Android 平台的数据存储技术①

彭 艳，杨 欧

(深圳职业技术学院，深圳 518055)

摘 要：针对应用程序开发过程中不可避免的数据存储问题，从分析 Android 平台及其数据存储途径入手，详细阐述了 SQLite 数据库在该平台上的应用，并以电话簿为例，给出了开发的具体过程以及仿真结果。最后，提出了在应用程序间进一步共享数据库的解决方案。

关键词：Android；数据存储；SQLite

Analysis on the Data Storage of Android Application

PENG Yan, YANG Ou

(Shenzhen Polytechnic, Shenzhen 518055, China)

Abstract: Aiming to the inevitable problems of data storage in application development process, the paper analysed Android and its data storage methods, and elaborated SQLite database on Android. Taking the phonebook as an example, the paper gave out the specific development process as well as the simulation results. Finally, this paper presented the further solutions to share SQLite database.

Key words: Android; data storage; SQLite

1 引言

Android 是 Google 公司于 2007 年宣布的基于 Linux 平台的开源手机操作系统的名称，该平台由操作系统、中间件、用户界面和应用软件组成，号称是首个为移动终端打造的真正开放和完整的移动软件，具有显著的开放性、丰富的硬件平台支持、自由的第三方软件市场以及无缝结合 Google 服务等明显的优势。

随着 Android 在智能手机市场上的迅速崛起，2011 年 7 月推出的 Android 3.2 已经开始支持 7 英寸设备，并引入了应用显示缩放功能，可以让那些针对手机开发的应用，更平滑的显示在平板电脑上。国内外也已经有越来越多的专家正在将开源的 Android 向各种嵌入式硬件平台移植，这些都意味着，Android 正在从一个单纯的智能手机操作系统向多元化的嵌入式操作系统转变，未来会涌现出更多基于 Android 的嵌入式应用开发。

2 Android的数据存储

作为应用开发中非常重要的一环，数据存储从来都是开发过程中不可回避的关键技术。在 Android 中，可供选择的存储方式有五种，分别是：系统配置，文件存储，SQLite 数据库方式，内容提供器（Content provider）和网络。其中，前三种方式主要用于应用程序的内部存储，后两种主要用于外部存储。实际应用开发中，主要使用前三种数据存储的方式。

(1) 系统配置(Shared Preferences)

为了保存应用程序的系统配置信息，Android 平台提供了 SharedPreferences 类，它是一个轻量级的存储类。通过 SharedPreferences 可以将 NVP（Name/Value Pair,名称/值对）保存在 Android 的文件系统中。

(2) 文件（Files）

Android 使用的是基于 Linux 的文件系统，允许应用程序创建仅能够自身访问的私有文件，用于应用数据的存取。文件可以保存在设备的内部存储器上，还可以保存在 SD 卡等外部存储设备中。在文件的访问方式上，Android 系统不仅支持标准 Java 的 IO 类和方法，还提供了能够简化读写流式文件过程的函数，如

① 收稿时间:2011-08-23;收到修改稿时间:2011-09-21

openFileOutput()、openFileInput()等。

值得注意的是，Android 中每一个应用程序都使用不同的 User ID，因此，创建的文件仅应用程序自身可见。若两个程序之间需要进行数据交换，就必须通过前面提到的外部数据存储方式 ContentProvider。

(3) 数据库(SQLite Database)

考虑到节省系统资源，Andriod 选择并封装了了开源的超轻量级嵌入式数据库 SQLite。SQLite 由 D. Richard Hipp 开发，SQLite3.0 全部源代码不足 3 万行，编译后的动态链接库大小为 300kb 左右，管理的数据量达到 2TB，提供 B-Tree 存储数据的模式，数据以 ASCII 码形式存储，支持 SQL 快速查询，具有小、快、简单、可靠、安全、稳定、完全免费等特点。

Android 提供了两个类，用于简化 SQLite 数据库的操作。第一个是 SQLiteOpenHelper，该类主要用于自动完成打开、关闭数据库，当数据库不存在的情况下还可自动创建数据库，并根据需要更新数据库。第二个是 SQLiteDatabase 类，该类封装了数据库操作的 API，主要用于查询数据库、向数据库中新建数据项、删除数据项等。

创建一个数据库的一般流程如下：

① 根据应用设计出所需要的数据表格。

② 在应用程序中，通过继承和改写 SQLite OpenHelper，创建、打开数据库。

③ 创建 SQLiteDatabase 对象，建立数据库接口。

④ 利用接口，对数据库进行读写操作。

⑤ 创建数据库的查询接口，实现查询操作。

⑥ 关闭数据库。

应用程序访问 SQLite 数据库的模型如图 1 所示。

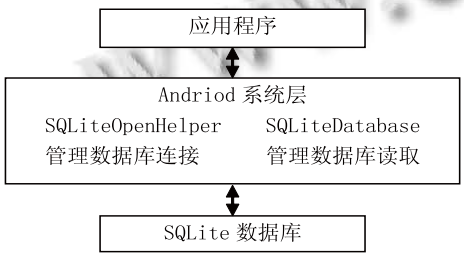

图 1 应用程序访问 SQLite 数据库模型

3 基于SQLite数据库的电话簿

下面，以电话簿为例，给出在 Android 平台上进行 SQLite 数据库应用开发的具体过程与方法。电话簿表格中设置的字段有：姓名，电话，邮箱等。

1) 新建项目 MySQLite，在 src 目录下新建一个 PhoneBook.java 类，并在该类中声明数据库的基本信息，包括数据库文件的名称、数据库表格名称和数据库版本，以及数据库表中的属性名称等。

