B:71:3B:B2:46:75:A1:87:78:2E:4C:49:3F:E3:B6:FD:20:74:D3

    SHA-1 certificate fingerprint:
    20:0B:71:3B:B2:46:75:A1:87:78:2E:4C:49:3F:E3:B6:FD:20:74:D3

    Alternatively, follow the directions here:
    https://developers.google.com/maps/documentation/android/start#get-key

    Once you have your key (it starts with "AIza"), replace the "google_maps_key"
    string in this file.
    -->
    <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR_KEY_HERE</string>
</resources>
```

Paste the copied URL at the browser. It will open the following page.



Register your application for Google Maps Android API in Google API Console

Google API Console allows you to manage your application and monitor API usage.

Select a project where your application will be registered
You can use one project to manage all of your applications, or you can create a different project for each application.

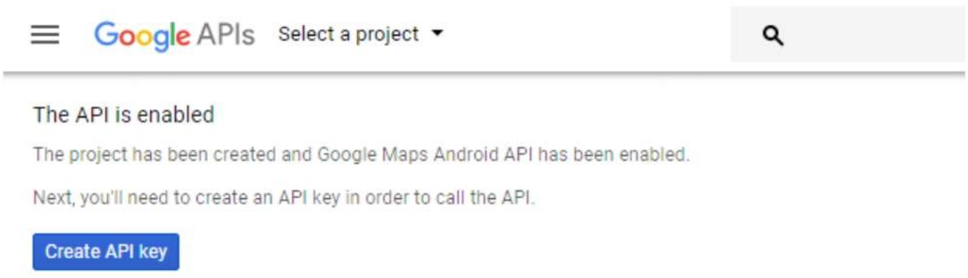
Create a project

Please email me updates regarding feature announcements, performance suggestions, feedback surveys and special offers.
Yes No

I have read and agree to the Firebase APIs/Services Terms of Service.
Yes No

Click on Create API key to generate API key.

ADVERTISEMENT

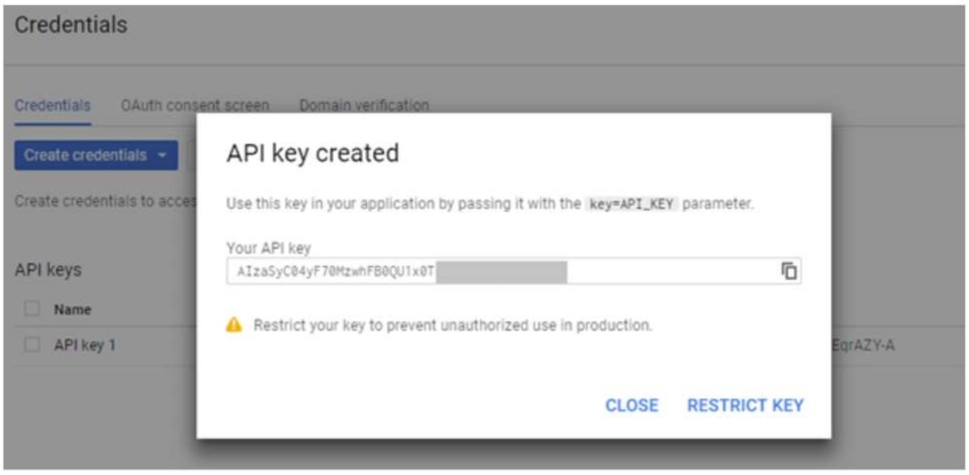


The API is enabled

The project has been created and Google Maps Android API has been enabled.

Next, you'll need to create an API key in order to call the API.

After clicking on Create API key, it will generate our API key displaying the following screen.



API key created

Use this key in your application by passing it with the `key=API_KEY` parameter.

Your API key

`AIzaSyC04yF70MzwhFB0QU1x0T`[redacted]

Restrict your key to prevent unauthorized use in production.

Copy this generated API key in our *google_map_api.xml* file

12:33 PM

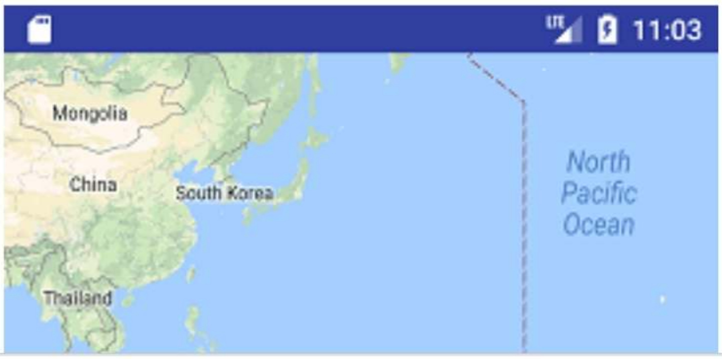
https://www.javatpoint.com/andrc

build.gradel

Add the following dependencies in *build.gradel* file.

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
            implementation   'com.android.support:appcompat-
v7:26.1.0'
    implementation 'com.google.android.gms:play-services-
maps:11.8.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:ru
    androidTestImplementation 'com.android.support.test.es
core:3.0.1'
}
```

