

# 第8章 数据存储

**2024年5月**

## 8.1 SQLite数据库

## 8.1.1 SQLite数据库简介

- SQLite数据库是一个关系型数据库，因为它很小，引擎本身只有一个大小不到 300KB的文件，所以常作为嵌入式数据库内嵌在应用程序中。
- SQLite生成的数据库文件是一个普通的磁盘文件，可以放置在任何目录下。
- SQLite数据库的管理工具很多，比较常用的有Navicat for SQLite 。



文件 编辑 查看 收藏夹 工具 窗口 帮助



连接



新建查询



表



视图



函数



用户



其它



SQL Server 备份



查询



自动运行



模型



图表



- Navicat Cloud
  - Office
  - Project MH-0015
- 总公司
  - ProjectDT-0052 (marybrown)
  - Production Server
    - 雇员信息
      - dbo
      - guest
      - HumanResources
      - 雇员
        - 表
        - 视图
        - 函数
        - 查询
- 我的连接
  - MongoDB
  - Oracle
  - SQL Server
  - MySQL
  - SQLite
  - PostgreSQL
  - MariaDB

对象

工作历史记录 @雇员信...

\*无标题 - 查询

打开表 设计表 新建表 删除表 导入 导出

名	行	创建日期	修改日期	OID
上班时间	3317	2010-06-11 12:21:17	2020-06-11 12:21:42	21375799
国家	25	2010-06-11 12:21:17	2020-06-11 12:21:42	597577167
地区	4	2010-06-11 12:21:17	2020-06-11 12:21:42	837578022
工作	19	2010-06-11 12:21:17	2020-06-11 12:21:42	1269579561
工作历史记录	10	2010-06-11 12:21:17	2020-06-11 12:21:44	1365579903
部门	27	2010-06-11 12:21:17	2020-10-17 10:55:28	1461580245
部门位置	23	2010-06-11 12:21:17	2020-06-11 12:21:43	1701581100
雇员	107	2010-06-11 12:21:17	2020-06-11 12:21:42	1749581271

信息 DDL 视图 模型 图表

工作历史记录  
表

- 总公司
- ProjectDT-0052
- Production Server
- 雇员信息
- 雇员

OID

1429580131

行

504

创建日期

2010-06-11 12:21:17

修改日期

2020-10-17 10:55:28

注释

Products sold or used in the manufacturing of sold products.



