Due:3/26/2014 http://npu85.npu.edu/~henry/npu/classes/android/db/slide/exercise_db.html Q5 Q6

Development Environment: OS: Windows 7 x64 // IDE: ADT v22.3.0-887826 // AVD: **Testing API-16** SKIN-320*480

Question5: Tab Activity and Database

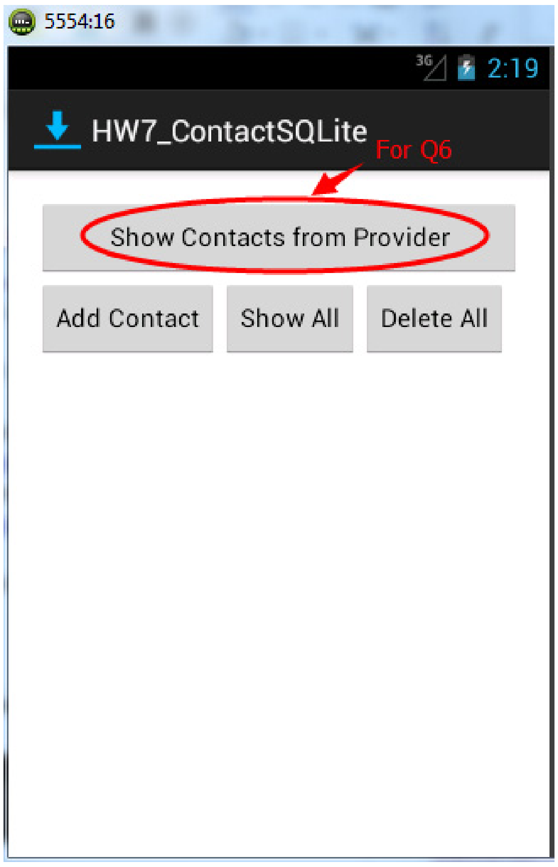
- Continue the Tab Activity so that the data entered by the users will be saved in a ***data base***.

Question6: Tab Activity and Content Provider/Content Resolver

- Continue the Tab Activity so that the data entered by the users will be saved in a data base. The database is then used as a ***content provider***.
- An Activity will then act as a ***content resolver*** to interact with the content provider to retrieve the data from the database.

Q5 Solution:

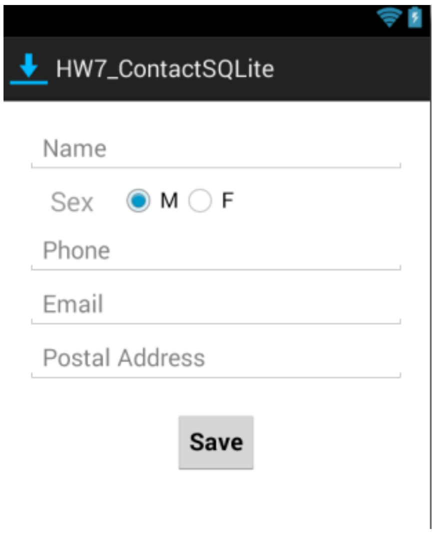
1. Create a new project: HW7_ContactsSQLite
Create main *layout* resource: contact_tester.xml



Root: Linearlayout

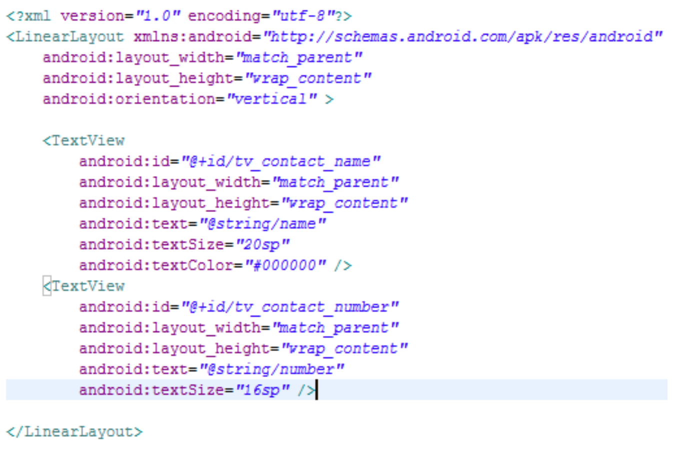
```
<Button
    android:id="@+id/bt_provider"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:onClick="showProvider"
    android:text="@string/show_provider"
    android:textSize="15sp" />
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >
    <Button
        android:id="@+id/bt_andcontact"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="goAndcontact"
        android:text="@string/and_contact"
        android:textSize="15sp" />
    <Button
        android:id="@+id/bt_show"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="showAll"
        android:text="@string/show_contacts"
        android:textSize="15sp" />
    <Button
        android:id="@+id/bt_delete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="deleteAll"
        android:text="@string/delete"
        android:textSize="15sp" />
</LinearLayout>
<View
    android:layout_width="match_parent"
    android:layout_height="20dp" />
<ListView
    android:id="@+id/lv1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
</ListView>
```

2.Create contact form *layout* resource: contacts.xml



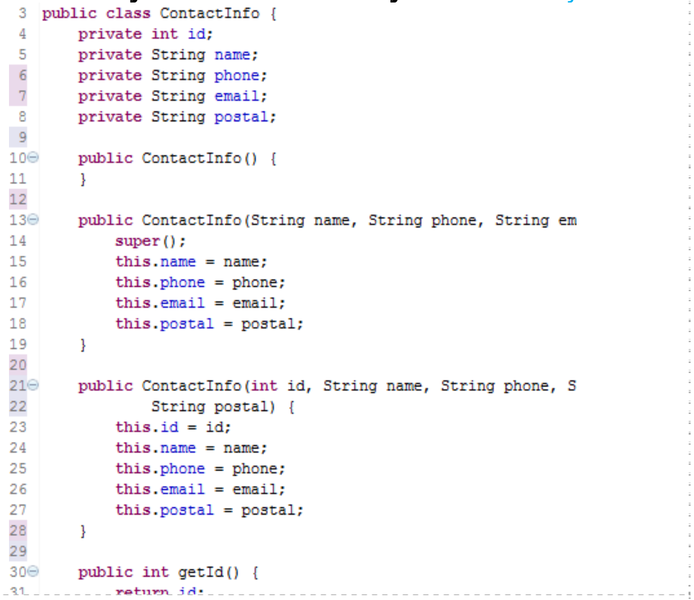
```
<Button
    android:id="@+id/bt_save"
    android:onClick="clickSave"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/bt_save"
    android:textStyle="bold" />
```

3. Create item *layout* resource for data adapter: contact_item.xml



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

    <TextView
        android:id="@+id/tv_contact_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/name"
        android:textSize="20sp"
        android:textColor="#000000" />
    <TextView
        android:id="@+id/tv_contact_number"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/number"
        android:textSize="16sp" />

</LinearLayout>
```

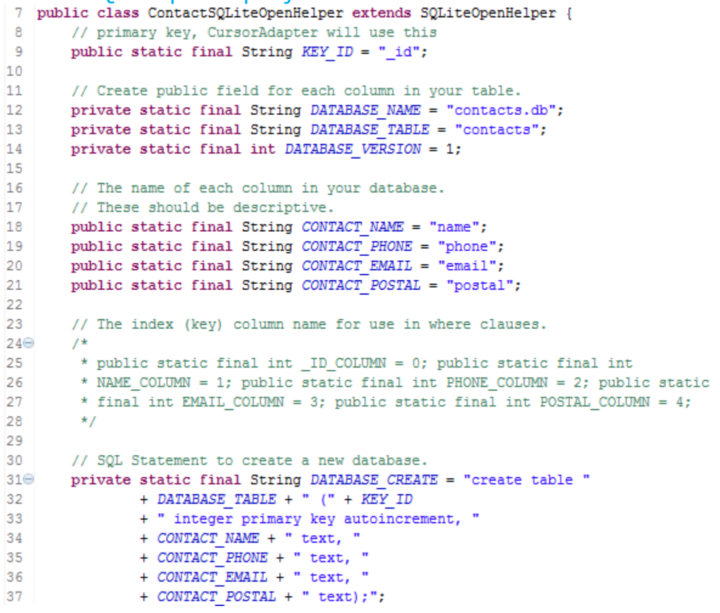
4. Create a java bean for contact object : ContactInfo.java



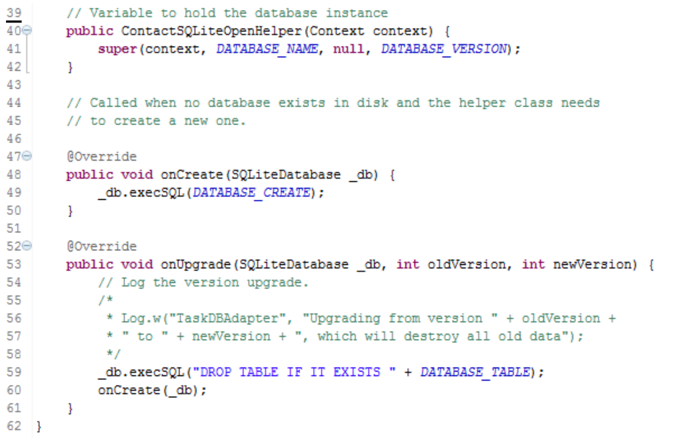
```java
public class ContactInfo {
    private int id;
    private String name;
    private String phone;
    private String email;
    private String postal;

