УНИВЕРСИТЕТ ИТМО

Базы данных SQLite. Работа с базами данных.

Санкт-Петербург, 2016

Варианты хранения данных:

Shared Preferences (Общие настройки)

Приложение автоматически создаёт файл в своей папке и хранит простые данные в виде «ключ — значение», для примитивных типов.

Internal Storage (Внутренняя память/хранилище)

Часть пространства встроенной flash-памяти, отведенная для установки приложений (apk-файлов) их баз данных, настроек и других локальных файлов. Все Android-телефоны имеют “Внутреннюю память”.

External Storage (Внешняя память/хранилище)

Сохранение данных на внешней памяти устройства.

SQLite Databases (База данных SQLite)

Хранение структурированных данных в базе данных.

Network Connection (Подключение к сети)

Хранение в сетевом сервером. Вы можете использовать сеть (если она доступна) для хранения и извлечения данных из собственных веб-служб. Для сетевых операций, классы использования в следующих пакетах:

java.net.*, android.net.*

Настройки (Shared preferences)

- 2 варианта загрузки настроек в приложение:

– getSharedPreferences() - используется, если необходимо несколько файлов настроек, определяемых именем (первый параметр);

– getPreferences() - используется, если нужен только один файл настроек. Имя не указывается.

- Для изменения настроек необходимо получить объект SharedPreferences.Editor
- Добавление значений производиться методами типа:
putBoolean(String key, boolean value); putString(String key, String value);
- Сохранение настроек производиться методом commit().
- Для чтения значений используются методы типа:
getBoolean(String key, boolean defValue); getString(String key, String defValue);

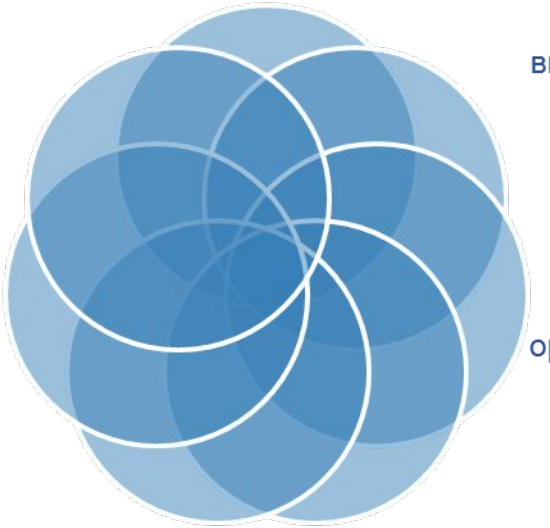
Настройки (Shared preferences)

```
public class Calc extends Activity {
    public static final String PREFS_NAME = "MyPrefsFile";
    @Override
    protected void onCreate(Bundle state) {
      super.onCreate(state);
      . . .
      // Восстановление настроек
      SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
      boolean silent = settings.getBoolean("silentMode", false);
      setSilent(silent);
    }

    @Override
    protected void onStop() {
      super.onStop();
      // Получаем объект Editor для изменения настроек
      SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
      SharedPreferences.Editor editor = settings.edit();
      editor.putBoolean("silentMode", mSilentMode);
      // Сохранить изменения!
      editor.commit();
    }
}
```

Внутренняя память приложения

- По-умолчанию файлы во внутренней памяти недоступны другим приложениям и пользователю.
- Для чтения из файла — метод read().
- Файлы удаляются вместе с приложением.
- Для открытия файла на чтение метод FileInputStream openFileInput(String name).
- Для создания файла применяется метод FileOutputStream openFileOutput(String name, int mode).
- Для закрытия файла — метод close().
- Для записи в файл — метод write().



Внешняя память

- SD-карта памяти или встроенная память для хранения информации.
- Файлы доступны всем и могут быть изменены (или удалены) пользователем.
- Метод для проверки доступности внешней памяти: getExternalStorageState()

```
boolean mExternalStorageAvailable = false;
boolean mExternalStorageWriteable = false;
String state = Environment.getExternalStorageState();

if (Environment.MEDIA_MOUNTED.equals(state)) {
    // Можно читать и записывать медиа файлы
    mExternalStorageAvailable = mExternalStorageWriteable = true;
} else if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
    // Можно только читать медиа файлы
    mExternalStorageAvailable = true;
    mExternalStorageWriteable = false;
} else {
    // Что-то произошло. Это может быть одно из множества состояний,
    // но точно известно, что нет возможности ни записывать, ни читать файлы
    mExternalStorageAvailable = mExternalStorageWriteable = false;
}
```

SQLite

- База данных с открытым исходным кодом.
- Поддерживает SQL, транзакции и процедуры.
- Не требует много ресурсов (примерно 250 кБ памяти).
- Является встраиваемой библиотекой.
- 3 основных типа данных: TEXT, INTEGER, REAL.
- Всё содержимое БД храниться в одном файле.

Особенности SQLite в Android

- SQLite интегрирован во все Android устройства.
- Не требуется установка или настройка БД.
- Необходимо определить только процедуры создания и обновления БД.
- Выполняется обращение к файлам устройства.
- Желательно выполнять запросы асинхронно (в отдельном потоке).

Виды SQL запросов

Обращения к базе данных SQL выполняются посредством запросов, существует три основных вида SQL запросов: DDL, Modification и Query.

DDL-запросы

• DDL запросы.

