

宝贵建议请发送至：wangzhenyang@itcast.cn



黑马程序员

itheima.com

-编程，始于黑马

Android 课程同步笔记

Alpha 0.01 版

By 阳哥

Android 手机卫士-04

1. 号码归属地查询 (★★)

在功能列表中点击高级工具，就进入了我们的高级工具界面，该界面主要有号码归属地查询、短信备份、常用号码查询、程序锁等功能。那么本文档中我们需要实现的是号码归属地查询功能。号码归属地查询界面比较简单，如下图所示。



1.1 高级工具、号码归属地查询 UI 的实现

步骤：1、创建 atools_activity.xml 布局文件

```
<?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:orientation="vertical"
>
<TextView
    android:layout_width="match_parent"
    android:layout_height="55dp"
    android:background="#8866ff00"
    android:gravity="center"
    android:text="高级工具"
    android:textColor="#000000"
    android:textSize="20sp" />
<TextView
    android:background="@drawable/button_select_contact"
    android:gravity="center_vertical"
    android:text="号码归属地查询"
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:drawableLeft="@android:drawable/ic_menu_camera"
    android:textSize="18sp"
    android:textColor="#000000"
    android:onClick="numberQuery"
    android:clickable="true"
    />
<ProgressBar
    android:id="@+id/progressBar1"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
<TextView
    android:background="@drawable/button_select_contact"
    android:gravity="center_vertical"
    android:text="短信备份"
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:drawableLeft="@android:drawable/ic_menu_camera"
    android:textSize="18sp"
    android:textColor="#000000"
    android:onClick="smsBackup"
    android:clickable="true"
    />
<TextView
    android:background="@drawable/button_select_contact"
    android:gravity="center_vertical"
    android:text="常用号码查询"
```

```
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:drawableLeft="@android:drawable/ic_menu_camera"
        android:textSize="18sp"
        android:textColor="#000000"
        android:onClick="commonNumberQuery"
        android:clickable="true"
    />
<TextView
    android:background="@drawable/button_select_contact"
    android:gravity="center_vertical"
    android:text="程序锁"
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:drawableLeft="@android:drawable/ic_menu_camera"
    android:textSize="18sp"
    android:textColor="#000000"
    android:onClick="entryAppLock"
    android:clickable="true"
/>
</LinearLayout>
```

2、编写 AToolsActivity 类，同时在 AndroidManifest.xml 中注册该 Activity。

```
public class AToolsActivity extends Activity {
    private ProgressBar progressBar1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.atools_activity);
        progressBar1 = (ProgressBar) findViewById(R.id.progressBar1);
    }
    // 号码归属地查询
    public void numberQuery(View view) {
        Intent intent = new Intent(this, AddressQueryActivity.class);
        startActivity(intent);
    }
    //短信备份功能
    public void smsBackup(View view) {
        final File file = new File(Environment.getExternalStorageDirectory(),
"smsbackup.xml");
```

```
final ProgressBar progressBar = new ProgressBar(this);
new Thread(new Runnable() {

    @Override
    public void run() {
        if
(Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED)) {
            SmsBackupUtil.smsBackup(AToolsActivity.this,
file.getAbsolutePath(), progressBar1);
            Looper.prepare();
            Toast.makeText(AToolsActivity.this, "短信备份成功", 0).show();
            Looper.loop();
        }
    }
}).start();
}
//打开常用号码查询界面
public void commonNumberQuery(View view){
    Intent intent = new Intent(this, CommonNumberQueryActivity.class);
    startActivity(intent);
}
//打开程序锁界面
public void entryAppLock(View view){
    Intent intent = new Intent(this, AppLockActivity.class);
    startActivity(intent);
}
}
```