    public ContactInfo() {
    }

    public ContactInfo(String name, String phone, String em
        super();
        this.name = name;
        this.phone = phone;
        this.email = email;
        this.postal = postal;
    }

    public ContactInfo(int id, String name, String phone, S
            String postal) {
        this.id = id;
        this.name = name;
        this.phone = phone;
        this.email = email;
        this.postal = postal;
    }

    public int getId() {
        return id;
```

5. Create an SQLite Open Helper:

ContactSQLiteOpenHelper.java



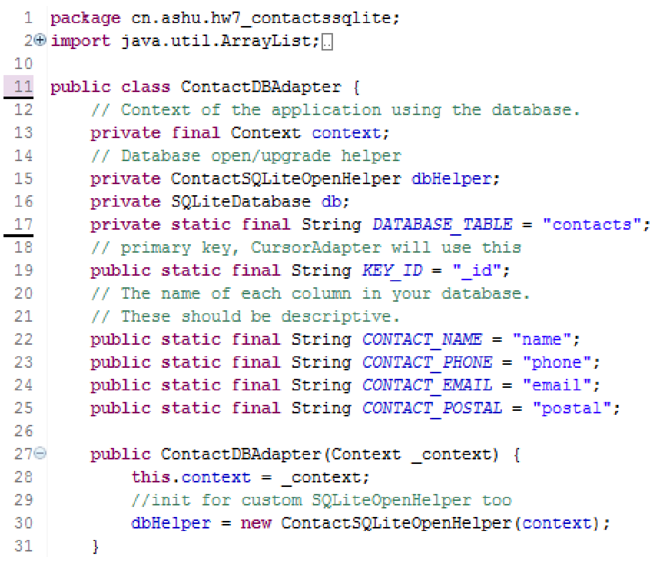
```java
public class ContactSQLiteOpenHelper extends SQLiteOpenHelper {
    // primary key, CursorAdapter will use this
    public static final String KEY_ID = "_id";

    // Create public field for each column in your table.
    private static final String DATABASE_NAME = "contacts.db";
    private static final String DATABASE_TABLE = "contacts";
    private static final int DATABASE_VERSION = 1;

    // The name of each column in your database.
    // These should be descriptive.
    public static final String CONTACT_NAME = "name";
    public static final String CONTACT_PHONE = "phone";
    public static final String CONTACT_EMAIL = "email";
    public static final String CONTACT_POSTAL = "postal";

    // The index (key) column name for use in where clauses.
    /*
     * public static final int _ID_COLUMN = 0; public static final int
     * NAME_COLUMN = 1; public static final int PHONE_COLUMN = 2; public static
     * final int EMAIL_COLUMN = 3; public static final int POSTAL_COLUMN = 4;
     */

    // SQL Statement to create a new database.
    private static final String DATABASE_CREATE = "create table "
            + DATABASE_TABLE + " (" + KEY_ID
            + " integer primary key autoincrement, "
            + CONTACT_NAME + " text, "
            + CONTACT_PHONE + " text, "
            + CONTACT_EMAIL + " text, "
            + CONTACT_POSTAL + " text);";
```



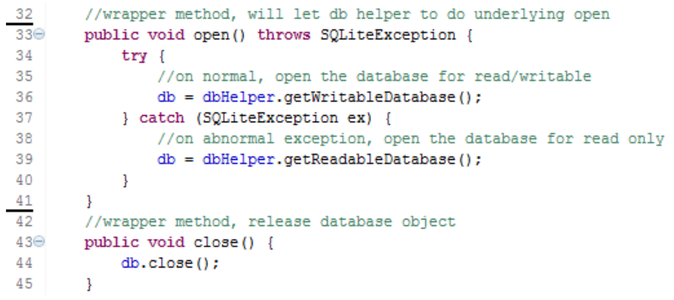
```java
    // Variable to hold the database instance
    public ContactSQLiteOpenHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    // Called when no database exists in disk and the helper class needs
    // to create a new one.

    @Override
    public void onCreate(SQLiteDatabase _db) {
        _db.execSQL(DATABASE_CREATE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase _db, int oldVersion, int newVersion) {
        // Log the version upgrade.
        /*
         * Log.w("TaskDBAdapter", "Upgrading from version " + oldVersion +
         * " to " + newVersion + ", which will destroy all old data");
         */
        _db.execSQL("DROP TABLE IF IT EXISTS " + DATABASE_TABLE);
        onCreate(_db);
    }
}
```

6. Create a database adapter class for the database:

ContactDBAdapter.java

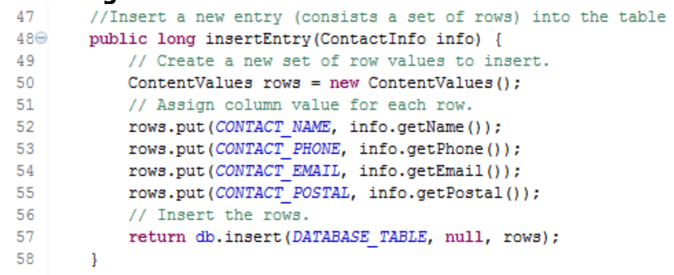


```java
package cn.ashu.hw7_contactssqlite;
import java.util.ArrayList;

public class ContactDBAdapter {
    // Context of the application using the database.
    private final Context context;
    // Database open/upgrade helper
    private ContactSQLiteOpenHelper dbHelper;
    private SQLiteDatabase db;
    private static final String DATABASE_TABLE = "contacts";
    // primary key, CursorAdapter will use this
    public static final String KEY_ID = "_id";
    // The name of each column in your database.
    // These should be descriptive.
    public static final String CONTACT_NAME = "name";
    public static final String CONTACT_PHONE = "phone";
    public static final String CONTACT_EMAIL = "email";
    public static final String CONTACT_POSTAL = "postal";

    public ContactDBAdapter(Context _context) {
        this.context = _context;
        //init for custom SQLiteOpenHelper too
        dbHelper = new ContactSQLiteOpenHelper(context);
    }
```



```java
    //wrapper method, will let db helper to do underlying open
    public void open() throws SQLiteException {
        try {
            //on normal, open the database for read/writable
            db = dbHelper.getWritableDatabase();
        } catch (SQLiteException ex) {
            //on abnormal exception, open the database for read only
            db = dbHelper.getReadableDatabase();
        }
    }
    //wrapper method, release database object
    public void close() {
        db.close();
    }
```

Inserting method:



```java
    //Insert a new entry (consists a set of rows) into the table
    public long insertEntry(ContactInfo info) {
        // Create a new set of row values to insert.
        ContentValues rows = new ContentValues();
        // Assign column value for each row.
        rows.put(CONTACT_NAME, info.getName());
        rows.put(CONTACT_PHONE, info.getPhone());
        rows.put(CONTACT_EMAIL, info.getEmail());
        rows.put(CONTACT_POSTAL, info.getPostal());
        // Insert the rows.
        return db.insert(DATABASE_TABLE, null, rows);
    }
```

Querying one row method:

```java
    // return a single DataModel object based on what name to search
    public ContactInfo getEntry(String searchname) throws SQLException {
        Cursor cursor = db.query(DATABASE_TABLE, new String[] { KEY_ID,
                CONTACT_NAME, CONTACT_PHONE, CONTACT_EMAIL, CONTACT_POSTAL },
                CONTACT_NAME + "=" + "'" + searchname.trim() + "'", null, null,
                null, null, null);
        if ((cursor.getCount() == 0) || !cursor.moveToFirst()) {
            throw new SQLException("No item found for name: " + searchname);
        }
        int theId = cursor.getInt(cursor.getColumnIndex(KEY_ID));
        String name = cursor.getString(cursor.getColumnIndex(CONTACT_NAME));
        String phone = cursor.getString(cursor.getColumnIndex(CONTACT_PHONE));
        String email = cursor.getString(cursor.getColumnIndex(CONTACT_EMAIL));
        String postaladdr = cursor.getString(cursor
                .getColumnIndex(CONTACT_POSTAL));
        ContactInfo result = new ContactInfo(theId, name, phone, email,
                postaladdr);
        return result;
    }
```

Querying all rows method:

```java
    public List<ContactInfo> getAllEntries() {
        List<ContactInfo> list = new ArrayList<ContactInfo>();
        Cursor cursor = db.query(DATABASE_TABLE, new String[] { KEY_ID,
                CONTACT_NAME, CONTACT_PHONE, CONTACT_EMAIL, CONTACT_POSTAL },
                null, null, null, null, null);
        while (cursor.moveToNext()) {
            int theId = cursor.getInt(cursor.getColumnIndex(KEY_ID));
            String name = cursor.getString(cursor.getColumnIndex(CONTACT_NAME));
            String phone = cursor.getString(cursor
                    .getColumnIndex(CONTACT_PHONE));
            String email = cursor.getString(cursor
                    .getColumnIndex(CONTACT_EMAIL));
            String postal = cursor.getString(cursor
                    .getColumnIndex(CONTACT_POSTAL));
            ContactInfo info = new ContactInfo(theId, name, phone, email,
                    postal);
            list.add(info);
        }
        return list;
    }
```

Deleting all rows and deleting one row methods:

```java
    // Remove all entries in the table
    public int deleteAllEntries() {
        int num = db.delete(DATABASE_TABLE, null, null);
        // int num = db.delete(DATABASE_TABLE, null, null);
        return num;
    }

    // Remove an entry based on its row index
    public int removeEntry(long _rowIndex) {
        int num = db.delete(DATABASE_TABLE, KEY_ID + "=?",
                new String[] { _rowIndex + "" });
        return num;
    }

}
```

7. Create contact form Activity: ContactFormActivty.java

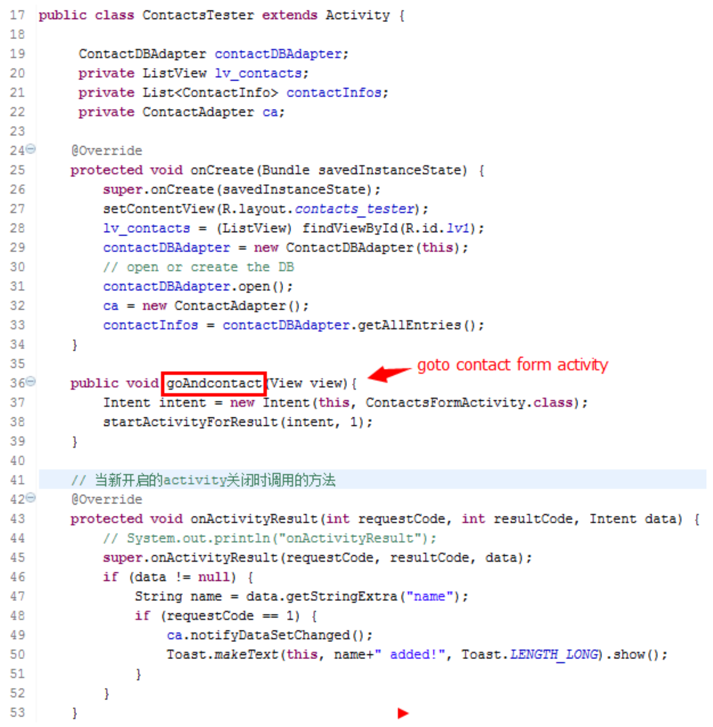
```java
public class ContactsFormActivity extends Activity {
    EditText et_name;
    EditText et_phone;
    EditText et_email;
    EditText et_postal;
    ContactDBAdapter contactDBAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        contactDBAdapter = new ContactDBAdapter(this);
        // open or create the DB
        contactDBAdapter.open();
        super.onCreate(savedInstanceState);
        setContentView(R.layout.contacts);

        et_name = (EditText) findViewById(R.id.et_name);
        et_phone = (EditText) findViewById(R.id.et_phone);
        et_email = (EditText) findViewById(R.id.et_email);
        et_postal = (EditText) findViewById(R.id.et_postal);
    }

    public void clickSave(View view) {
        String name = et_name.getText().toString();
        String phone = et_phone.getText().toString();
        String email = et_email.getText().toString();
        String postal = et_postal.getText().toString();

        // System.out.println("save"+ name+phone+email+postal);
        if (TextUtils.isEmpty(name)) {
            Toast.makeText(this, "The name is empty!!", Toast.LENGTH_SHORT)
                    .show();
        } else {
            ContactInfo info = new ContactInfo(name, phone, email, postal);
            if (contactDBAdapter.insertEntry(info) != 0) {
                Intent data = new Intent();
                data.putExtra("name", name);
                setResult(0, data);
            }
            finish();
        }
    }

    public void onDestroy() {
        // close the DB
        super.onDestroy();
        contactDBAdapter.close();
    }
}
```

for save button

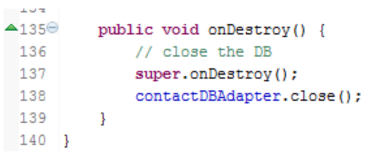
8. Registing form activity to manifest:

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cn.ashu.hw7_contactssqlite"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="18" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="cn.ashu.hw7_contactssqlite.ContactsTester"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="cn.ashu.hw7_contactssqlite.ContactsFormActivity"
            android:label="@string/and_contact" >
            </activity>
    </application>

</manifest>
```

9. Create main Activity Class: Contactstester.java

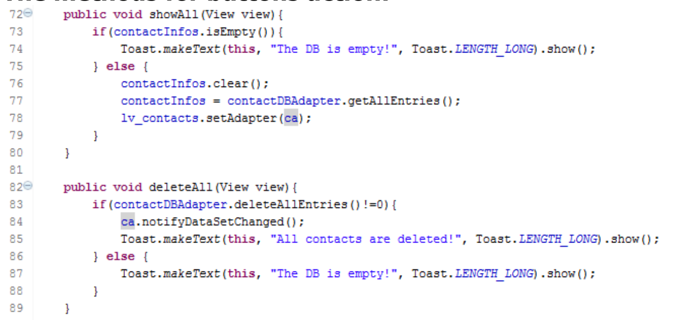


```java
public class ContactsTester extends Activity {

     ContactDBAdapter contactDBAdapter;
     private ListView lv_contacts;
     private List<ContactInfo> contactInfos;
     private ContactAdapter ca;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.contacts_tester);
        lv_contacts = (ListView) findViewById(R.id.lv1);
        contactDBAdapter = new ContactDBAdapter(this);
        // open or create the DB
        contactDBAdapter.open();
        ca = new ContactAdapter();
        contactInfos = contactDBAdapter.getAllEntries();
    }

    public void goAndcontact(View view){
        Intent intent = new Intent(this, ContactsFormActivity.class);
        startActivityForResult(intent, 1);
    }

    // 当新开启的activity关闭时调用的方法
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // System.out.println("onActivityResult");
        super.onActivityResult(requestCode, resultCode, data);
        if (data != null) {
            String name = data.getStringExtra("name");
            if (requestCode == 1) {
                ca.notifyDataSetChanged();
                Toast.makeText(this, name+" added!", Toast.LENGTH_LONG).show();
            }
        }
    }
```



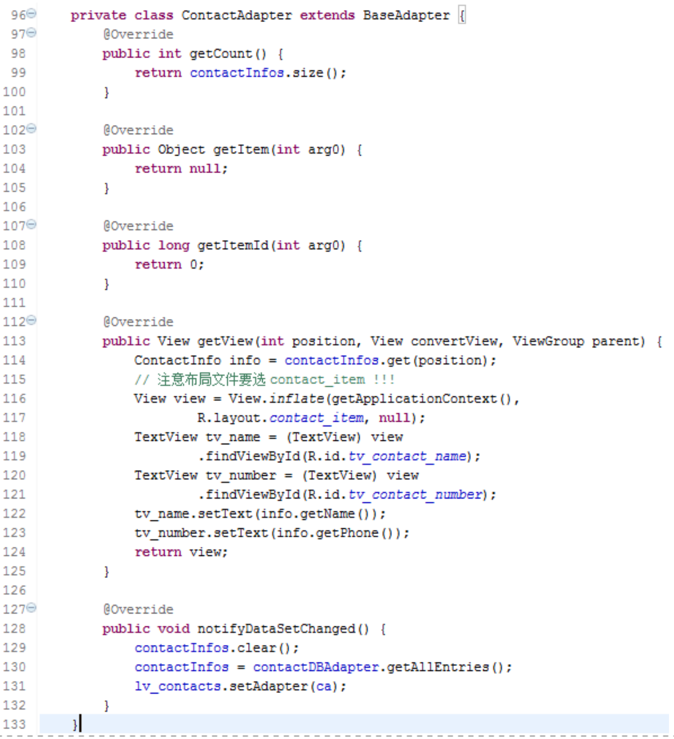
```java
    public void onDestroy() {
        // close the DB
        super.onDestroy();
        contactDBAdapter.close();
    }
}
```

The methods for buttons action:



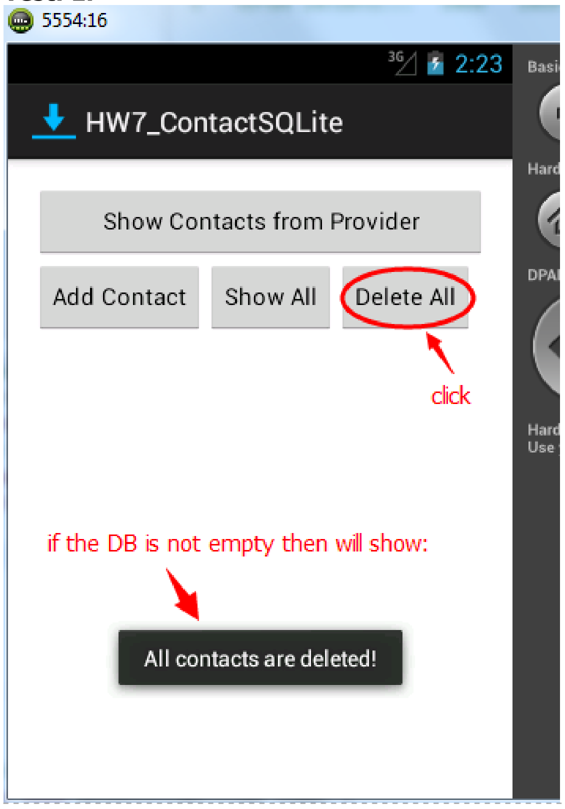
```java
    public void showAll(View view){
        if(contactInfos.isEmpty()){
            Toast.makeText(this, "The DB is empty!", Toast.LENGTH_LONG).show();
        } else {
            contactInfos.clear();
            contactInfos = contactDBAdapter.getAllEntries();
            lv_contacts.setAdapter(ca);
        }
    }

    public void deleteAll(View view){
        if(contactDBAdapter.deleteAllEntries()!=0){
            ca.notifyDataSetChanged();
            Toast.makeText(this, "All contacts are deleted!", Toast.LENGTH_LONG).show();
        } else {
            Toast.makeText(this, "The DB is empty!", Toast.LENGTH_LONG).show();
        }
    }
```