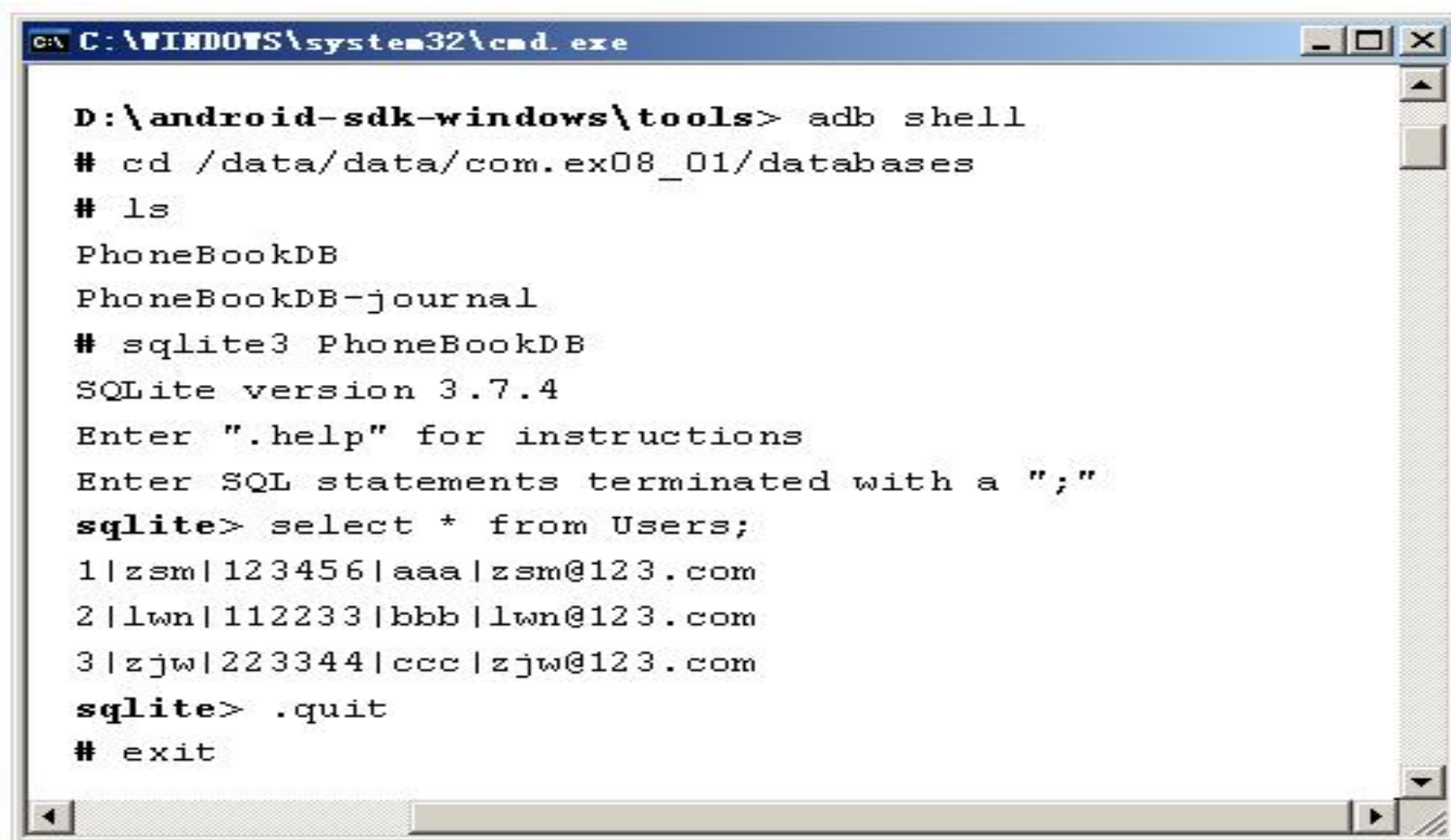
搜索

已选择 1 个对象



- 在Android 系统的内部集成了 SQLite数据库，所以 Android 应用程序可以很方便地使用 SQLite 数据库来存储数据。
- 在Android SDK的tools目录中有一个SQLite的命令行工具 `sqlite3.exe`。
- 下面简单说明其使用方法和步骤：

- (1) 在控制台窗口的命令行中输入“adb shell”，进入adb调试的shell命令行状态。在“#”提示符下输入“cd /data/data/<packageName>/databases”，（其中<packageName>是包名，如com.ex08\_01），进入到项目存放数据库文件的目录中。
- (2) 再输入“sqlite3 <DatabaseName>”（其中，DatabaseName是数据库名称，例如，sqlite3 phoneBookDB），进入SQL命令状态。
- (3) 在“sqlite>”提示符下输入SQL查询命令“select \* from users”（其中，users是数据库phoneBookDB中的数据表），则显示数据表users中的全部记录。
- (4) 输入“.quit”则退出SQL命令状态，再输入“exit”则退出shell命令行。



```
C:\WINDOWS\system32\cmd.exe

D:\android-sdk-windows\tools> adb shell
# cd /data/data/com.ex08_01/databases
# ls
PhoneBookDB
PhoneBookDB-journal
# sqlite3 PhoneBookDB
SQLite version 3.7.4
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> select * from Users;
1|zsm|123456|aaa|zsm@123.com
2|lwn|112233|bbb|lwn@123.com
3|zjw|223344|ccc|zjw@123.com
sqlite> .quit
# exit
```

- 可以通过DDMS工具将数据库文件复制到本地计算机上。
- 应用SQLite Expert Professional对数据库进行操作，完成后再通过DDMS工具放回到设备中，当需要操作大量数据时，这是比较方便的方法。



## 8.1.2 管理和操作SQLite数据库的对象

- Android提供了创建和使用SQLite数据库的API（Application Programming Interface,应用程序编程接口）。
- 在Android系统中，主要由类SQLiteDatabase和SQLiteOpenHelper对SQLite数据库进行管理和操作。

# 1、SQLiteDatabase类

方 法	说 明
openOrCreateDatabase(String path, SQLiteDatabase.CursorFactory factory)	打开或创建数据库
openDatabase(String path, SQLiteDatabase.CursorFactory factory, int flags)	打开指定的数据库
insert(String table, String nullColumnHack, ContentValues values)	新增一条记录
delete(String table,String whereClause, String[] whereArgs)	删除一条记录
query(String table,String[] columns, String selection, String[]selectionArgs, String groupBy,String having, String orderBy)	查询一条记录
update(String table,ContentValues values, String whereClause,String[] whereArgs)	修改记录
execSQL(String sql)	执行一条SQL语句
close()	关闭数据库

## 2、SQLiteOpenHelper类

**SQLiteOpenHelper** 是一个抽象类，要定义一个继承**SQLiteOpenHelper**的子类，并实现其方法。

方 法	说 明
onCreate (SQLiteDatabase)	首次生成数据库时候调用该方法。
onOpen (SQLiteDatabase)	调用已经打开的数据库。
onUpgrade (SQLiteDatabase, int, int)	升级数据库时调用。
getWritableDatabase()	读写方式创建或打开数据库。
getReadableDatabase()	创建或打开数据库。

## 8.1.3 SQLite数据库的操作命令

对数据库的操作有**3**个层次，各层次的操作内容有：

- 对数据库操作：建立数据库或删除数据库；
- 对数据表操作：建立、修改或删除数据库中的数据表；
- 对记录操作：对数据表中的数据记录进行添加、删除、修改、查询等操作。

# (1) 创建数据库

- 创建数据库的方法有多种，可以应用SQLiteDatabase对象openDatabase()方法及openOrCreateDatabase () 方法创建数据库；
- 也可以应用SQLiteOpenHelper的子类创建数据库；
- 还可以应用Activity继承于父类android.content.Context创建数据库的方法openOrCreateDatabase () 来创建数据库。

例如，要创建一个名称为**PhoneBook.db**的数据库，其数据库的结构为：

```
String TABLE_NAME = "Users";      //数据表名
String ID = "_id";                  //ID
String USER_NAME = "user_name";    //用户名
String ADDRESS = "address";        //地址
String TELEPHONE = "telephone";    //联系电话
String MAIL_ADDRESS = "mail_address"; //电子邮箱
```