3、创建号码归属地查询的 number_address_query_activity.xml 布局文件

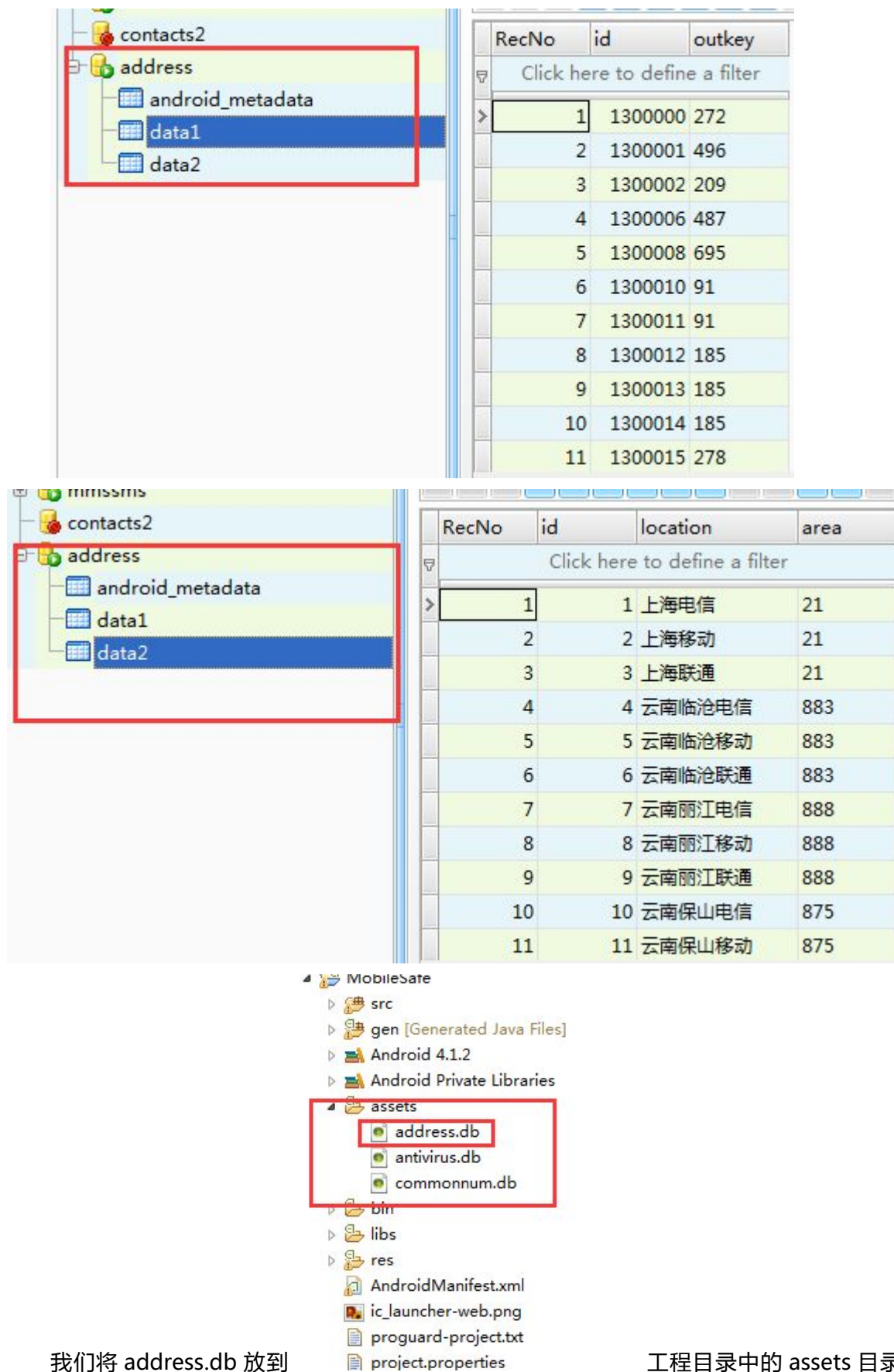
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="55dp"
        android:background="#8866ff00"
        android:gravity="center"
        android:text="号码归属地查询"
```