Create an inner class extents BaseAdapter to bind data to the ListView of contacts:

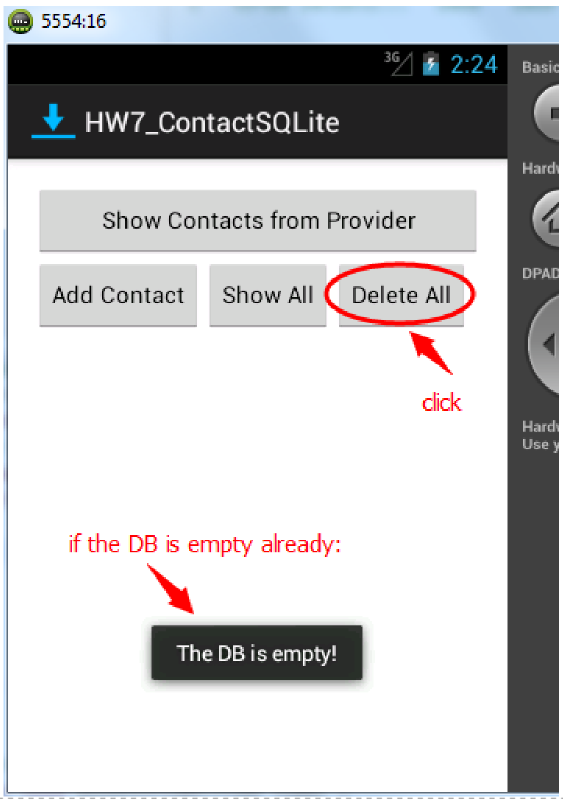


```java
    private class ContactAdapter extends BaseAdapter {
        @Override
        public int getCount() {
            return contactInfos.size();
        }

        @Override
        public Object getItem(int arg0) {
            return null;
        }

        @Override
        public long getItemId(int arg0) {
            return 0;
        }

        @Override
        public View getView(int position, View convertView, ViewGroup parent) {
            ContactInfo info = contactInfos.get(position);
            // 注意布局文件要选 contact_item !!!
            View view = View.inflate(getApplicationContext(),
                    R.layout.contact_item, null);
            TextView tv_name = (TextView) view
                    .findViewById(R.id.tv_contact_name);
            TextView tv_number = (TextView) view
                    .findViewById(R.id.tv_contact_number);
            tv_name.setText(info.getName());
            tv_number.setText(info.getPhone());
            return view;
        }

        @Override
        public void notifyDataSetChanged() {
            contactInfos.clear();
            contactInfos = contactDBAdapter.getAllEntries();
            lv_contacts.setAdapter(ca);
        }
    }
```

Test: 1.

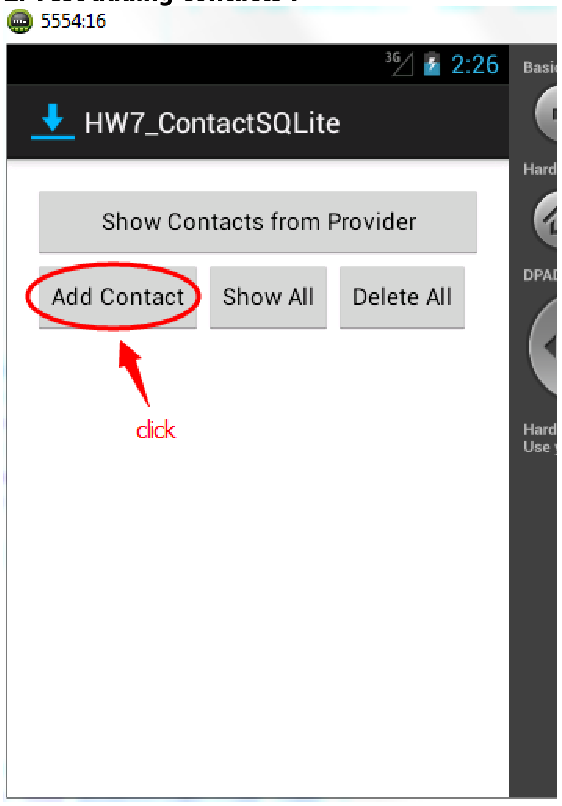

5554:16
2:23
HW7_ContactSQLite
Show Contacts from Provider
Add Contact
Show All
Delete All
click
if the DB is not empty then will show:

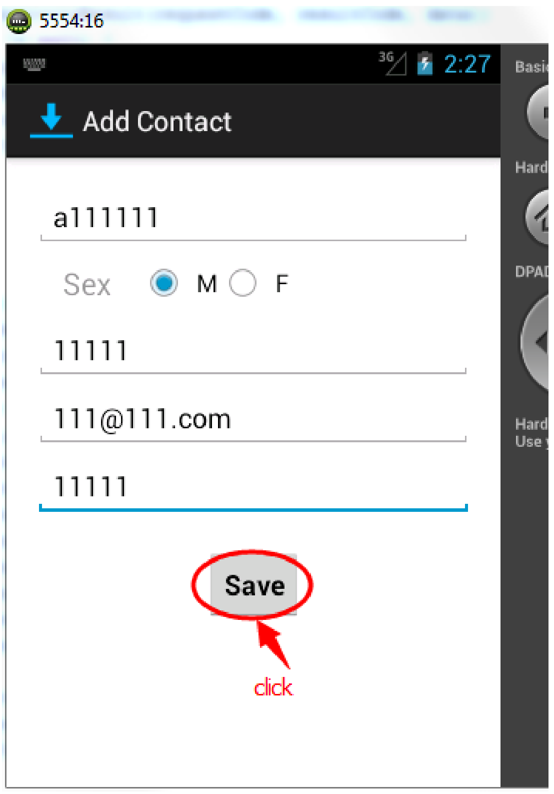
All contacts are deleted!

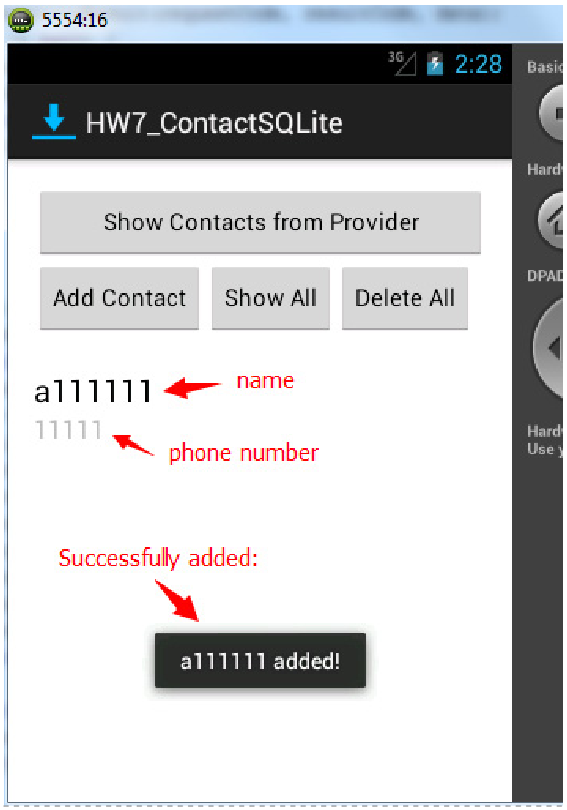


The DB is empty!