则用Activity的openOrCreateDatabase () 方法创建数据库的代码如下:

```
SQLiteDatabase db;  
String db_name = "PhoneBook.db";  
String sqlStr =  
    "CREATE TABLE " + TABLE_NAME + " ("  
    + ID + " INTEGER primary key autoincrement, "  
    + USER_NAME + " text not null, "  
    + TELEPHONE + " text not null, "  
    + ADDRESS + " text not null, "  
    + MAIL_ADDRESS + " text not null "+ ");";  
int mode = Context.MODE_PRIVATE;  
db = this.openOrCreateDatabase(Database_name, mode, null);  
db.execSQL(sqlStr);
```

创建数据表的SQL语句

创建数据库

执行创建数据库的SQL语句

- 这时，通过DDMS可以看到，在/data/data/xxxx（包名）/databases路径下，创建了数据库PhoneBook.db。

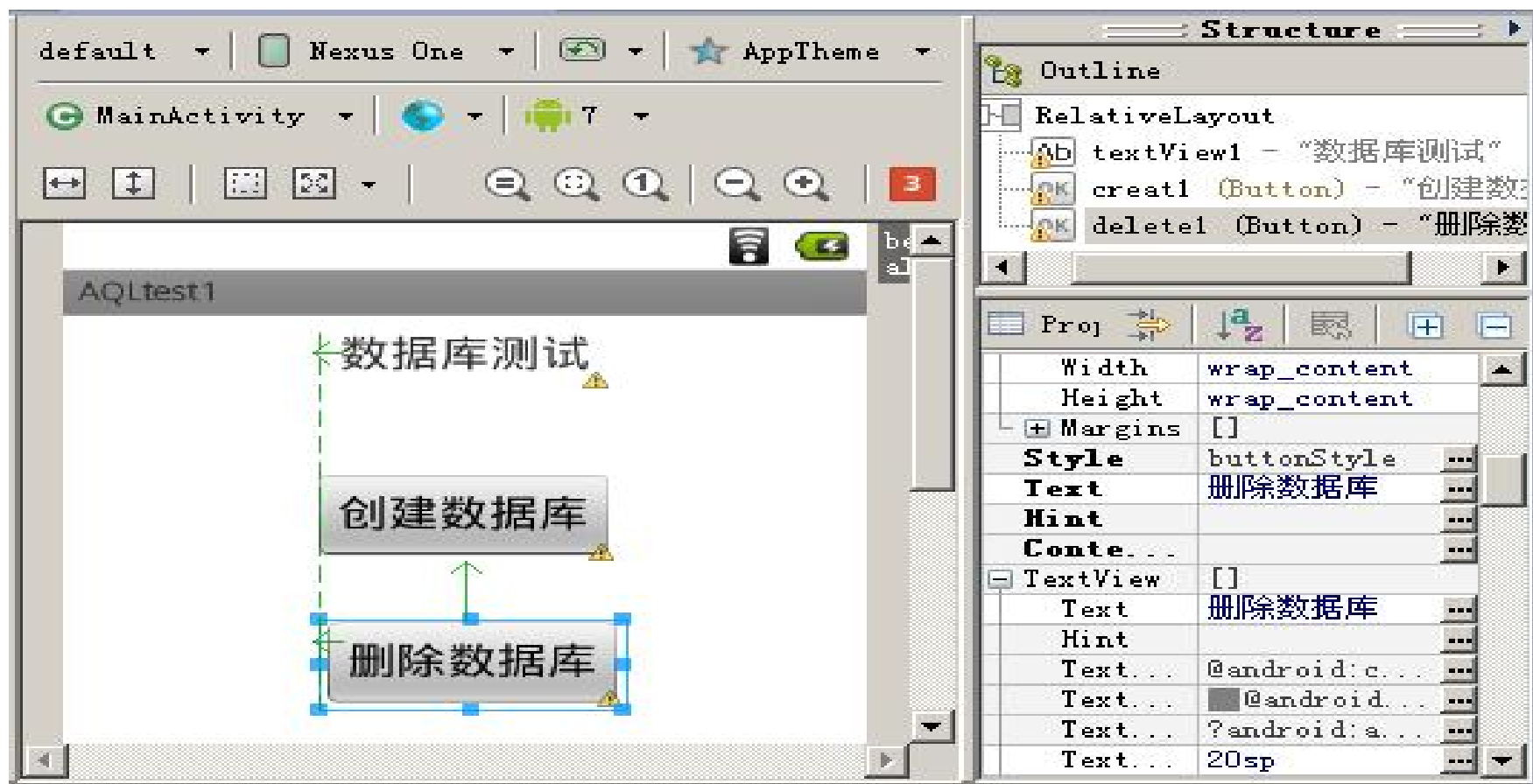
## (2) 删除数据库

- 当要删除一个指定的数据库文件时，需要应用android.content.Context类的deleteDatabase(String name)方法来删除这个指定的数据库。