```
        android:textColor="#000000"
        android:textSize="20sp" />
    <EditText
        android:id="@+id/et_phoneNumber"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:hint="请输入号码"
        android:inputType="number"
    />
    <Button
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:text="查询"
        android:onClick="query"
    />
    <TextView
        android:id="@+id/tv_addressResult"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="查询结果"
    />
</LinearLayout>
```

1.2 号码归属地查询代码实现

实现号码归属地查询有多种方法，比如联网查询，通过可以调用互联网服务，还有就是将常用的电话号码前缀存放在数据库库，然后将该数据库保存在本地。在我们的文档中我们只演示如何从数据库中进行归属地的查询。

为了演示，我们提供了一个手机号码号码归属地数据库 address.db。用 SQLite Expert Professional 工具打开该数据库，截图如下。该数据库有两张表，data1 和 data2，其中 data1 保存的是电话号码前缀和归属地编号。data2 保存的是地区信息和归属地编号。



The top screenshot shows a database table with the following data:

RecNo	id	outkey
1	1300000	272
2	1300001	496
3	1300002	209
4	1300006	487
5	1300008	695
6	1300010	91
7	1300011	91
8	1300012	185
9	1300013	185
10	1300014	185
11	1300015	278

The middle screenshot shows a database table with the following data:

RecNo	id	location	area
1	1	上海电信	21
2	2	上海移动	21
3	3	上海联通	21
4	4	云南临沧电信	883
5	5	云南临沧移动	883
6	6	云南临沧联通	883
7	7	云南丽江电信	888
8	8	云南丽江移动	888
9	9	云南丽江联通	888
10	10	云南保山电信	875
11	11	云南保山移动	875

The bottom screenshot shows a file explorer with the following structure:

- MobileSafe
 - src
 - gen [Generated Java Files]
 - Android 4.1.2
 - Android Private Libraries
 - assets
 - address.db
 - antivirus.db
 - commonnum.db
 - bin
 - libs
 - res
 - AndroidManifest.xml
 - ic_launcher-web.png
 - proguard-project.txt
 - project.properties

我们将 address.db 放到

工程目录中的 assets 目录下。在 SplashActivity

类中，我们将该资源拷贝到/data/data/com.itheima.mobileSafe/files/address.db 中。拷贝代码清单如下：

```
private void copyDB(String name) {
    InputStream inputStream = null;
    FileOutputStream outputStream = null;
    try {
        inputStream = getAssets().open(name);
        File file = new File(getFilesDir(), name);
        if (file.exists()) {
            Toast.makeText(this, "数据已经初始化完成", Toast.LENGTH_SHORT).show();
            return;
        }
        outputStream = new FileOutputStream(file);
        int len = -1;
        byte[] buff = new byte[1024];
        while((len=inputStream.read(buff))!=-1){
            outputStream.write(buff, 0, len);
        }
        inputStream.close();
        outputStream.close();
    } catch (IOException e) {
        e.printStackTrace();
    }finally{
        if (inputStream!=null) {
            try {
                inputStream.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        if (outputStream!=null) {
            try {
                outputStream.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

在 com.itheima.mobileSafe.db.dao 包（如果没有该包则创建）下创建 AddressQueryDao 类，我们在该类中实现对数据库的操作。代码清单如下：


```
public class AddressQueryDao {
    private static String path = "/data/data/com.itheima.mobileSafe/files/address.db";
    public static String queryAddress(String number) {
        if (number.length() < 7) {
            return "来自服务"+number;
        }
        SQLiteDatabase database = SQLiteDatabase.openDatabase(path, null,
        SQLiteDatabase.OPEN_READONLY);
        String sql = "select b.id,a.location from data1 as b left join data2 as a on
a.area=b.outKey where b.id='"+number.substring(0, 7)+'' ";
        Cursor cursor = database.rawQuery(sql, null);
        String location = null;
        while(cursor.moveToNext()){
            location = cursor.getString(0);
            break;
        }
        cursor.close();
        return location;
    }
}
```