Output

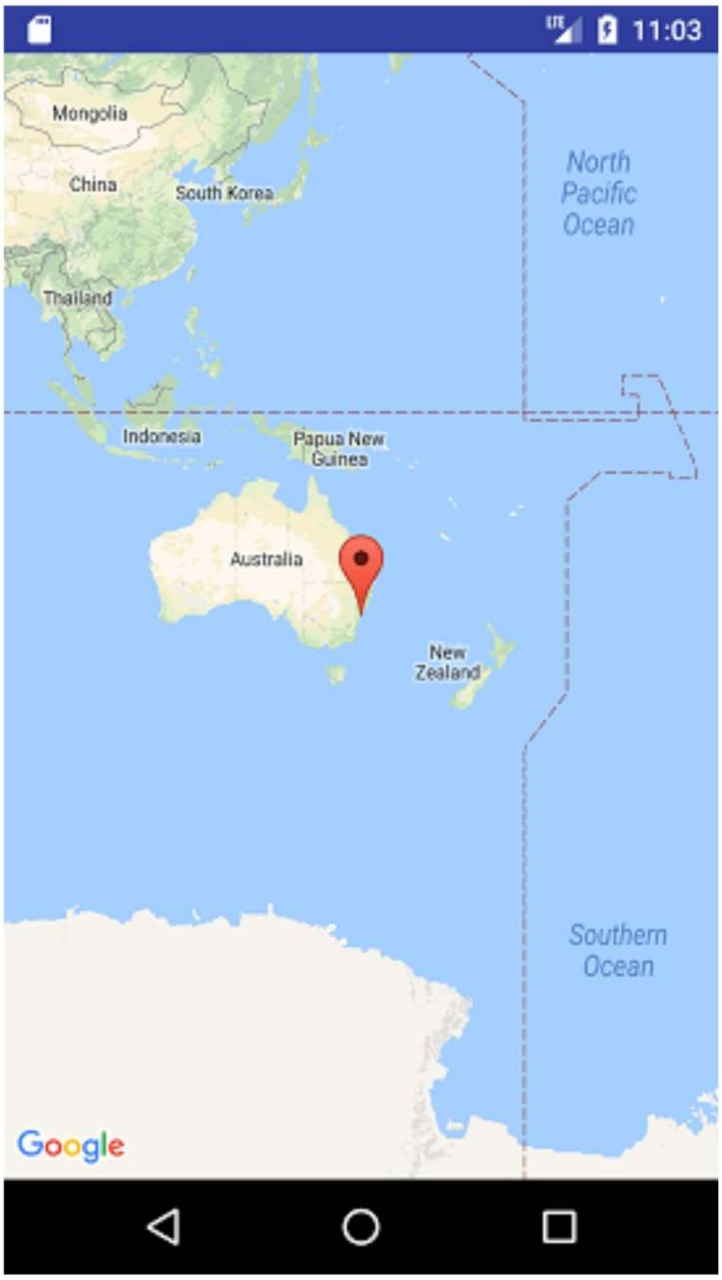


12:33 PM

https://www.javatpoint.com/andrc

```
    androidTestImplementation 'com.android.support.test:ru
    androidTestImplementation 'com.android.support.test.es
core:3.0.1'
}
```

Output



Get Current Location in Android

Last modified on October 24th, 2014 by Joe.

This android tutorial is to help learn location based service in android platform. Knowing the current location in an android mobile will pave the way for developing many innovative Android apps to solve peoples daily problem. For developing location aware application in android, it needs location providers. There are two types of location providers,

1. GPS Location Provider
2. Network Location Provider

Any one of the above providers is enough to get current location of the user or user’s device. But, it is recommended to use both providers as they both have different advantages. Because, GPS provider will take time to get location at indoor area. And, the Network Location Provider will not get location when the network connectivity is poor.

Network Location Provider vs GPS Location Provider

- Network Location provider is comparatively faster than the GPS provider in providing the location co-ordinates.
- GPS provider may be very very slow in in-door locations and will drain the mobile battery.
- Network location provider depends on the cell tower and will return our nearest tower location.
- GPS location provider, will give our location accurately.

Steps to get location in Android

1. Provide permissions in manifest file for receiving location update
2. Create LocationManager instance as reference to the location service
3. Request location from LocationManager
4. Receive location update from LocationListener on change of location

Provide permissions for receiving location update

Provide permissions for receiving location update

To access current location information through location providers, we need to set permissions with android manifest file.

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

ACCESS_COARSE_LOCATION is used when we use network location provider for our Android app. But, ACCESS_FINE_LOCATION is providing permission for both providers. INTERNET permission is must for the use of network provider.

Create LocationManager instance as reference to the location service

For any background Android Service, we need to get reference for using it.

permission is must for the use of network provider.

Create LocationManager instance as reference to the location service

For any background Android Service, we need to get reference for using it. Similarly, location service reference will be created using getSystemService() method. This reference will be added with the newly created LocationManager instance as follows.

```
locationManager = (LocationManager) getSy
```

Request current location from LocationManager

After creating the location service reference, location updates are requested using requestLocationUpdates() method of LocationManager. For this function, we need to send the type of location provider, number of seconds, distance and the LocationListener object over

Request current location from LocationManager

After creating the location service reference, location updates are requested using requestLocationUpdates() method of LocationManager. For this function, we need to send the type of location provider, number of seconds, distance and the LocationListener object over which the location to be updated.

```
locationManager.requestLocationUpdates(Lo
```

Receive location update from LocationListener on change of location

LocationListener will be notified based on the distance interval specified or the number seconds.

Sample Android App: Current Location Finder

12:36 PM

68

Sample Android App: Current Location Finder

This example provides current location update using GPS provider. Entire Android app code is as follows,

```
package com.javapapers.android.geolocatic

import android.os.Bundle;
import android.app.Activity;
import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.widget.TextView;

import android.util.Log;

public class MainActivity extends Activit
protected LocationManager locationManager
protected LocationListener locationLister
protected Context context;
TextView txtLat;
String lat;
```

XML files for layout and android manifest are as shown below

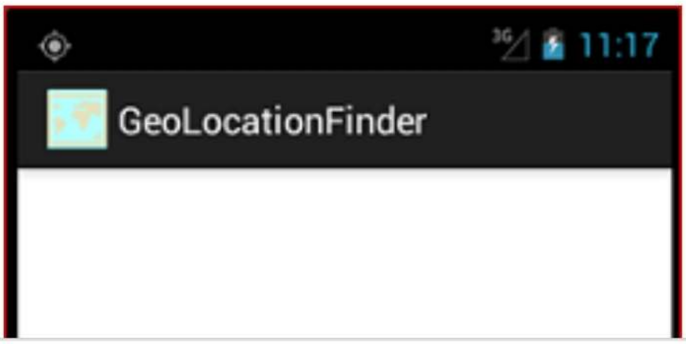
```
<RelativeLayout xmlns:android="http://sch
    xmlns:tools="http://schemas.android.c
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textview1"
```

12:36 PM

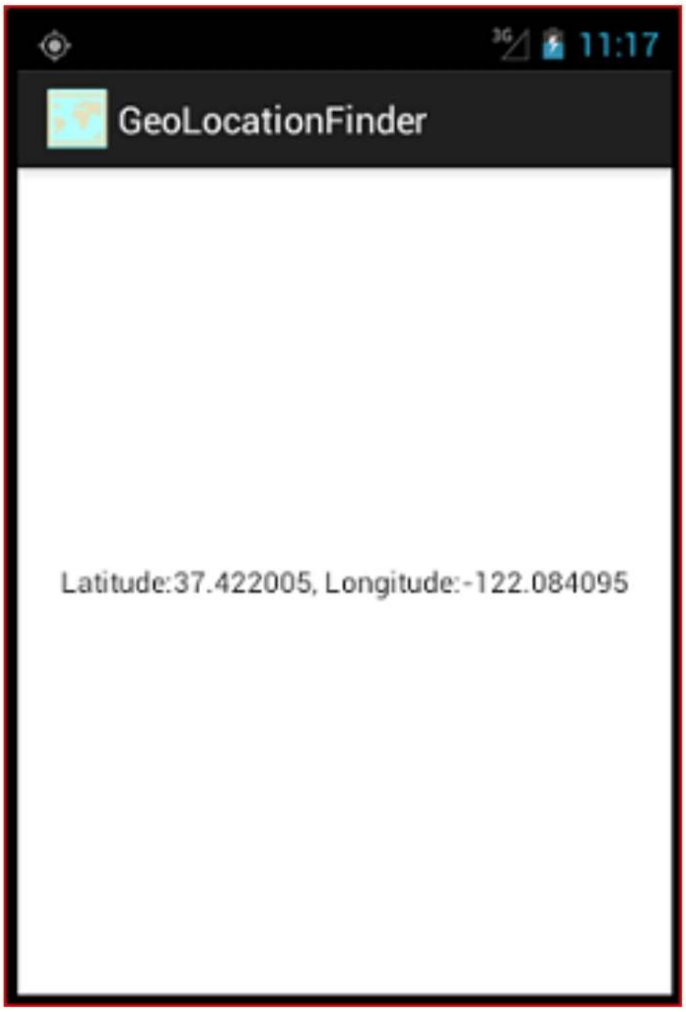
```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textview1"
        android:layout_width="wrap_conter
        android:layout_height="wrap_conte
        android:layout_centerHorizontal='
        android:layout_centerVertical="tr
        android:text="@string/hello_worl

</RelativeLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.a
    package="com.javapapers.android.geol
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17"
    <uses-permission android:name="andro
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launch
        android:label="@string/app_name"
        android:theme="@style/AppBaseThem
        <activity
            android:name="com.javapapers.
            android:label="@string/app_na
```

Android Output



Android Output



```
Latitude:37.422005, Longitude:-122.084095
```

Note: If you are running this Android app with emulator, you need to send the latitude and longitude explicitly for the emulator.