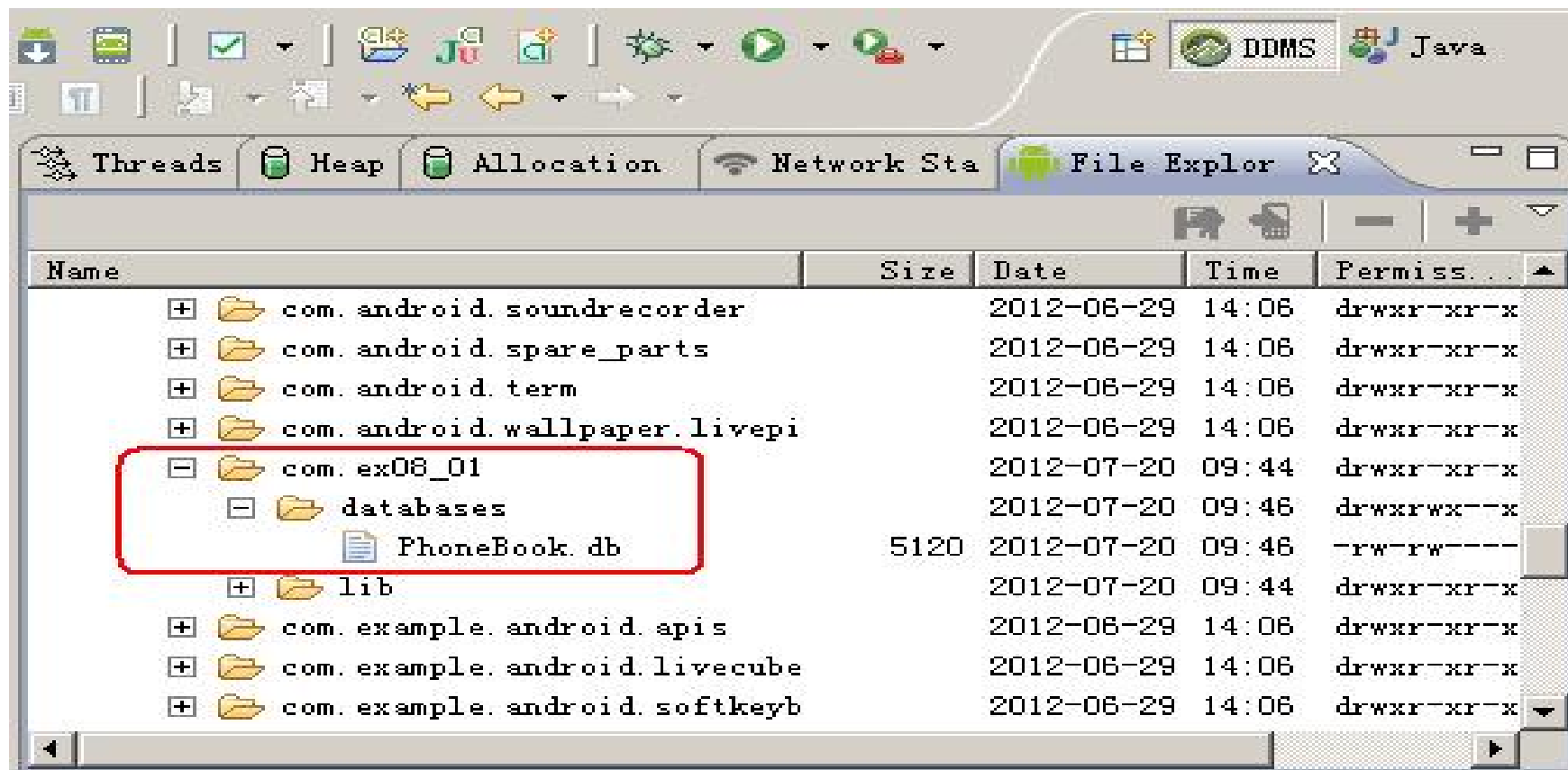


## 【例8-1】编写一个创建与删除数据库的演示程序。

- (1) 用户界面设计



- 运行程序，当单击“创建数据库”按钮后，通过DDMS工具调试监控视图，在/data/data/项目包名/databases路径下，可以看到创建的数据库PhoneBook.db文件。如图8.4所示。再单击“删除数据库”按钮，数据库文件则被删除。



## 2、数据表操作

### (1) 创建数据表

创建数据表步骤为：

- 用SQL语句编写创建数据表的命令；
- 调用SQLiteDatabase的execSQL()方法执行SQL语句。

## (2) 删除数据表

- 删除数据表的步骤与创建数据表类似，先编写删除表的SQL语句，再调用execSQL()方法执行SQL语句。
- 例如，要删除名为Users数据表，则删除数据表Users的SQL语句

```
String sql = "DROP TABLE Users";  
db.execSQL(sql);
```



执行SQL语句

### 3、对数据记录操作

- 在数据表中，把数据表的列称为字段，数据表的每一行称为记录。对数据表中的数据进行操作处理，主要是对其记录进行操作处理。
- 对数据记录的操作处理有两种方法。
  - 一种方法是编写一条对记录进行增、删、改、查的SQL语句，通过execSQL()方法来执行。
  - 另一种方法是使用Android系统SQLiteDatabase对象的相应方法进行操作。

# (1) 新增记录

新增记录的方法是使用SQLiteDatabase对象的insert()方法。

在insert(String table, String nullColumnHack, ContentValues values)方法中的3个参数其意义为：

- 第1个参数table：增加记录的数据表；
- 第2个参数nullColumnHack：空列的默认值，通常为null；
- 第3个参数ContentValues：为ContentValues对象，其实就是一个键值对的字段名称，键名为表中的字段名，键值为要增加的记录数据值。通过ContentValues对象的put ()方法把数据存放到ContentValues对象中。

## (2) 修改记录

- 修改记录是使用SQLiteDatabase对象的update()方法。
- 在update(String table, ContentValues values, String whereClause, String[] whereArgs)方法中有4个参数:
  - 第1个参数table: 修改记录的数据表;
  - 第2个参数ContentValues: ContentValues对象, 存放已作修改的数据的对象;
  - 第3个参数whereClause: 修改数据的条件, 相当于SQL语句的where子句;
  - 第4个参数whereArgs: 修改数据值的数组。