Такие запросы используются для создания таблиц. Каждая таблица характеризуется именем и описанием столбцов, которое содержит имя столбца и тип данных.
Пример запроса для создания таблицы:

```
create Table_Name (_id integer primary key autoincrement,
                   field_name_1 text,
                   field_name_2 text);
```

Первый столбец обозначен, как **primary key** (первичный ключ), т.е. уникальное число, которое однозначно идентифицирует строку.
Слово **autoincrement** указывает, что база данных будет автоматически увеличивать значение ключа при добавлении каждой записи, что и обеспечивает его уникальность.

Modification-запросы

- Modification запросы.

Такие запросы используются для добавления, изменения или удаления записей.

Пример запроса на добавление строки:

```
insert into Table_Name values(null, value1, value2);
```

В этом случае значения разместятся в соответствующие столбцы таблицы, первое значение задается для поля _id и равно null, т. к. SQLite вычисляет значение этого поля самостоятельно. При добавлении можно указывать столбцы, в которые будут размещаться значения, остальные столбцы заполнятся значениями по умолчанию, в этом случае можно добавлять элементы в измененном порядке.

Query -запросы

• Query запросы. Такие запросы позволяют получать выборки из таблицы по различным критериям.

Пример запроса:

```
select from Table_Name where (_id = smth);
select Field_Name_1,
Field_Name_2 from Table_Name
Field_Name_1 = smth);
```

Первый запрос выводит строку с _id равным smth, второй - вы- водит два элемента Field_Name_1 и Field_Name_2 строк, в которых Field_Name_1 равен smth.

Создание и обновление БД

- Создать класс-наследник класса SQLiteOpenHelper.
- Вызвать конструктор родителя с параметрами: имя_бд и версия_бд.
- Переопределить методы onCreate() и onUpgrade().
- Параметром методов является класс SQLiteDatabase.
- getReadableDatabase() и getWritableDatabase() предоставляют доступ к БД.

Класс SQLiteDatabase

- Базовый класс для работы с БД SQLite.
- Предоставляет методы для выполнения запросов к БД, открытия/обновления/закрытия БД.
- insert(), update(), delete(), query().
- execSQL().
- Запросы к БД выполняются через
  – rawQuery()
  – query()
  – SQLiteQueryBuilder класс

Параметры метода query()

Параметр	Описание
String dbName	Название таблицы, для которой выполняется запрос.
String[] columnNames	Список столбцов, которые нужно вернуть. null — все столбцы.
String whereClause	Раздел where sql-запроса, фильтрует результат. null — все данные.
String[] selectionArgs	Знаки «?» в разделе where заменяются значениями из массива selectionArgs.
String[] groupBy	Определяет группировку столбцов.
String[] having	Фильтр групп
String[] orderBy	Столбцы, по которым данные сортируются

Класс ContentValues

- Определяет пары ключ/значение.
- Ключ — имя столбца таблицы.
- Значение — содержимое записи в данном столбце.
- Используется для добавления/обновления данных в БД.

Для добавления новых строк в таблицу используется класс ContentValues, каждый объект этого класса представляет собой одну строку таблицы.

Для получения результатов запросов к базе данных используется класс Cursor, объекты этого класса ссылаются на результирующий набор данных.

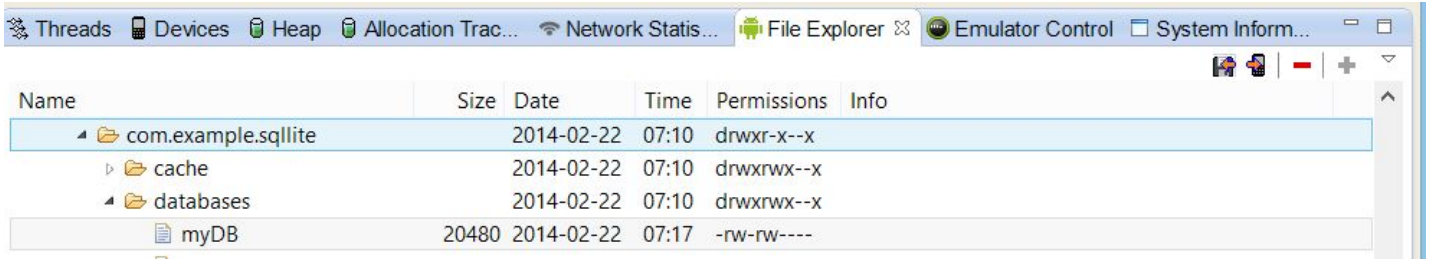
Класс Cursor

- Запрос возвращает объект типа Cursor.
- Cursor указывает на одну строку результата.
- getCount().
- moveToFirst(), moveToNext(), isAfterLast().
- Типизированные методы get*() для извлечения данных (getLong(columnIndex), getString(columnIndex)).
- getColumnIndex(columnName)
- close()

База данных SQLite

Это обычный файл, копирование и перемещение которого не отражается на работе базы данных. Android хранит файл базы данных в папке data/data/packagename/databases/

Для доступа к этому файлу необходимо запускать команды SQL.



Дополнительный материал

Полезные ссылки
https://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html
http://blog.reigndesign.com/blog/using-your-own-sqlite-database-in-android-applications/
http://www.vogella.com/tutorials/AndroidSQLite/article.html
http://microsin.net/programming/android/saving-files.html
http://sqlitebrowser.org
https://habrahabr.ru/post/125883/
http://www.enterra.ru/blog/android_issues_with_sqlite/

Работа в консоли sqlite.exe
http://developer.alexanderklimov.ru/android/sqlite/cathouse.php