How to send latitude and longitude to android emulator

- Open DDMS perspective in Eclipse (Window -> Open Perspective)
- Select your emulator device

emulator.

How to send latitude and longitude to android emulator

- Open DDMS perspective in Eclipse (Window -> Open Perspective)
- Select your emulator device
- Select the tab named emulator control
- In ‘Location Controls’ panel, ‘Manual’ tab, give the Longitude and Latitude as input and ‘Send’.

Popular Articles

Find Places Nearby in Google Maps using Google Places API–Android App
Android Get Address with Street Name, City for Location with Geocoding
Android Location Fused Provider

Comments on "Get Current Location in Android"

Popular Articles

Find Places Nearby in Google Maps using Google Places API–Android App
Android Get Address with Street Name, City for Location with Geocoding
Android Location Fused Provider

Comments on "Get Current Location in Android"

ANUJ says: *06/05/2013 at 12:09 pm*

nice tutorial sir...........

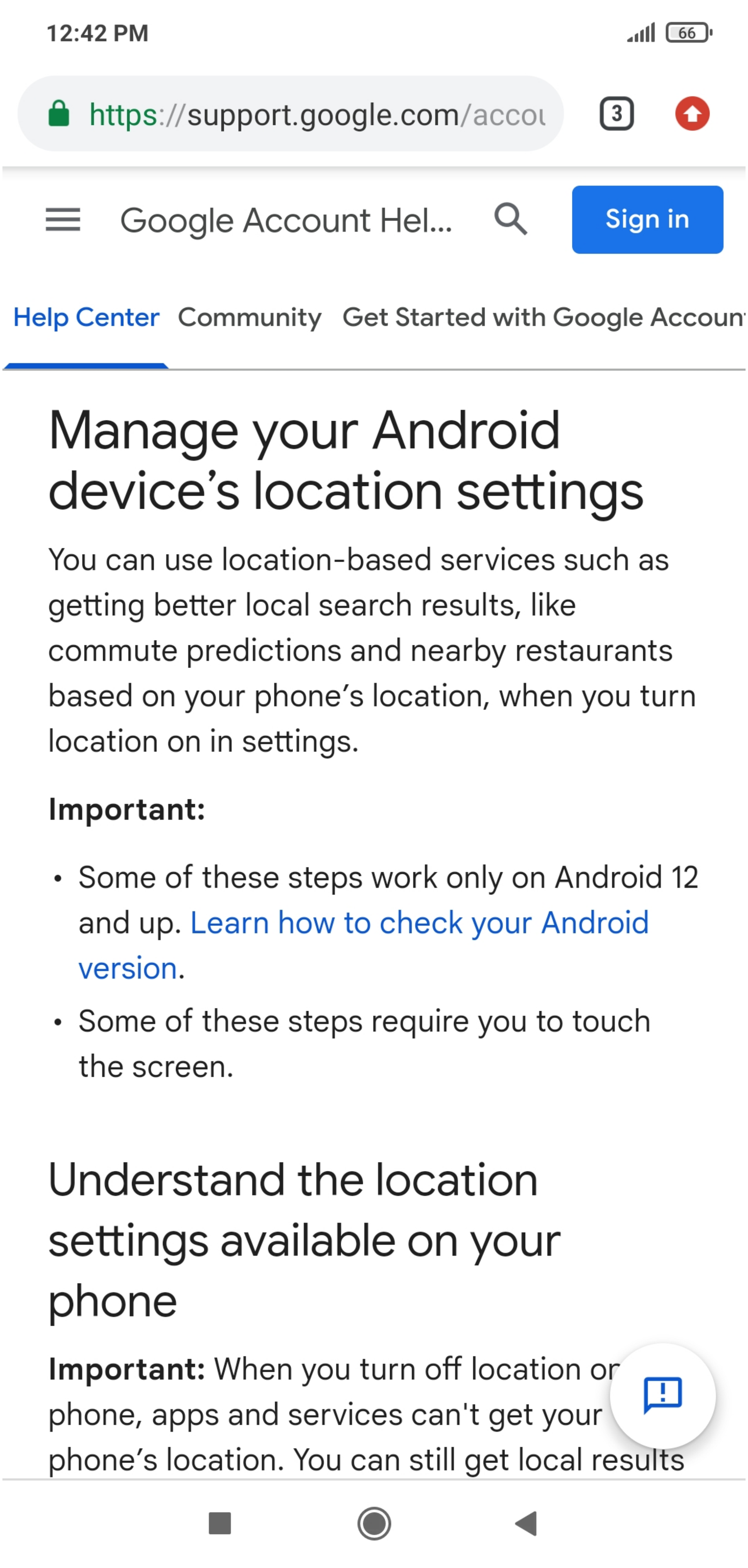
---

s.selvakumar says: *06/05/2013 at 4:50 pm*

Hi Sir,

I am a java developer but not a Android developer. I would like to develop one simple contact save application in android. Could you please publish step by step development procedure for contact book application with sqlite db interaction.

Thanks,
s.selvakumar



12:42 PM

https://support.google.com/accou

Google Account Hel...