## (3) 删除记录

- 删除记录使用SQLiteDatabase对象的delete()方法。
- 在delete(String table,String whereClause, String[] whereArgs)方法中有3个参数:
  - 第1个参数table: 修改记录的数据表;
  - 第2个参数whereClause: 删除数据的条件, 相当于SQL语句的where子句;
  - 第3个参数whereArgs: 删除条件的参数数组。



## (4) 查询记录

- 在数据库的操作命令中，查询数据的命令是最丰富最复杂的。  
在SQLite 数据库中，使用SQLiteDatabase对象的query()方法查询数据。
- query(String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy)方法有7个参数，其意义为：

- 第1个参数table: 查询记录的数据表;
- 第2个参数columns: 查询的字段, 如为null, 则为所有字段;
- 第3个参数selection: 查询条件, 可以使用通配符“?”;
- 第4个参数selectionArgs: 参数数组, 用于替换查询条件中的“?”;
- 第5个参数groupBy: 查询结果按指定字段分组;
- 第6个参数having: 限定分组的条件;
- 第7个参数orderBy: 查询结果的排序条件。

## (5) 对查询结果cursor的处理

- query()方法查询的数据均封装到查询结果Cursor对象之中，Cursor相当于SQL语句中resultSet结果集上的一个游标，可以向前向后移动。Cursor对象的常用方法有：

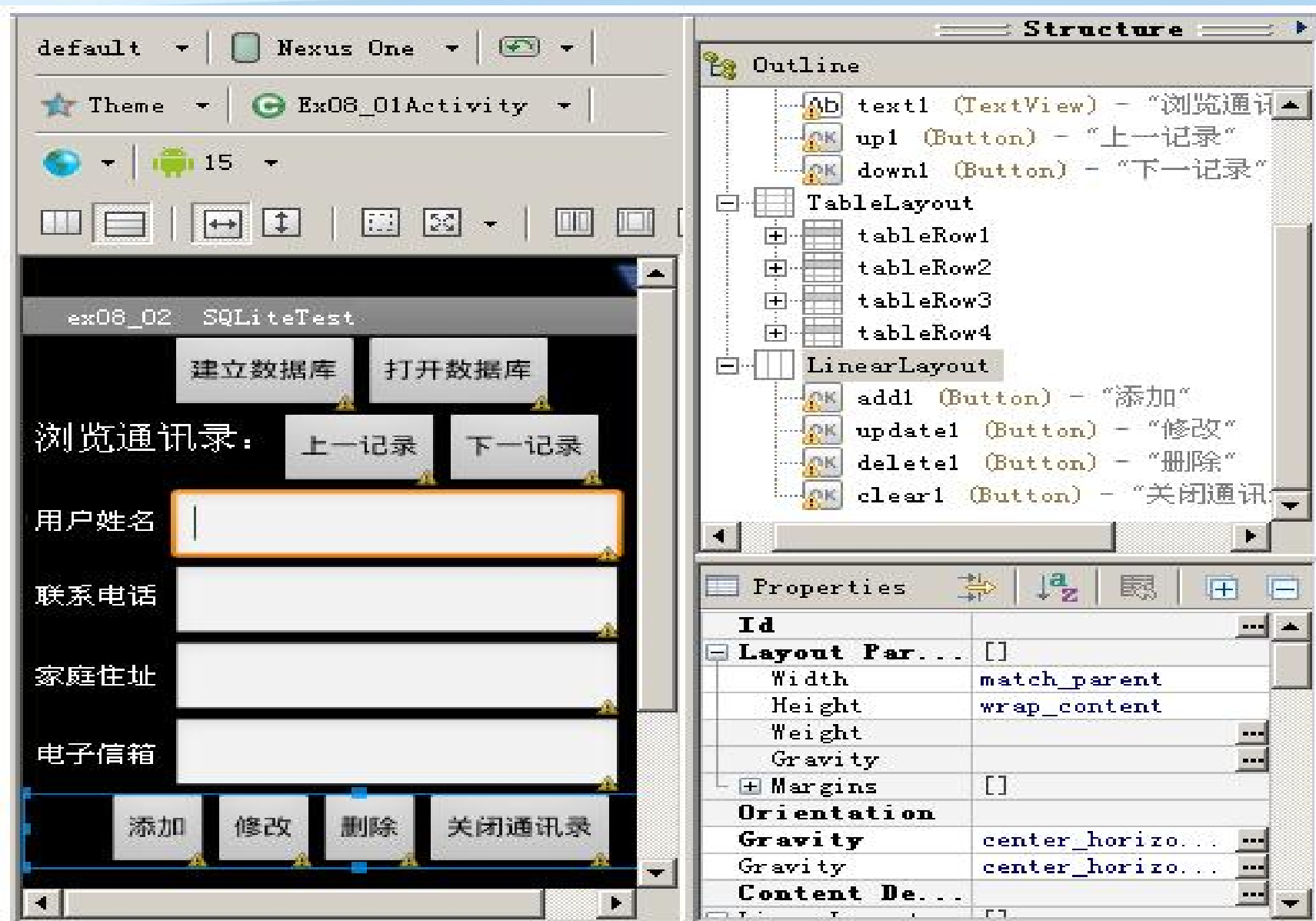
- moveToFirst(): 移动到第一行;
- moveToLast(): 移动到最后一行;
- moveToNext(): 向前移动一行;
- MoveToPrevious(): 向后移动一行;
- moveToPosition(positon): 移动到指定位置;
- isBeforeFirst(): 判断是否指向第1条记录之前;
- isAfterLast(): 判断是否指向最后一条记录之后。