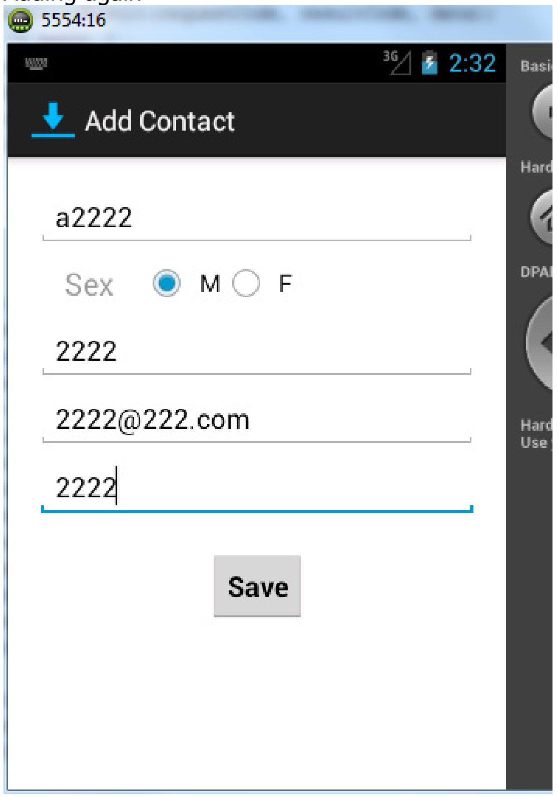
2. Test adding contacts :

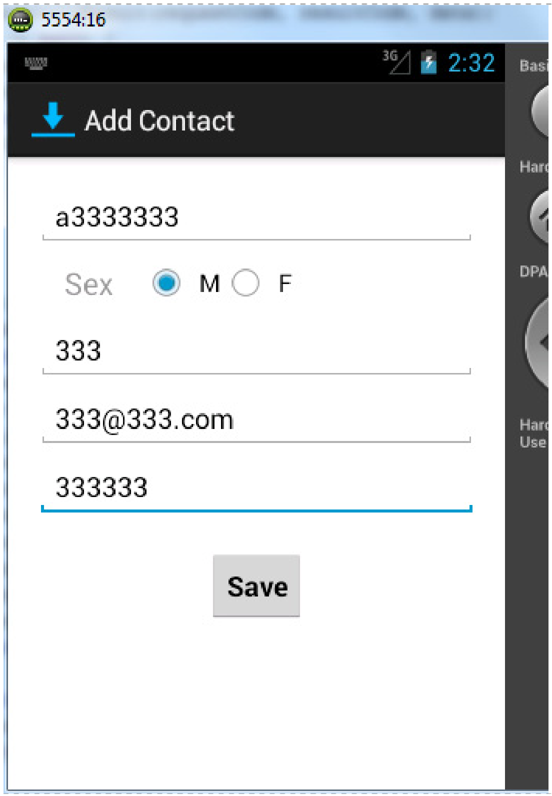






Adding again



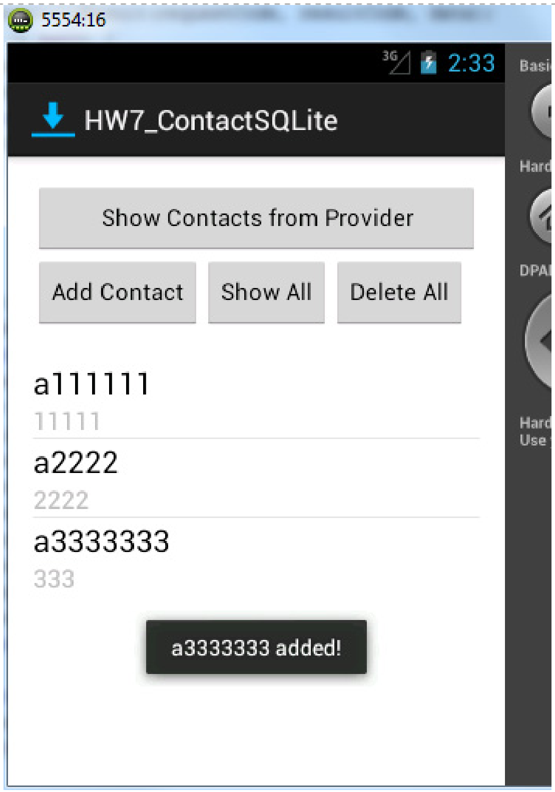


a3333333

333

333@333.com

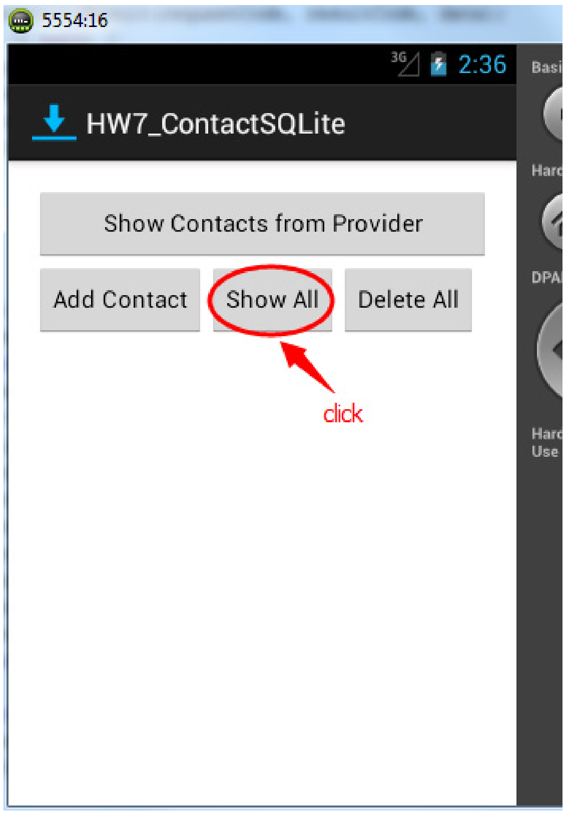
333333

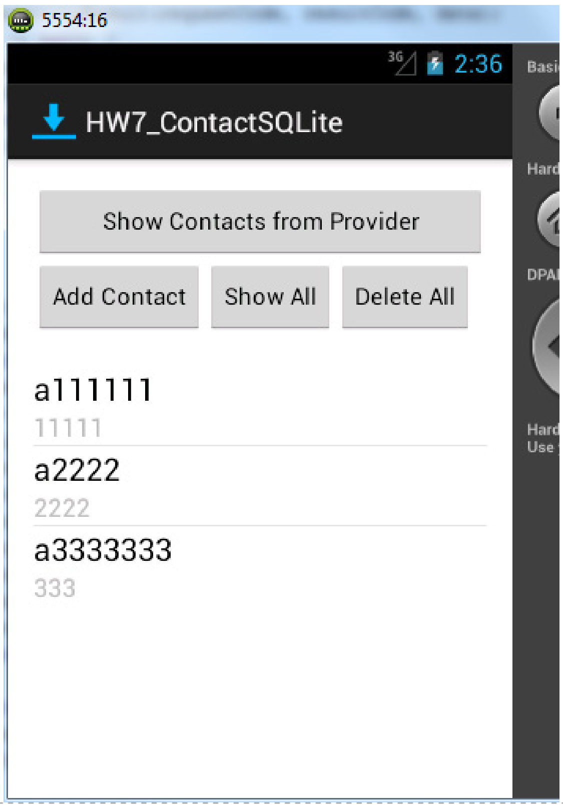


a111111
11111
a2222
2222
a3333333
333

a3333333 added!

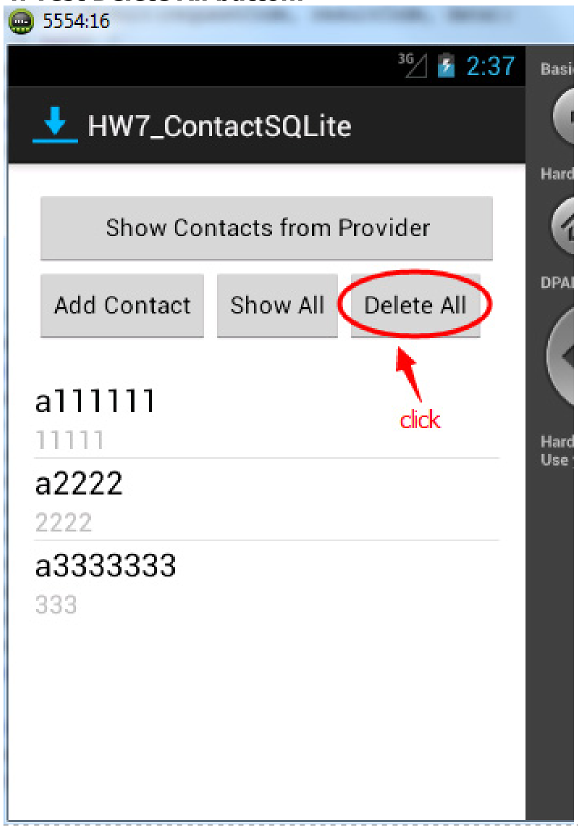
3. Test show all button :
Quite the app and run app again it will showes:

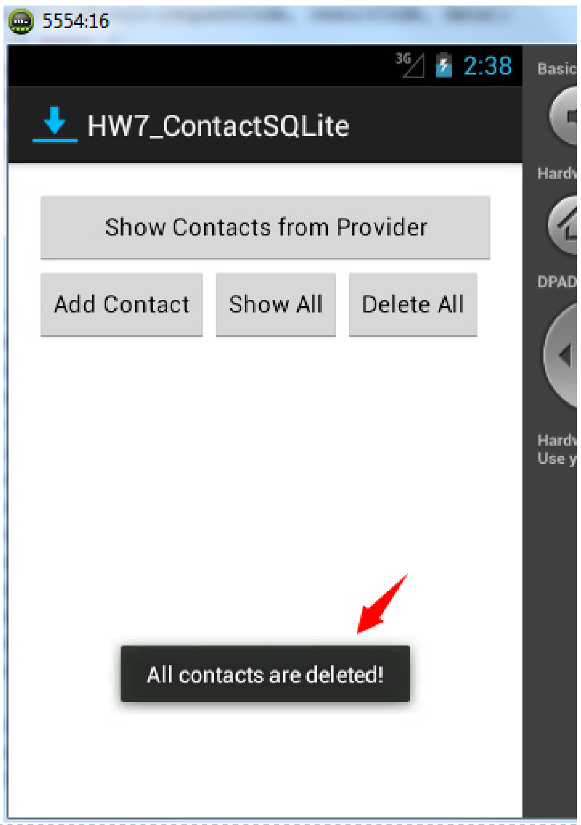




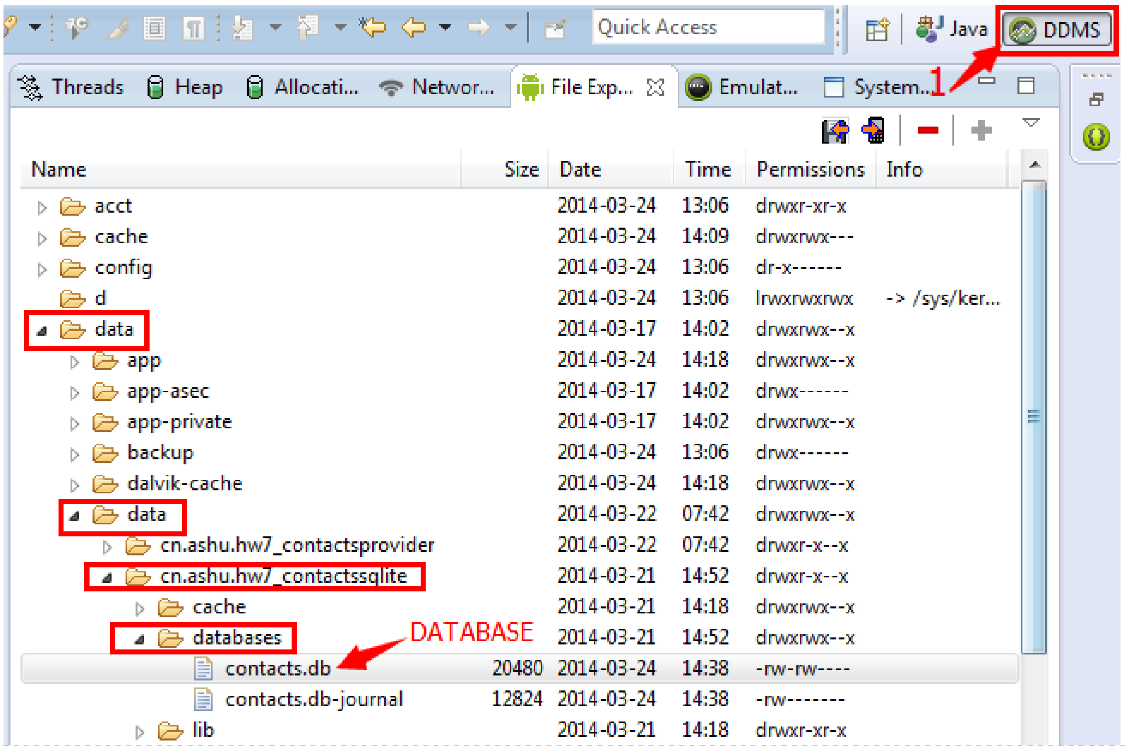
a111111
11111
a2222
2222
a3333333
333

4. Test Delete All button:





5. Showing the location of database in emulator:

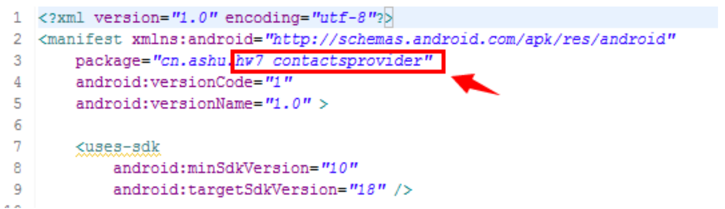


Name	Size	Date	Time	Permissions	Info
acct		2014-03-24	13:06	drwxr-xr-x	
cache		2014-03-24	14:09	drwxrwx---	
config		2014-03-24	13:06	dr-x------	
d		2014-03-24	13:06	lrwxrwxrwx	-> /sys/ker...
data		2014-03-17	14:02	drwxrwx--x	
app		2014-03-24	14:18	drwxrwx--x	
app-asec		2014-03-17	14:02	drwx------	
app-private		2014-03-17	14:02	drwxrwx--x	
backup		2014-03-24	13:06	drwx------	
dalvik-cache		2014-03-24	14:18	drwxrwx--x	
data		2014-03-22	07:42	drwxrwx--x	
cn.ashu.hw7_contactsprovider		2014-03-22	07:42	drwxr-x--x	
cn.ashu.hw7_contactssqlite		2014-03-21	14:52	drwxr-x--x	
cache		2014-03-21	14:18	drwxrwx--x	
databases		2014-03-21	14:52	drwxrwx--x	
contacts.db	20480	2014-03-24	14:38	-rw-rw----	
contacts.db-journal	12824	2014-03-24	14:38	-rw-------	
lib		2014-03-21	14:18	drwxr-xr-x	

Q6 Solution: (base on Q5)

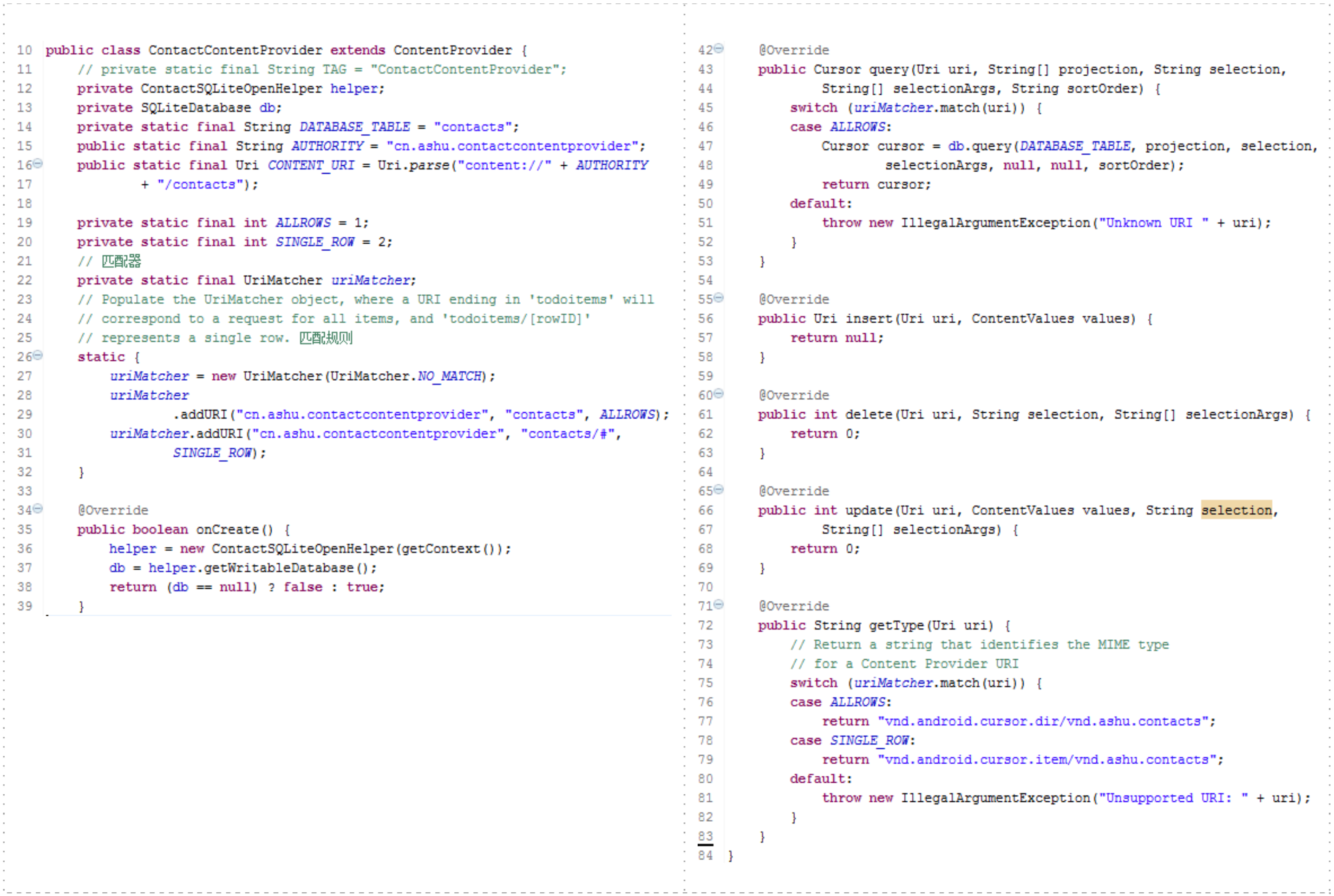
1. Copy Q5 project as a new project, rename new project to HW7_ContactsProvider