Sign in

Help Center Community Get Started with Google Accoun

Manage your Android device’s location settings

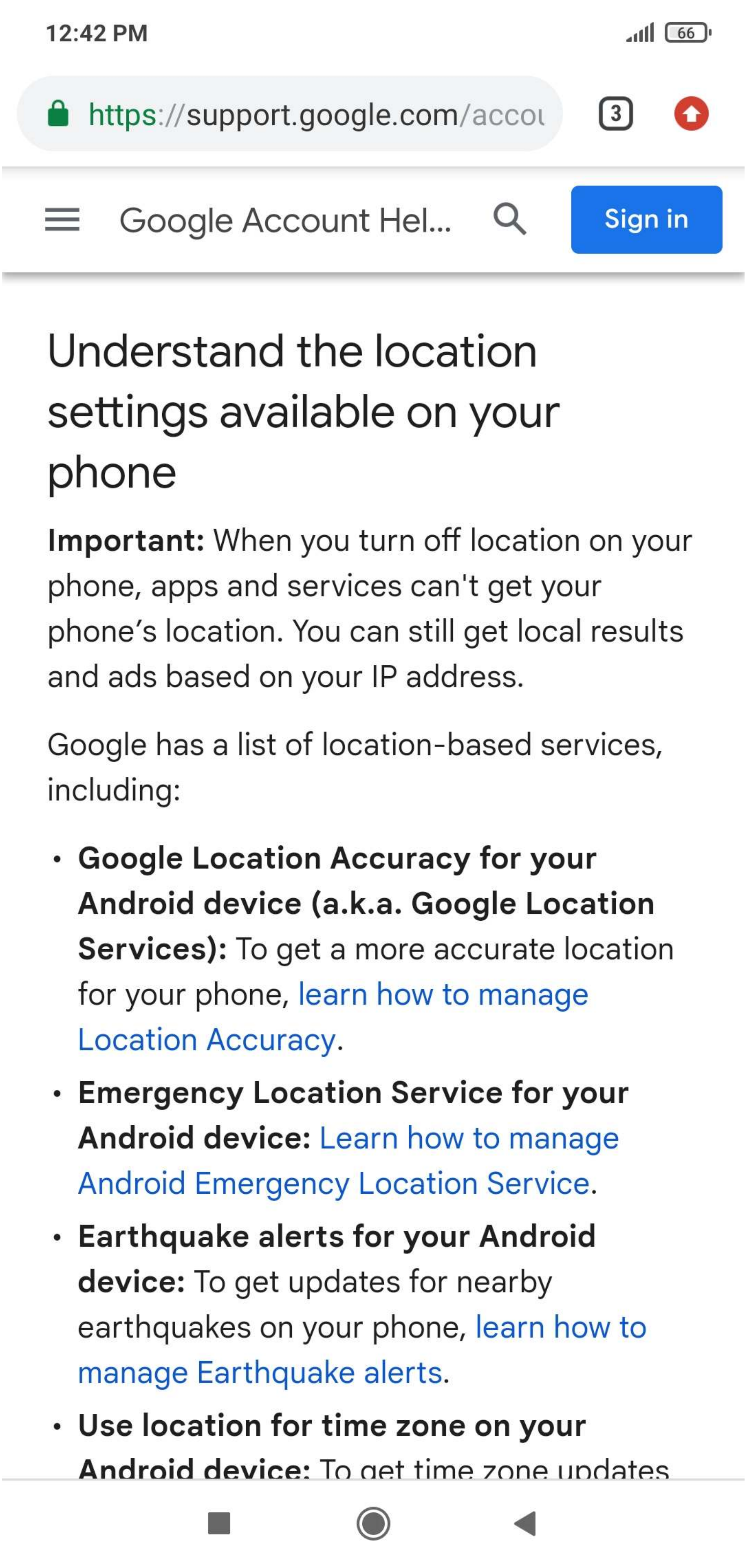
You can use location-based services such as getting better local search results, like commute predictions and nearby restaurants based on your phone’s location, when you turn location on in settings.

Important:

- Some of these steps work only on Android 12 and up. Learn how to check your Android version.
- Some of these steps require you to touch the screen.

Understand the location settings available on your phone

Important: When you turn off location or phone, apps and services can't get your phone’s location. You can still get local results


12:42 PM

https://support.google.com/accou

Google Account Hel...

Sign in

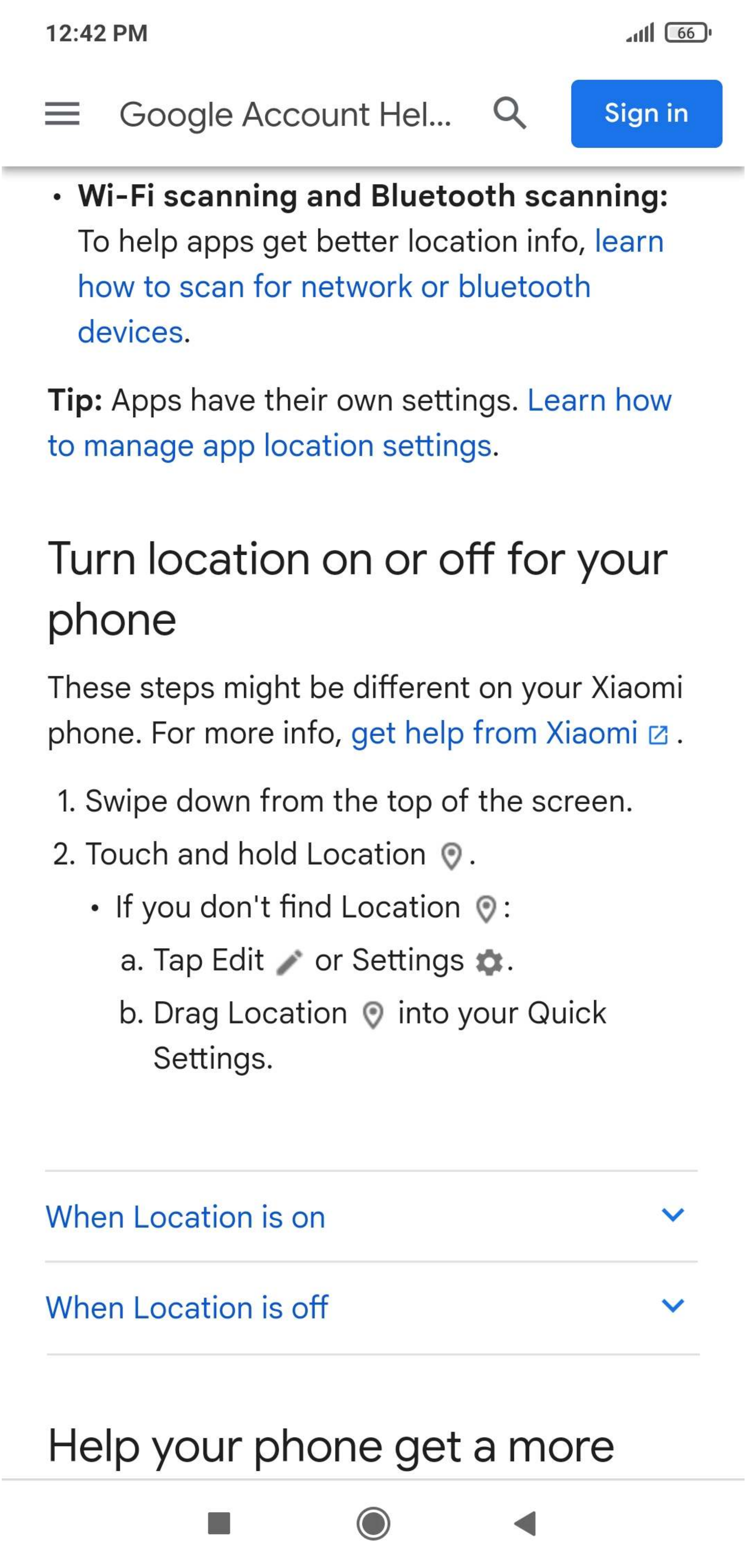
Understand the location settings available on your phone

Important: When you turn off location on your phone, apps and services can't get your phone’s location. You can still get local results and ads based on your IP address.