编写 AddressQueryActivity 类。

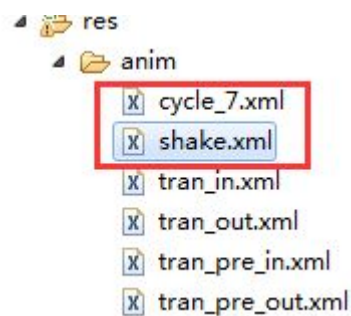
```
public class AddressQueryActivity extends Activity {
    private EditText et_phoneNumber;
    private TextView tv_addressResult;
    private Vibrator vibrator;
    Handler handler = new Handler() {
        public void handleMessage(android.os.Message msg) {
            if (msg.what == RESULT_OK) {
                tv_addressResult.setText(msg.obj.toString());
            }
        };
    };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.number_address_query_activity);
        vibrator = (Vibrator) getSystemService(VIBRATOR_SERVICE);
        et_phoneNumber = (EditText) findViewById(R.id.et_phoneNumber);
        tv_addressResult = (TextView) findViewById(R.id.tv_addressResult);
        //给 EditText 对象添加文本变化监听
    }
}
```

```
et_phoneNumber.addTextChangedListener(new TextWatcher() {
    //当文本内容变化时调用该方法
    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count)
    {
        if (s.length() >= 7) {
            query(null);
        } else {
            tv_addressResult.setText("");
        }
    }
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int
after) {
    }
    @Override
    public void afterTextChanged(Editable s) {
    }
});
}
//执行查询方法
public void query(View view) {
    String phone = et_phoneNumber.getText().toString();
    if (TextUtils.isEmpty(phone)) {
        Toast.makeText(this, "号码不能为空", Toast.LENGTH_SHORT).show();
        //如果号码为空，则抖动编辑框，抖动效果是在 xml 文件中定义的
        Animation shake = AnimationUtils.loadAnimation(this, R.anim.shake);
        et_phoneNumber.startAnimation(shake);
        vibrator.vibrate(2000);
        return;
    } else if (phone.length() < 7) {
        Toast.makeText(this, "号码至少为 7 位", Toast.LENGTH_SHORT).show();
        Animation shake = AnimationUtils.loadAnimation(this, R.anim.shake);
        et_phoneNumber.startAnimation(shake);
        return;
    }
    String reg = "^1[345678]\\d{5,9}$";
    if (!phone.matches(reg)) {
        Toast.makeText(this, "输入的电话号码不合法。", Toast.LENGTH_SHORT).show();
        Animation shake = AnimationUtils.loadAnimation(this, R.anim.shake);
        et_phoneNumber.startAnimation(shake);
        return;
    }
}
```

```
String address = AddressQueryDao.queryAddress(phone);
Message msg = new Message();
msg.what = RESULT_OK;
if (TextUtils.isEmpty(address)) {
    msg.obj = "对不起，没有查询到该号码";
} else {
    msg.obj = address;
}
handler.sendMessage(msg);
}
```

在上面代码中当用户没有输入任何内容却要查询号码时，我们给了用户抖动编辑框的提示。这个抖动效果主要是通过动画实现的。代码很简单，只有如下的两行。

```
Animation shake = AnimationUtils.loadAnimation(this, R.anim.shake);
et_phoneNumber.startAnimation(shake);
```



我们需要在 anim 目录中创建 shake.xml 文件。文件目录结构如左图所示。

shake.xml 文件清单如下：

```
<?xml version="1.0" encoding="utf-8"?>
<translate xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="1000"
    android:fromXDelta="0"
    android:interpolator="@anim/cycle_7"
    android:toXDelta="10" />
```