2. Edit AndroidManifest.xml**, edit package value** (for the path of R.java) :



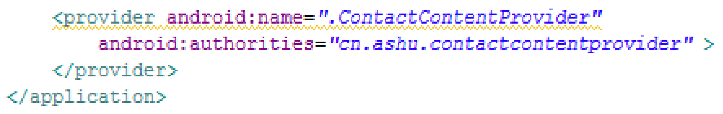
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cn.ashu.hw7_contactsprovider"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="18" />
```

3.Creating a new ***ContactContentProvider*** **class. It will be used to host the data-base using the** ***ContactSQLiteOpenHelper*** **and manage the database interactions by extending the ContentProvider class**



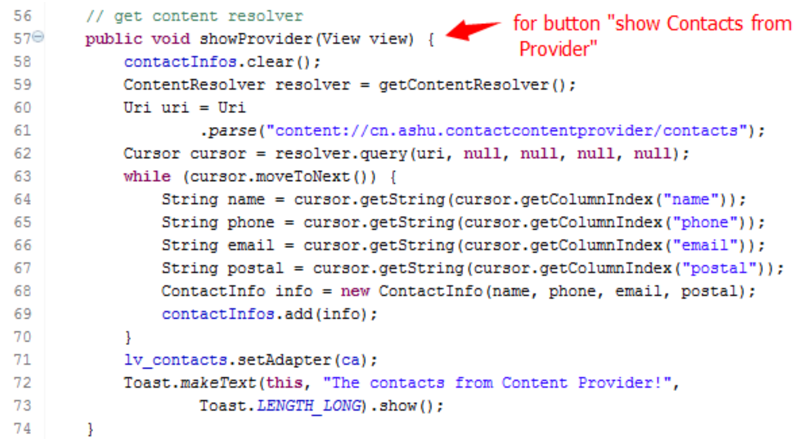
```java
public class ContactContentProvider extends ContentProvider {
    // private static final String TAG = "ContactContentProvider";
    private ContactSQLiteOpenHelper helper;
    private SQLiteDatabase db;
    private static final String DATABASE_TABLE = "contacts";
    public static final String AUTHORITY = "cn.ashu.contactcontentprovider";
    public static final Uri CONTENT_URI = Uri.parse("content://" + AUTHORITY
            + "/contacts");

    private static final int ALLROWS = 1;
    private static final int SINGLE_ROW = 2;
    // 匹配器
    private static final UriMatcher uriMatcher;
    // Populate the UriMatcher object, where a URI ending in 'todoitems' will
    // correspond to a request for all items, and 'todoitems/[rowID]'
    // represents a single row. 匹配规则
    static {
        uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
        uriMatcher
                .addURI("cn.ashu.contactcontentprovider", "contacts", ALLROWS);
        uriMatcher.addURI("cn.ashu.contactcontentprovider", "contacts/#",
                SINGLE_ROW);
    }

    @Override
    public boolean onCreate() {
        helper = new ContactSQLiteOpenHelper(getContext());
        db = helper.getWritableDatabase();
        return (db == null) ? false : true;
    }

    @Override
    public Cursor query(Uri uri, String[] projection, String selection,
            String[] selectionArgs, String sortOrder) {
        switch (uriMatcher.match(uri)) {
        case ALLROWS:
            Cursor cursor = db.query(DATABASE_TABLE, projection, selection,
                    selectionArgs, null, null, sortOrder);
            return cursor;
        default:
            throw new IllegalArgumentException("Unknown URI " + uri);
        }
    }

    @Override
    public Uri insert(Uri uri, ContentValues values) {
        return null;
    }

    @Override
    public int delete(Uri uri, String selection, String[] selectionArgs) {
        return 0;
    }

    @Override
    public int update(Uri uri, ContentValues values, String selection,
            String[] selectionArgs) {
        return 0;
    }

    @Override
    public String getType(Uri uri) {
        // Return a string that identifies the MIME type
        // for a Content Provider URI
        switch (uriMatcher.match(uri)) {
        case ALLROWS:
            return "vnd.android.cursor.dir/vnd.ashu.contacts";
        case SINGLE_ROW:
            return "vnd.android.cursor.item/vnd.ashu.contacts";
        default:
            throw new IllegalArgumentException("Unsupported URI: " + uri);
        }
    }
}
```

4. Adding the Content Provider to the application Manifest, specifying the base URI to use as its authority.