Google has a list of location-based services, including:

- **Google Location Accuracy for your Android device (a.k.a. Google Location Services):** To get a more accurate location for your phone, learn how to manage Location Accuracy.
- **Emergency Location Service for your Android device:** Learn how to manage Android Emergency Location Service.
- **Earthquake alerts for your Android device:** To get updates for nearby earthquakes on your phone, learn how to manage Earthquake alerts.
- **Use location for time zone on your Android device:** To get time zone updates

- **Earthquake alerts for your Android device:** To get updates for nearby earthquakes on your phone, learn how to manage Earthquake alerts.
- **Use location for time zone on your Android device:** To get time zone updates based on location, learn how to manage location for time zone.
- **Location History for your Google Account:** Location History is a Google Account setting that creates Timeline, a personal map that helps you remember places you've been, and routes and trips you've taken. Learn how to turn on Location History.
- **Location Sharing for Google Maps:** To let others know where your phone is, learn how to share your real-time location via Google Maps.
- **Location in Search:** To get more helpful results when you search on Google, learn how to manage location permissions for websites and apps.
- **Wi-Fi scanning and Bluetooth scanning:** To help apps get better location info, learn how to scan for network or bluetooth devices.



- **Wi-Fi scanning and Bluetooth scanning:** To help apps get better location info, learn how to scan for network or bluetooth devices.

Tip: Apps have their own settings. Learn how to manage app location settings.

Turn location on or off for your phone

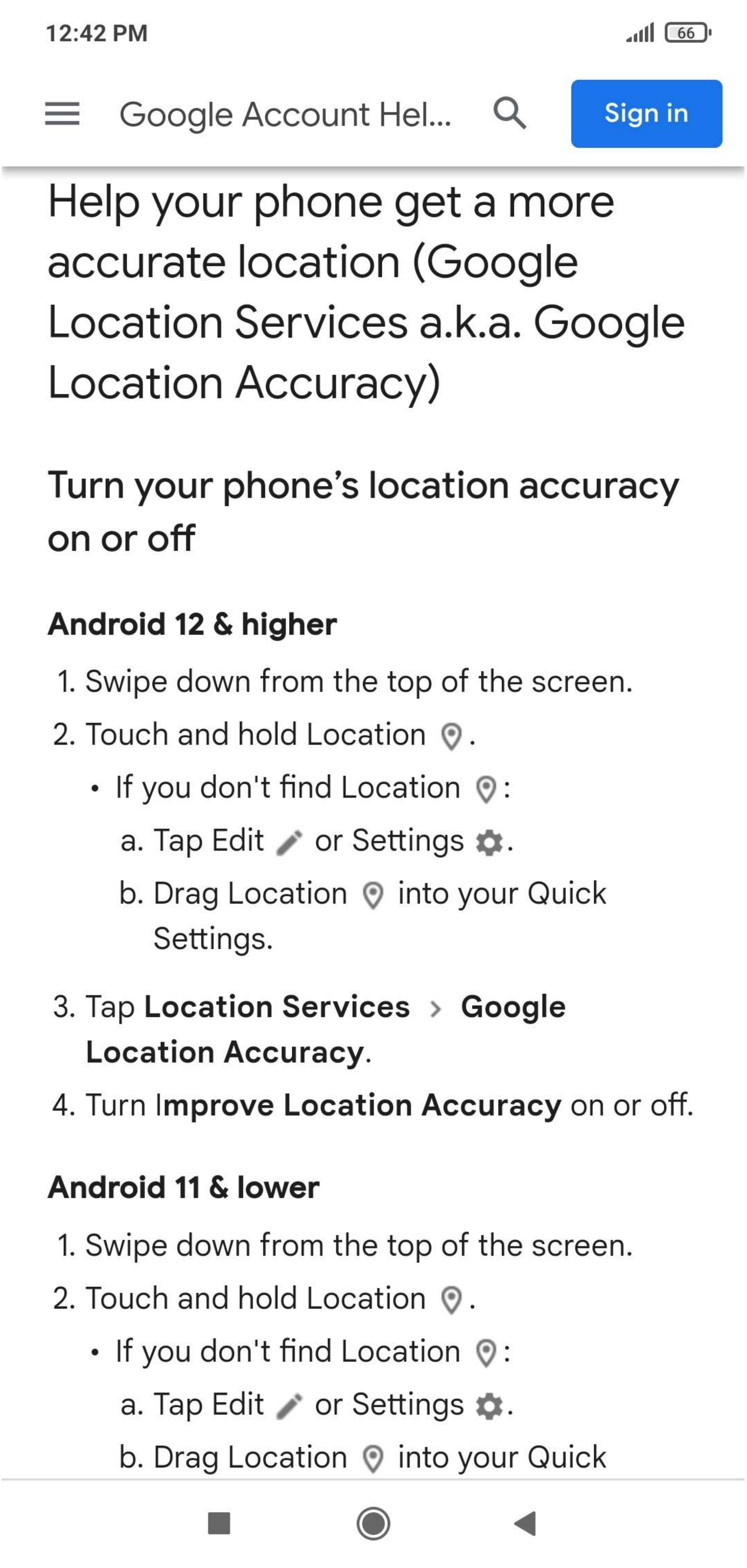
These steps might be different on your Xiaomi phone. For more info, get help from Xiaomi .

1. Swipe down from the top of the screen.
2. Touch and hold Location .
   - If you don't find Location :
     a. Tap Edit or Settings .
     b. Drag Location into your Quick Settings.

When Location is on

When Location is off

Help your phone get a more



12:42 PM 66

Google Account Hel... Sign in

Help your phone get a more accurate location (Google Location Services a.k.a. Google Location Accuracy)