上面的代码中用到了 android:interpolator 属性，该属性在这里代表当前动画执行的次数，cycle_7.xml 文件清单如下：

```
<?xml version="1.0" encoding="utf-8"?>
<cycleInterpolator xmlns:android="http://schemas.android.com/apk/res/android"
    android:cycles="7" />
```

2.手机震动效果 (★★)

当用户输入某些内容不合法时我们可以通过手机震动效果来提示用户，那么手机震动效果怎么实现呢？

步骤：

1、在 AndroidManifest.xml 中添加权限

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

2、在代码中声明 Vibrator 对象

```
private Vibrator vibrator;
```

3、获取 vibrator 实例

```
vibrator = (Vibrator) getSystemService(VIBRATOR_SERVICE);
```

4、调用 vibrator 的 vibrate 方法，参数代表震动时长（毫秒）

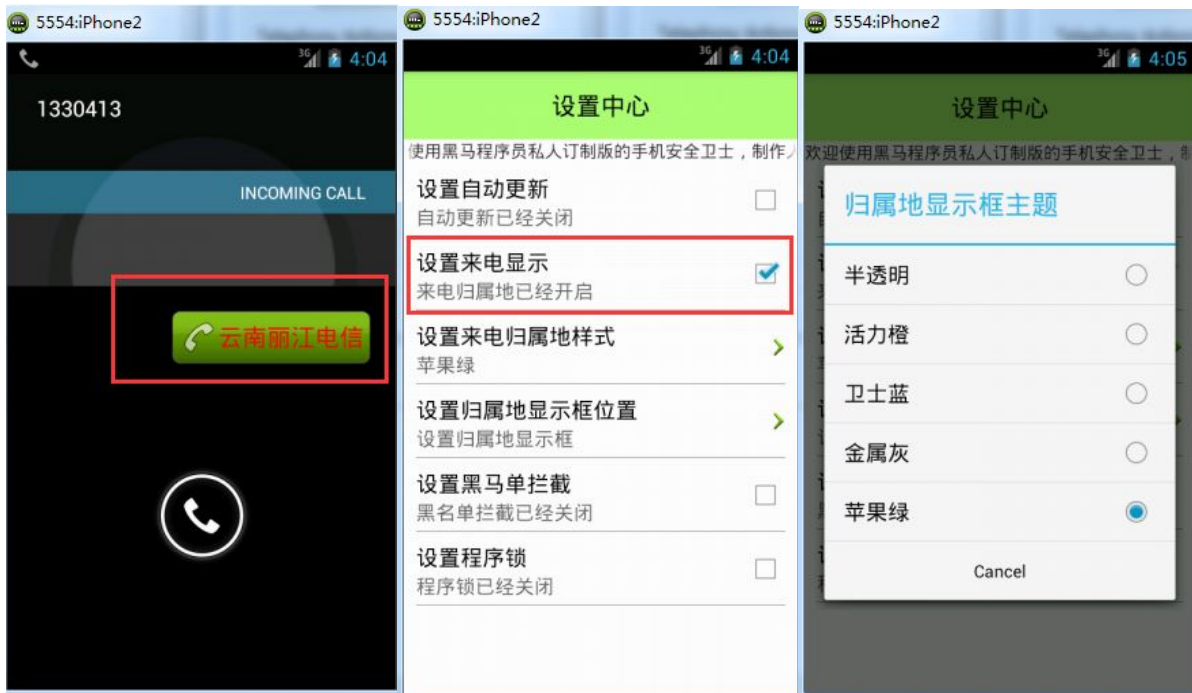
```
vibrator.vibrate(500);  
  
//-1不重复，非-1为从pattern的指定下标开始重复  
long[] pattern = {100,200,100,200,50,50};  
vibrator.vibrate(pattern, 1);
```