**【例8-2】** 编写程序，建立一个通讯簿，可以向前向后浏览数据记录，也可以添加、修改、删除数据。

### (1) 用户界面设计

在界面设计中，用一个垂直线性布局嵌套了三个水平线性布局和一个表格布局，从而把整个界面划分为4部分。

- 在第一个水平线性布局中嵌套了“建立数据库”和“打开数据库”按钮；
- 在第二个水平线性布局中嵌套了“浏览通讯录”文本标签、“建立数据库”和“打开数据库”按钮；
- 在第三个水平线性布局中嵌套了“添加”、“修改”、“删除”和“关闭通讯录”按钮；
- 在表格布局中设置表格为4行2列，每一行均包含一个文本标签和一个文本编辑框。



建立数据库

打开数据库

浏览通讯录：

上一记录

下一记录

用户姓名：

abc

联系电话：

123456

家庭住址：

123456

电子信箱：

abc@123.com

添加

修改

删除

关闭通讯录

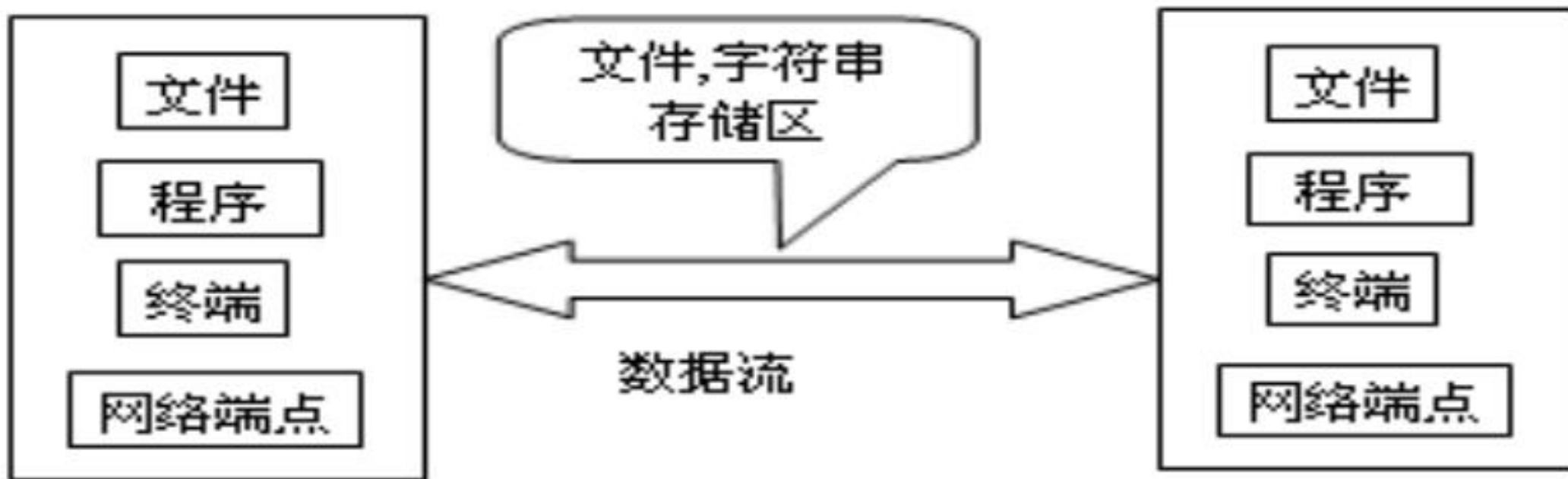
## 8.2 文件处理



## 8.2.1 输入/输出流

- 程序可以理解为数据输入、输出以及数据处理的过程，程序执行过程中，通常需要读取处理数据，并且将处理后的结果保存起来。Android系统提供了对数据流进行输入输出的方法。

# 1、流的概念



## 2、文件与目录管理的File类

- Android系统处理文件时直接调用java语言的java.io包中的File类。每个File类的对象都对应了系统的一个文件或目录，所以创建File类对象时需指明它所对应的文件或目录名。
- File类共提供了三个不同的构造函数，以不同的参数形式灵活地接收文件和目录名信息。

- File ( String path )
  - 这个构造方法的字符串参数path指明了新创建的File对象对应的文件及其路径名。

```
File f1 = new File (“\data\data\jtest”);  
File f2 = new File(“testfile.dat”);
```

- `File(String path, String name)`

- 第二个构造方法有两个参数，`path`表示所对应的文件或目录的绝对或相对路径，`name`表示文件或目录名。将路径与名称分开的好处是相同路径的文件或目录可共享同一个路径字符串，这样管理和修改都较方便。

```
File f4 = new File(" \\sdcard", "file.dat");
```

- File(File dir, String name)

- 第三个构造方法使用另一个已有的某SD卡目录的File对象作为第一个参数，表示文件或目录的路径，第二个字符串参数表述文件或目录名。

```
String sdir = "data" + System.dirSep + "jtest";
```

```
String sfile = "FileIO.data";
```

```
File Fdir = new File ( sdir );
```

```
File Ffile = new File ( Fdir, sfile );
```