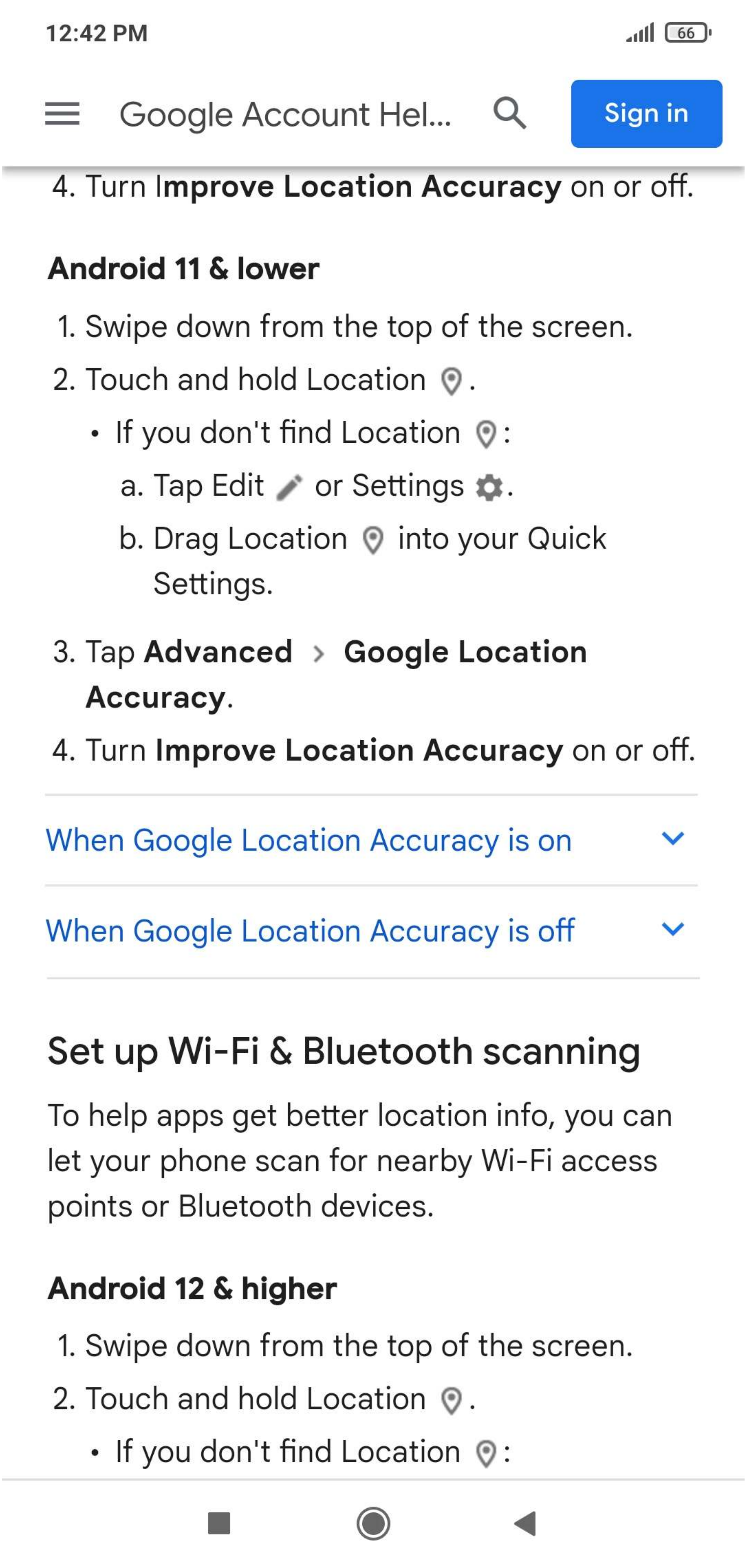
Turn your phone's location accuracy on or off

Android 12 & higher

1. Swipe down from the top of the screen.
2. Touch and hold Location .
   - If you don't find Location :
     a. Tap Edit or Settings .
     b. Drag Location into your Quick Settings.
3. Tap **Location Services** › **Google Location Accuracy**.
4. Turn I**mprove Location Accuracy** on or off.

Android 11 & lower

1. Swipe down from the top of the screen.
2. Touch and hold Location .
   - If you don't find Location :
     a. Tap Edit or Settings .
     b. Drag Location into your Quick



4. Turn **Improve Location Accuracy** on or off.

Android 11 & lower

1. Swipe down from the top of the screen.
2. Touch and hold Location .
   - If you don't find Location :
     a. Tap Edit or Settings .
     b. Drag Location into your Quick Settings.
3. Tap **Advanced** › **Google Location Accuracy**.
4. Turn **Improve Location Accuracy** on or off.

When Google Location Accuracy is on

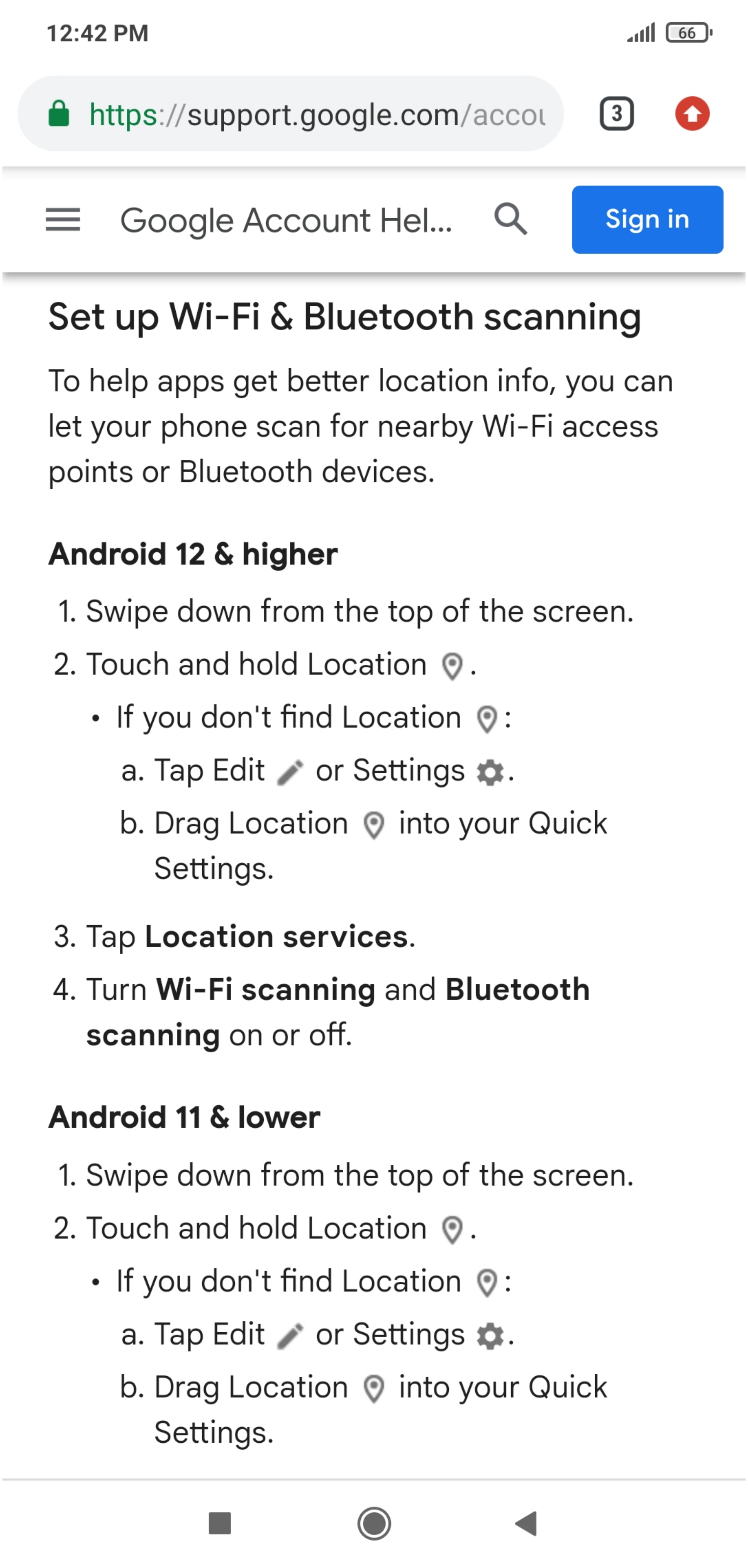
When Google Location Accuracy is off

Set up Wi-Fi & Bluetooth scanning

To help apps get better location info, you can let your phone scan for nearby Wi-Fi access points or Bluetooth devices.

Android 12 & higher

1. Swipe down from the top of the screen.
2. Touch and hold Location .
   - If you don't find Location :



Set up Wi-Fi & Bluetooth scanning

To help apps get better location info, you can let your phone scan for nearby Wi-Fi access points or Bluetooth devices.