```xml
    <provider android:name=".ContactContentProvider"
        android:authorities="cn.ashu.contactcontentprovider" >
    </provider>
</application>
```

5. Open the project ***HW7_ContactsSQLite*** **(Q5) and edit the main activity** *ContactsTester.java* **and adding a method:**

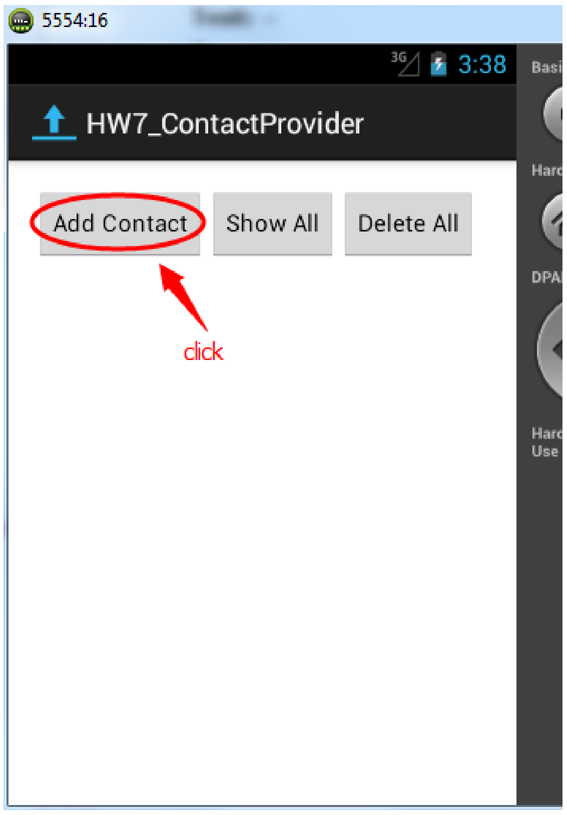


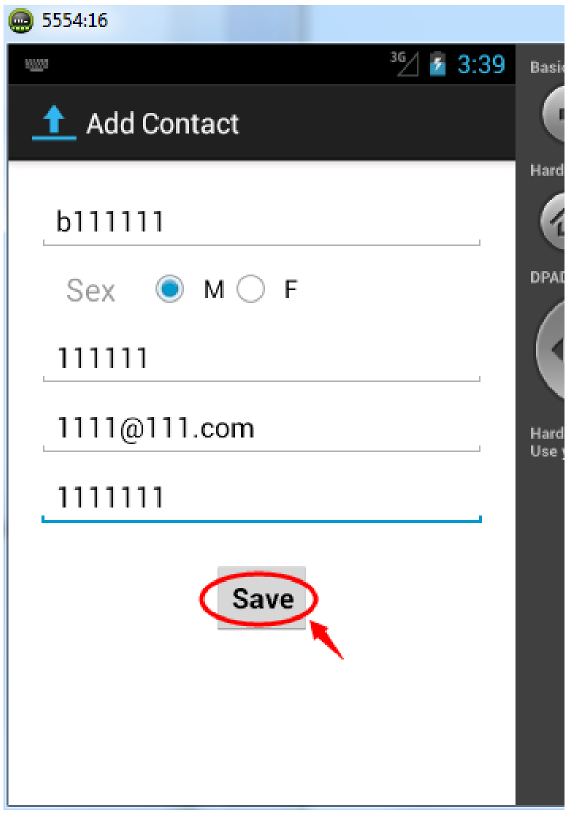
for button "show Contacts from Provider"

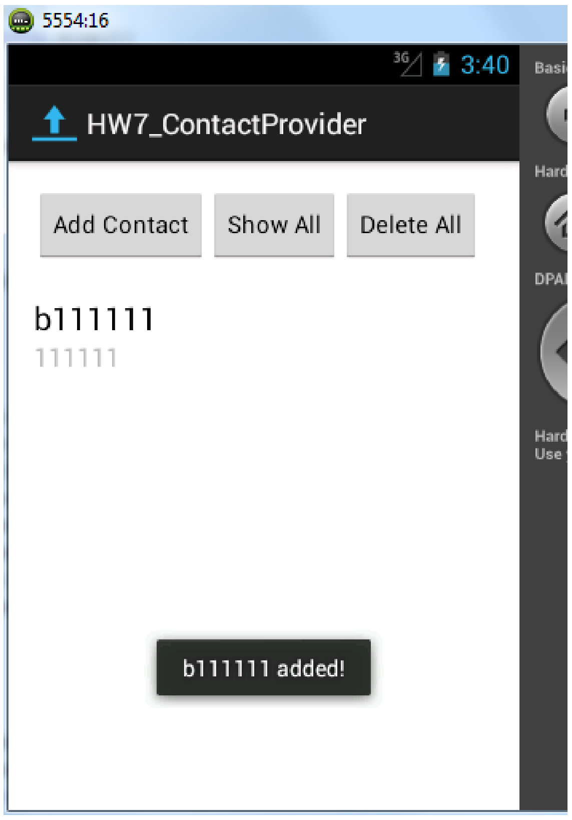
```java
    // get content resolver
    public void showProvider(View view) {
        contactInfos.clear();
        ContentResolver resolver = getContentResolver();
        Uri uri = Uri
                .parse("content://cn.ashu.contactcontentprovider/contacts");
        Cursor cursor = resolver.query(uri, null, null, null, null);
        while (cursor.moveToNext()) {
            String name = cursor.getString(cursor.getColumnIndex("name"));
            String phone = cursor.getString(cursor.getColumnIndex("phone"));
            String email = cursor.getString(cursor.getColumnIndex("email"));
            String postal = cursor.getString(cursor.getColumnIndex("postal"));
            ContactInfo info = new ContactInfo(name, phone, email, postal);
            contactInfos.add(info);
        }
        lv_contacts.setAdapter(ca);
        Toast.makeText(this, "The contacts from Content Provider!",
                Toast.LENGTH_LONG).show();
    }
```

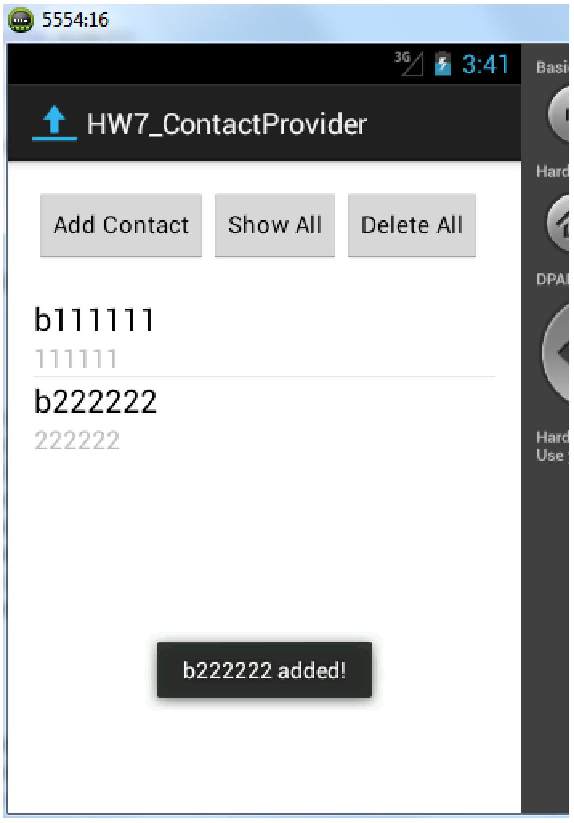
Test:

1. Adding contacts to database

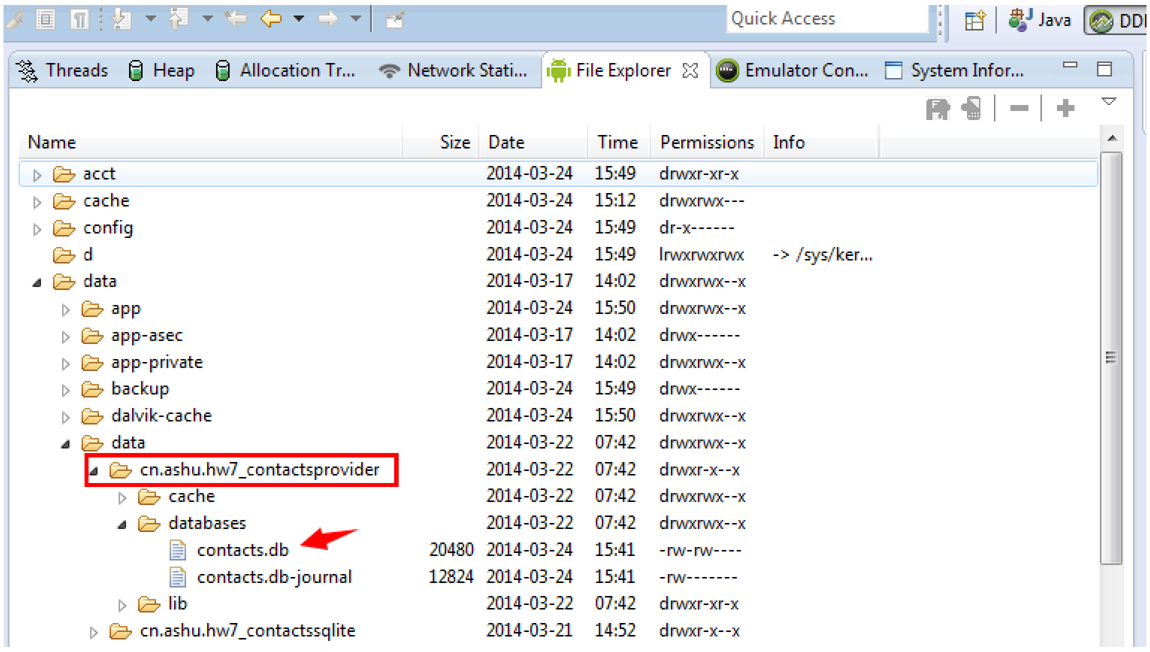






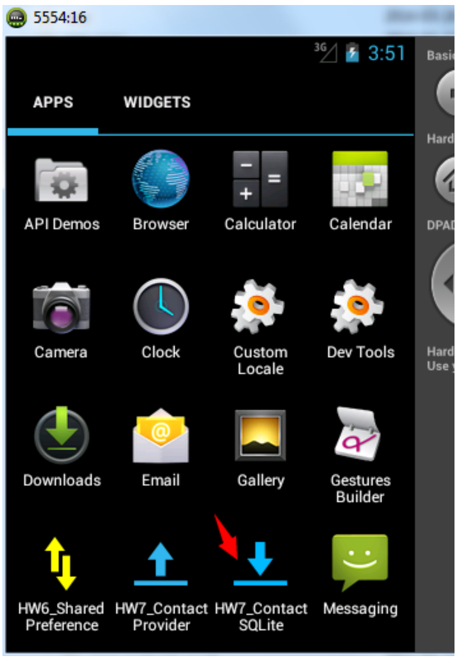


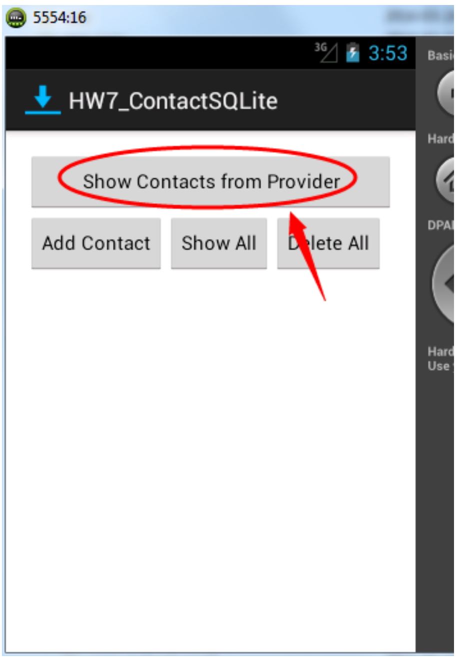
2. Showing the location of database in emulator:

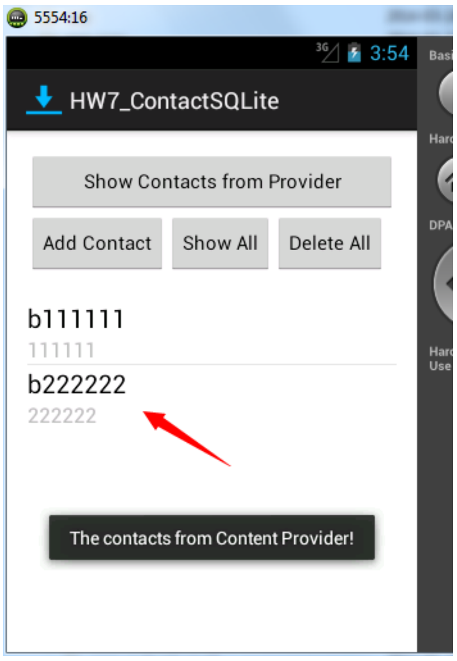


Name	Size	Date	Time	Permissions	Info
acct		2014-03-24	15:49	drwxr-xr-x	
cache		2014-03-24	15:12	drwxrwx---	
config		2014-03-24	15:49	dr-x------	
d		2014-03-24	15:49	lrwxrwxrwx	-> /sys/ker...
data		2014-03-17	14:02	drwxrwx--x	
app		2014-03-24	15:50	drwxrwx--x	
app-asec		2014-03-17	14:02	drwx------	
app-private		2014-03-17	14:02	drwxrwx--x	
backup		2014-03-24	15:49	drwx------	
dalvik-cache		2014-03-24	15:50	drwxrwx--x	
data		2014-03-22	07:42	drwxrwx--x	
cn.ashu.hw7_contactsprovider		2014-03-22	07:42	drwxr-x--x	
cache		2014-03-22	07:42	drwxrwx--x	
databases		2014-03-22	07:42	drwxrwx--x	
contacts.db	20480	2014-03-24	15:41	-rw-rw----	
contacts.db-journal	12824	2014-03-24	15:41	-rw-------	
lib		2014-03-22	07:42	drwxr-xr-x	
cn.ashu.hw7_contactssqlite		2014-03-21	14:52	drwxr-x--x	

3. Opening the Q5 APP: HW7_ContactSQLite , and click "Show Contacts from Provider"







b111111
111111
b222222
222222

The contacts from Content Provider!