在上面代码中 vibrate 方法有两种方法重载形式，第二种比较难理解，pattern 是一个数组，存放的是震动时长（毫秒），第二个参数代表从 pattern 数组的脚标位置开始依次震动数组中的时长。

3.来电号码归属地显示 (★★)

打开手机卫士的设置中心，添加 3 条功能设置项，设置来电显示、设置来电归属地样式、设置归属地显示框位置。

勾选上设置来电显示后当有电话拨打进来手机界面会显示归属地，该归属地显示在一个控件中，该控件可以拖动，并且在设置来电归属地样式中可以设置该控件的主题。效果图如下：



1、编辑设置中心布局文件 setting_activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:itheima="http://schemas.android.com/apk/res/com.itheima.mobileSafe"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="match_parent"
        android:layout_height="55dp"
        android:background="#8866ff00"
        android:gravity="center"
        android:text="设置中心"
        android:textColor="#000000"
        android:textSize="20sp" />

    <com.itheima.mobileSafe.ui.SettingItemView
        android:id="@+id/siv_update"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        itheima:desc_off="自动更新已经关闭"
        itheima:desc_on="自动更新已经开启"
        itheima:title="设置自动更新" />
```

```
<com.itheima.mobileSafe.ui.SettingItemView
    android:id="@+id/siv_showAddress"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    itheima:desc_off="来电归属地已经关闭"
    itheima:desc_on="来电归属地已经开启"
    itheima:title="设置来电显示" />

<com.itheima.mobileSafe.ui.SettingClickView
    android:id="@+id/scv_change_bg"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    itheima:desc_off="来电归属地样式为设置"
    itheima:desc_on="来电归属地已经开启"
    itheima:title="设置来电归属地样式" />

<com.itheima.mobileSafe.ui.SettingClickView
    android:id="@+id/scv_change_position"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    itheima:desc_off="来电归属地样式为设置"
    itheima:desc_on="来电归属地已经开启"
    itheima:title="设置来电归属地位置" />

<com.itheima.mobileSafe.ui.SettingItemView
    android:id="@+id/siv_blackNumber"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    itheima:desc_off="黑名单拦截已经关闭"
    itheima:desc_on="黑名单拦截已经开启"
    itheima:title="设置黑名单拦截" />
<com.itheima.mobileSafe.ui.SettingItemView
    android:id="@+id/siv_appLock"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    itheima:desc_off="程序锁已经关闭"
    itheima:desc_on="程序锁已经开启"
    itheima:title="设置程序锁" />
</LinearLayout>
```

注意：上面的布局文件我给出了所有的设置中心的布局内容，但是具体的代码实现会在以后的文档中实现。在本文档中我们只实现来电号码归属地显示功能。

2、创建 SettingActivity 类，对应 setting_activity.xml 布局文件。

3、在类中声明来电显示控件

```
//设置来电显示
private SettingItemView siv_showAddress;
```

4、在 onCreate 方法中，初始化 siv_showAddress 控件，同时判断来电显示服务是否已经开启，如果开启则将 checkbox 勾选上，否则不勾选。

```
//获取设置来电显示控件
siv_showAddress = (SettingItemView) findViewById(R.id.siv_showAddress);
//获取来电显示服务是否开启
boolean running = ServiceStatusUtil.isRunning(this,
AddressService.class.getName());
    if (running) {
        //如果开启则将 checkbox 勾选上
        siv_showAddress.setChecked(true);
    } else {
        siv_showAddress.setChecked(false);
    }
```

5、创建用于判断服务是否启动的工具类 ServiceStatusUtil，该类代码清单如下：