接下来，在该类中添加一个 SQLiteOpenHelper 的子类 DBOpenHelper，实现数据库的创建和打开。

在 DBOpenHelper 类的构造函数中，通过调用 SQLiteDatabase 对象的 execSQL()方法，执行创建表的 SQL 命令。值得注意的是，继承 SQLiteOpenHelper 类必须重载两个函数：onCreate()函数和 onUpgrade()函数。onCreate()函数在数据库第一次建立时被调用，一般用来创建数据库中的表，并做适当的初始化工作。onUpgrade()函数在数据库需要升级时被调用，一般用来删除旧的数据库表，并将数据转移到新版本的数据库表中。具体代码如下：

```
private static class DBOpenHelper extends SQLite OpenHelper {
    public DBOpenHelper(Context context, String name, CursorFactory factory, int version){
        super(context, name, factory, version);
    }
    private static final String DB_CREATE = "create table phonebook (key_id integer primary key autoincrement, name text not null, telephone text,email text);";
    @Override
    public void onCreate(SQLiteDatabase _db) {
        _db.execSQL(DB_CREATE);
    }
    @Override
    public void onUpgrade(SQLiteDatabase _db, int _oldVersion, int _newVersion) {
        _db.execSQL("DROP TABLE IF EXISTS phonebook);
        onCreate(_db);
    }
```

接下来，在 DBOpenHelper 类中定义一个 open()函数，用来初始化 SQLiteDatabase 类的对象 db 并打开数据库，该函数调用了 SQLiteOpenHelper 类的 getWritableDatabase()函数和 getReadableDatabase()函

数。这两个函数会根据数据库是否存在、版本号和是否可写等情况，决定在返回数据库对象前，是否需要建立数据库。

```
public void open() throws SQLiteException {
    dbOpenHelper = new DBOpenHelper(context, DB_NAME, null, DB_VERSION);
        try {
            db = dbOpenHelper. getWritableDatabase();
            }catch (SQLiteException ex) {
            db = dbOpenHelper.getReadableDatabase();
            }
}
```

最后还要在 DBOpenHelper 类中定义 close()函数，通过调用 SQLiteDatabase 对象的 close()方法关闭数据库。当数据库使用完毕时，必须及时调用此函数。

(2) 数据库创建成功后，只要利用 SQLiteDatabase 类的对象 db 建立好连接，即可通过下面两种途径，完成数据的添加、删除、更新等操作。一种是调用 SQLiteDatabase 类封装的 insert()、delete()、update()等方法，另一种是调用 SQLiteDatabase 类封装的 execSQL()方法执行 SQL 语句来完成相应操作。下面是执行添加和删除的具体代码。

```
SQLiteDatabase db= dbOpenHelper. getWritableDatabase();
String sql1=
    ”insert into phonebook(name,telephone,email) values(‘aaa’,’10086’,’10086@163.com’);”;
 String sql2=
    ”insert into phonebook(name,telephone,email) values(‘bbb’,’10001’,’10001@163.com’);”;
try{
db.execSQL(sql1);
db.execSQL(sql2);
db.delete(“phonebook”,”name=’bbb’”,null);
}catch(SQLException e){ }
```

(3) 数据库的查询。SQLiteDatabase 类的方法 rawQuery() 用于执行 select 语句，查询数据库获得目标记录集。

```
SQLiteDatabase db = dbOpenHelper. getWritableDatabase();
Cursor cursor = db.rawQuery(“select * from phonebook”, null);
while (cursor.moveToNext()) {
String name = cursor.getInt(0); //获取第一列的值
String telephone = cursor.getString(1);
String email = cursor.getInt(2);
}
cursor.close();
```

必须说明，Android 中查询返回的是 Cursor 对象，即结果集游标。Cursor 类提供了各种导航方法，用于对结果集进行随机访问。

在仿真器上运行电话簿，可以看到数据库中的测试信息，如图 2 所示。

图 2　读取电话簿内容

4　结语

Android 为应用程序提供了由内到外的丰富的数据存储方式。本文以电话簿为例，围绕 Android 平台上的 SQLite 应用进行了较为深入的研究。应用程序创建的数据库，属于私有文件。若其他应用程序需要访问这个资源，还必须将该数据源发布为内容提供器，使数据可以共享。

参考文献

1 李宗恒,李俭伟.主要智能手机操作系统发展现状及前景展望.移动通信,2010,2(3-4):115−118.
2 陈昱,江兰帆.基于 GoogleAndroid 平台的移动开发研究.福建电脑,2008,(11):156−157.
3 姚昱旻,刘卫国.Android 的架构与应用开发研究.计算机系统应用,2008,17(11):63−66.
4 彭艳.基于嵌入式数据库 SQLite 的智能导游系统.计算机系统应用,2011,20(4):254−256.
5 柯元旦,宋锐.Android 程序设计.北京:北京航空航天大学出版社,2010.
6 Meier R.王超译.Android 2 高级编程.第 2 版.北京:清华大学出版社,2010.