Android 12 & higher

1. Swipe down from the top of the screen.
2. Touch and hold Location.
   - If you don't find Location:
     a. Tap Edit or Settings.
     b. Drag Location into your Quick Settings.
3. Tap **Location services**.
4. Turn **Wi-Fi scanning** and **Bluetooth scanning** on or off.

Android 11 & lower

1. Swipe down from the top of the screen.
2. Touch and hold Location.
   - If you don't find Location:
     a. Tap Edit or Settings.
     b. Drag Location into your Quick Settings.

Settings.

3. Tap **Wi-Fi scanning** and **Bluetooth scanning**.
4. Turn **Wi-Fi scanning** and **Bluetooth scanning** on or off.

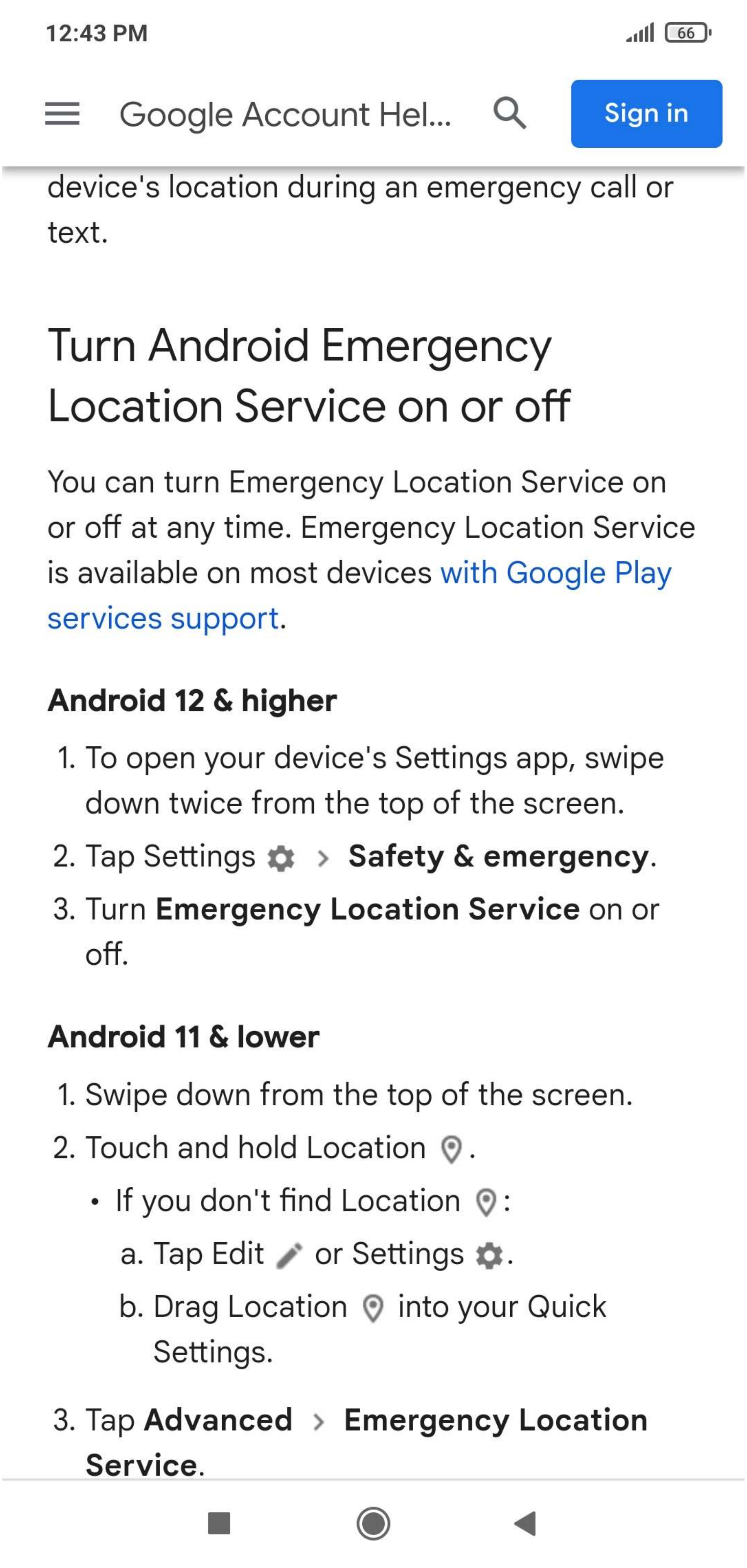
Send your location in an emergency

To help responders find you quickly, your phone's location can be sent when you dial or text an emergency number, like when you dial 911 in the US or 112 in Europe.

If Android Emergency Location Service (ELS) works in your country or region and on your mobile network, and you haven't turned ELS off, your phone will automatically send its location to first responders through ELS. If ELS is off, your mobile carrier might still send the device's location during an emergency call or text.

Turn Android Emergency Location Service on or off

You can turn Emergency Location Service on or off at any time. Emergency Location Service



device's location during an emergency call or text.

Turn Android Emergency Location Service on or off

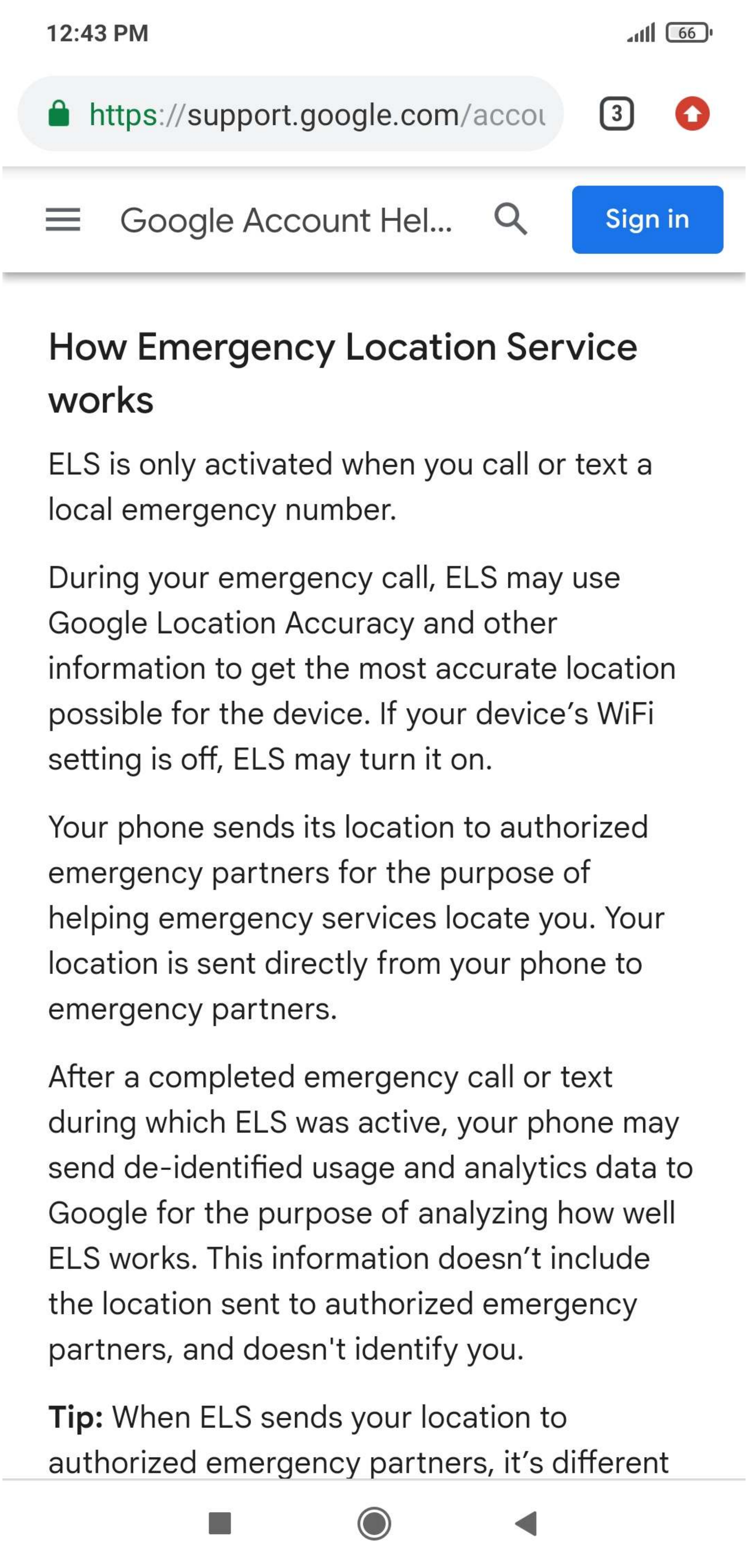
You can turn Emergency Location Service on or off at any time. Emergency Location Service is available on most devices with Google Play services support.