```
public class ServiceStatusUtil {
    public static boolean isRunning(Context context,String name){
        //获取 ActivityManager 对象
        ActivityManager activityManager = (ActivityManager)
context.getSystemService(Context.ACTIVITY_SERVICE);
        //通过 activityManager 对象获取正在运行的 service，参数代表获取的最大个数
        List<RunningServiceInfo> runningServices =
activityManager.getRunningServices(100);
        for(RunningServiceInfo info : runningServices){
            System.out.println("当前开启的服务有: "+info.service.getClassName()+"--- 目
标服务"+name);
            //如果服务的名称等于我们指定的名称则代表服务正在运行
            if (info.service.getClassName().equals(name)) {
                return true;
            }
        }
        return false;    }
```


6、在 SettingActivity 的 onCreate 方法中给 showAddress 绑定点击事件。

```
//来电显示服务意图
showAddressIntent = new Intent(this, AddressService.class);
//给设置来电显示控件设置点击监听事件
siv_showAddress.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        if (siv_showAddress.isChecked()) {
            siv_showAddress.setChecked(false);
            stopService(showAddressIntent);
        } else {
            siv_showAddress.setChecked(true);
            //开启来电显示服务
            startService(showAddressIntent);
        }
    }
}
```

7、编写 AddressService 类，在该类中实现了具体的电话归属地服务。

7.1 声明成员变量

```
//声明 TelephonyManager 对象
private TelephonyManager tm;
//声明自定义的监听器类继承了 API 的 PhoneStateListener
private MyPhoneStateListener listener;
```

7.2 编写 MyPhoneStateListener 类，继承 PhoneStateListener 类

```
class MyPhoneStateListener extends PhoneStateListener {
    @Override
    public void onCallStateChanged(int state, String incomingNumber) {
        super.onCallStateChanged(state, incomingNumber);
        switch (state) {
            case TelephonyManager.CALL_STATE_RINGING:// 电话打进来了
                //获取打进来的电话，然后查询归属地
                String address = AddressQueryDao.queryAddress(incomingNumber);
                //显示归属地
                Toast.makeText(getApplicationContext(), address, 1).show();
                myToast(address);
                break;
        }
    }
}
```

```
        case TelephonyManager.CALL_STATE_IDLE:// 电话被挂断了才空闲
            if (view != null) {
                wm.removeView(view);
                view = null;
            }
            break;
        default:
            break;
    }
}
```

注意：上面的上面的 wm 对象和 myToast 方法用到了自定义吐司的知识，这块知识在下一个文档中会作详细的讲解。

8、在 AddressService 类的 onCreate 中实现监听。

```
//获取 TelephonyManager 服务对象
tm = (TelephonyManager) getSystemService(TELEPHONY_SERVICE);
// 创建一个 PhoneStateListener 对象
listener= new MyPhoneStateListener();
//监听电话状态
tm.listen(listener, PhoneStateListener.LISTEN_CALL_STATE);
```

4.去电号码归属地显示 (★★)

在本文档上一节的基础上，我们在设置来电归属地显示的同时也同时打开了去电归属地的显示。去电归属地查询主要是通过监听系统拨号广播实现的，因此需要我们定义一个广播接收者接收系统发出的拨号广播。

步骤：1、在上一节中的 AddressService 类中声名一个自定义广播接收者对象。

```
//声明去电广播接收者
private OutCallReceiver receiver;
```

2、在 AddressService 类中定义 OutCallReceiver 类继承 BroadcastReceiver 类

// 服务里面的内部类-监听去电

```
class OutCallReceiver extends BroadcastReceiver {  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        String number = getResultData();  
        String address = AddressQueryDao.queryAddress(number);  
        // Toast.makeText(context, address, 1).show();  
        myToast(address);  
    }  
}
```

3、在 AddressService 类的 onCreate 方法中注册去电监听

```
//创建去电的意图过滤器  
IntentFilter filter = new IntentFilter();  
filter.setPriority(1000);  
filter.addAction("android.intent.action.NEW_OUTGOING_CALL");  
// 代码注册广播接收者-监听去电  
registerReceiver(receiver, filter);
```

至此，本文档完！

2015 年 1 月 6 日 星期二 15:53:25

北京市中关村软件园国际软件大厦