# File类的常用方法

方 法	说 明
exists( )	判断文件或目录是否存在
isFile( )	判断对象是否是文件
isDirectory( )	判断对象是否是目录
getName( )	返回文件名或目录名
getPath( )	返回文件或目录的路径
length( )	返回文件的字节数
renameTo( File newFile )	将文件重命名成newFile对应的文件名
delete( )	将当前文件删除
mkdir( )	创建当前目录的子目录

### 3、文件输入输出流

- 在Android中，处理二进制文件使用字节输入输出流，处理字符文件使用字符输入输出流。对文件进行输入输出处理有 四个类：
  - FileInputStream ： 字节文件输入流；
  - FileOutputStream ： 字节文件输出流；
  - FileReader ： 字符文件输入流；
  - FileWriter ： 字符文件输出流。



## 8.2.2 处理文件流

### 1、文件输出流保存文件

- (1) **FileOutputStream**类

**FileOutputStream**类是从**OutputStream**类派生出来的输出类，它具有处理向文件中写数据的能力。它的构造方法有以下三种形式：

- **FileOutputStream( String filename )**
- **FileOutputStream( File file )**
- **FileOutputStream(FileDescriptor fdObj)**

- (2) 把字节发送到文件输出流的write()方法

输出流只是建立了一条通往数据要去的目的地的通道，数据并不会自动进入输出流通道，我们要使用文件输出流的write()方法把字节发送到输出流。

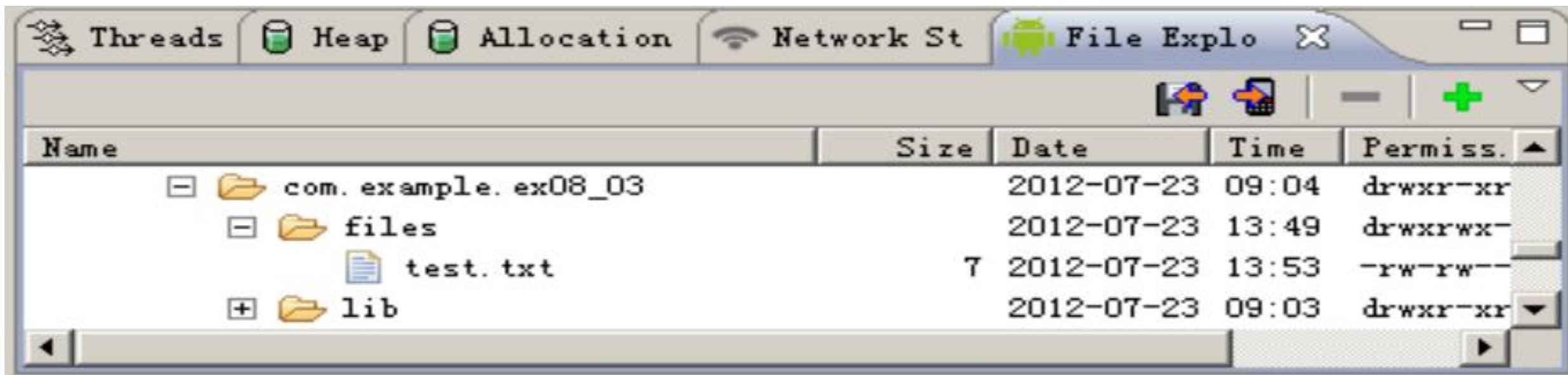
- 使用write()方法有三种格式：

- write(int b): 将指定字节写入此文件输出流。
- write(byte[] b) : 将 b.length 个字节从指定字节数组写入此文件输出流中。
- write(byte[] b, int off, int len) : 将指定字节数组中从偏移量 off 开始的 len 个字节写入此文件输出流。

## 【例8-3】 把字符串“Hello World!”保存到本地资源的test.txt文件中。

```
1 void savefile()
2 {
3     String fileName="test.txt";
4     String str = "Hello World!";
5     FileOutputStream f_out;
6     try {
7         f_out = openFileOutput(fileName,
                                Context.MODE_PRIVATE);
8         f_out.write(str.getBytes());
9     }
10    catch (FileNotFoundException e)
11        {e.printStackTrace();}
12    catch (IOException e) {e.printStackTrace();}
13 }
```

- 文件test.txt保存在/data/data/<包名>/files/目录之下。应用DDMS工具可以查看到保存在本地资源目录下的文件，如图所示。



## 2、文件输入流读取文件

### (1) FileInputStream类

- **FileInputStream**类是从**InputStream** 类中派生出来的输入流类, 它用于处理二进制文件的输入操作。
- 它的构造方法有下面三种形式:
  - `FileInputStream(String filename);`
  - `FileInputStream(File file);`
  - `FileInputStream( FileDescriptor fdObj);`

## (2) 从文件输入流中读取字节的read()方法

- 文件输入流只是建立了一条通往数据的通道，应用程序可以通过这个通道读取数据，要实现读取数据的操作，需要使用read()方法。
- 使用read()方法有三种格式：
  - `int read( );`
  - `int read( byte b[ ] );`
  - `int read( byte b[ ],int off, int len);`

## 【例8-4】 读取本地资源文件test.txt中的内容。

```
1 void readfile()
2 {
3     String fileName="test.txt", str ;
4     byte[] buffer = new byte[1024];
5     FileInputStream in_file=null;
6     try {
7         in_file = openFileInput(fileName);
8         int bytes = in_file.read(buffer);
9         str = new String(buffer, 0, bytes);
10        Toast.makeText(MainActivity.this,
11            "文件内容: " + str, Toast.LENGTH_LONG).show();
12    }
13    catch (FileNotFoundException e) { System.out.print("文件不存在");}
14    catch (IOException e) { System.out.print("IO流错误");
15 }
```