Android 12 & higher

1. To open your device's Settings app, swipe down twice from the top of the screen.
2. Tap Settings > **Safety & emergency**.
3. Turn **Emergency Location Service** on or off.

Android 11 & lower

1. Swipe down from the top of the screen.
2. Touch and hold Location .
   - If you don't find Location :
     a. Tap Edit or Settings .
     b. Drag Location into your Quick Settings.
3. Tap **Advanced** > **Emergency Location Service**.



How Emergency Location Service works

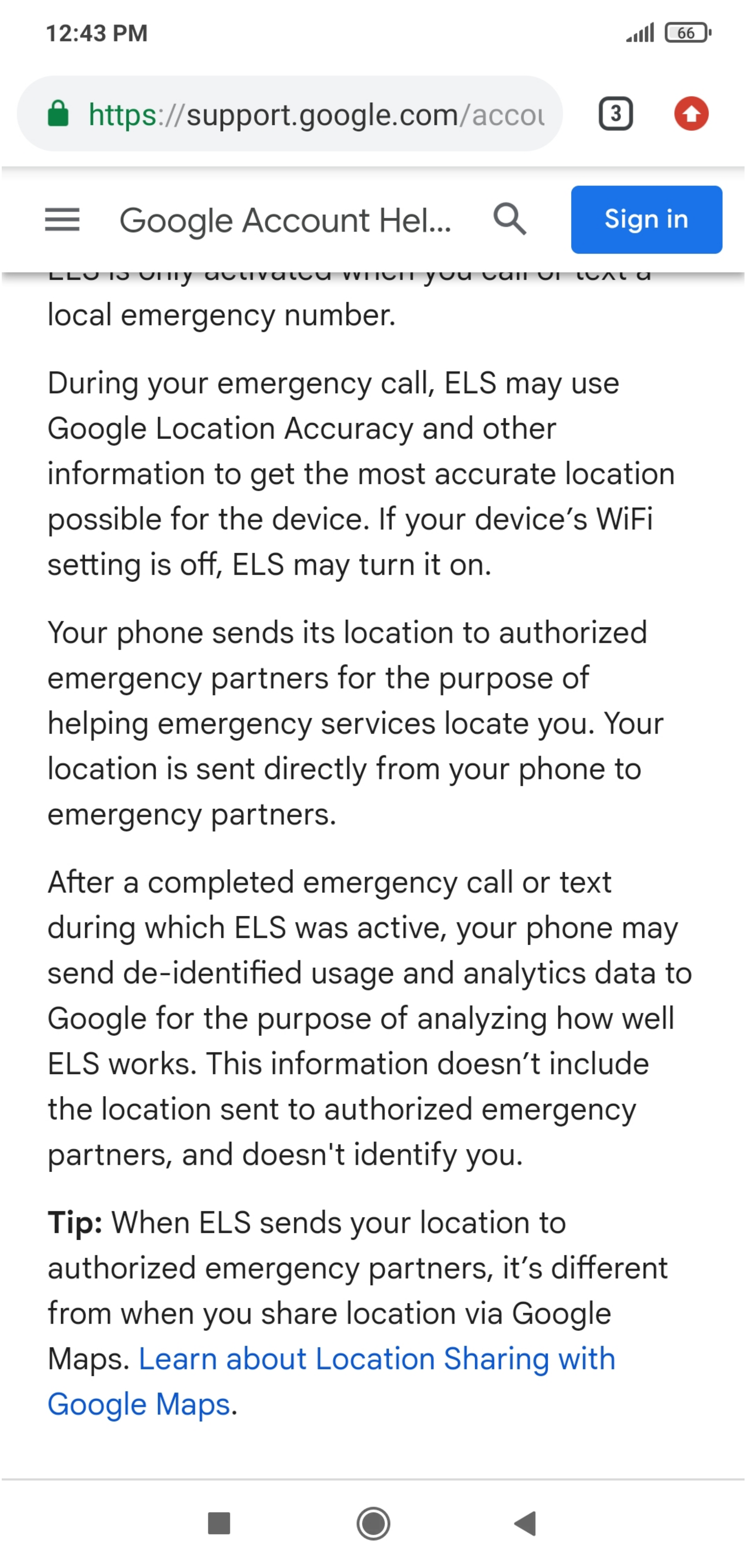
ELS is only activated when you call or text a local emergency number.

During your emergency call, ELS may use Google Location Accuracy and other information to get the most accurate location possible for the device. If your device's WiFi setting is off, ELS may turn it on.

Your phone sends its location to authorized emergency partners for the purpose of helping emergency services locate you. Your location is sent directly from your phone to emergency partners.

After a completed emergency call or text during which ELS was active, your phone may send de-identified usage and analytics data to Google for the purpose of analyzing how well ELS works. This information doesn't include the location sent to authorized emergency partners, and doesn't identify you.

Tip: When ELS sends your location to authorized emergency partners, it's different



ELS is only activated when you call or text a local emergency number.

During your emergency call, ELS may use Google Location Accuracy and other information to get the most accurate location possible for the device. If your device's WiFi setting is off, ELS may turn it on.

Your phone sends its location to authorized emergency partners for the purpose of helping emergency services locate you. Your location is sent directly from your phone to emergency partners.

After a completed emergency call or text during which ELS was active, your phone may send de-identified usage and analytics data to Google for the purpose of analyzing how well ELS works. This information doesn't include the location sent to authorized emergency partners, and doesn't identify you.

Tip: When ELS sends your location to authorized emergency partners, it's different from when you share location via Google Maps. Learn about Location Sharing with Google Maps.