# 3、对SD卡文件的读写

## (1) 环境变量访问类Environment

- Environment为提供对环境变量访问类，在Android程序中对SD卡文件进行读写操作时，经常需要应用它的以下两个方法：
  - `getExternalStorageState()`: 获取当前存储设备状态;
  - `getExternalStorageDirectory()`: 获取SD卡的根目录。



## (2) 读写SD卡的权限

- Environment的主要常量为Environment.MEDIA\_MOUNTED , 表示对SD卡具有读写权限。
- 判断是否具有对SD卡文件进行读写操作权限通常使用下列条件语句:

```
if(Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED))
```

- 在AndroidManifest.xml文件中，要加入允许对SD卡进行操作的权限语句。
  - 允许在SD卡中创建及删除文件的权限语句：

```
<uses-permission  
android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS">  
    </uses-permission>
```
  - 允许往SD卡中写入数据的权限语句：

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

# 【例8-5】 读取与保存文件的应用程序示例。

- 新建应用项目，设置1个文本编辑框和4个按钮，按钮分别为“保存文件”、“保存到SD卡”、“读取资源文件”、“读取SD卡文件”。
- 在AndroidManifest.xml文件中，加入允许对SD卡进行创建文件和写入数据的权限语句。



(代码详见教材)

## 8.3 轻量级存储SharedPreferences

- Android系统提供了一个存储少量数据的轻量级的数据存储方式 **SharedPreferences**。该存储方式类似于Web程序中的**Cookie**，通常用它来保存一些配置文件数据、用户名及密码等。**SharedPreferences**采用“键名-键值”的键值对形式组织和管理数据，其数据存储存储在**XML**格式文件中。

- 使用SharedPreferences方式存储数据需要用到SharedPreferences和SharedPreferences.Editor接口，这两个接口在android.content包中。
- SharedPreferences对象由Context.getSharedPreferences (String name, int mode)方法构造，。

## SharedPreferences接口的常用方法

方 法	说 明
edit()	建立一个SharedPreferences.Editor对象
contains(String key)	判断是否包含该键值
getAll()	返回所有配置信息
getBoolean(String key,Boolean defValue)	获得一个Boolean类型数据
getFloat(String key,float defValue)	获得一个Float类型数据
getInt(String key,int defValue)	获得一个Int类型数据
getLong(String key,long defValue)	获得一个Long类型数据
getString(String key,String defValue)	获得一个String类型数据

## SharedPreferences.Editor接口的常用方法

方 法	说 明
clear()	清除所有数据值
commit()	保存数据
putBoolean(String key, boolean value)	保存一个Boolean类型数据
putFloat(String key, float value)	保存一个Float类型数据
putInt(String key, int value)	保存一个Int类型数据
putLong(String key, long value)	保存一个Long类型数据
putString(String key, String value)	保存一个String类型数据
remove(String key)	删除键名key所对应的数据值



- **SharedPreferences.Editor**的**putXXX**方法以键值对的形式存储数据，最后一定要调用**commit**方法提交数据，文件才能保存。
- 读取数据非常简单，直接调用**SharedPreferences**对象相应的**getxxx**方法即可获得数据。

## 【例8-6】应用SharedPreferences对象将一个客户的联系电话保存到电话簿中。

- 设客户名为zsm，其电话为123456。故设电话簿的文件名为phoneBook，其数据的“键名-键值”对为（“name”，“zsm”）和（“phone”，“123456”）。
- 数据文件phoneBook.xml保存在/data/data/com.example.ex08\_06（包名）/shared\_prefs /目录之下，扩展名.xml由系统自动生成。应用DDMS工具可以查看到该文件，如图所示。

Threads Heap Allocation T Network Stat File Explore X					
Name Size Date Time Permiss...					
[-] com.example.ex08_06		2012-07-24	13:52	drwxr-xr-x	
[+] lib		2012-07-24	13:51	drwxr-xr-x	
[-] shared_prefs		2012-07-24	13:52	drwxrwx--x	
phoneBook.xml	140	2012-07-24	13:52	-rw-rw----	



- 单击DDMS工具的按钮，可以将数据文件phoneBook.xml从模拟器中保存到计算机的磁盘中，打开该XML格式的数据文件phoneBook.xml，其内容如下：

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<string name="phone">123456</string>
<string name="name">zsm</string>
</map>
```

## 8.4访问远程数据库

## 【例8-7】 编写一个访问远程数据库的应用程序。

- (1) 设有MySQL数据库，数据库名为testdb。该数据库中有数据表user，其数据结构如表8-6所示。

表 8-6 数据表 user 的数据结构

字段名	字段类型	说明
si d	i nt ( 5)	序号
name	Var char ( 10)	姓名
email	Var char ( 25)	邮箱

## (2) 界面布局设计

在界面中设置一个按钮Button和一个列表组件ListView。

(3) 把volley.jar复制到项目的app\libs目录下，并完成jar包的安装。

(4) 主类MainActivity.java设计（代码见教材）

(5) 修改配置文件，添加访问网络权限

(6) 服务器端连接数据库的PHP文件conn\_testdb.php





Q&A

谢谢大家！