Інформаційні технології

УДК 004.657

Намінас Владислав Вікторович
бакалавр комп'ютерних наук
Національного технічного університету України
«Київський політехнічний інститут імені Ігоря Сікорського»
Зацепін Олексій Артемович
бакалавр з безпеки інформаційних та комунікаційних систем
Національного технічного університету України
«Київський політехнічний інститут імені Ігоря Сікорського»

Наминас Владислав Викторович
бакалавр компьютерных наук
Национального технического университета Украины
«Киевский политехнический институт имени Игоря Сикорского»
Зацепин Алексей Артёмович
бакалавр по безопасности информационных и коммуникационных систем
Национального технического университета Украины
«Киевский политехнический институт имени Игоря Сикорского»

Naminas Vladyslav
Bachelor of computer science
The National Technical University of Ukraine
«Igor Sikorsky Kyiv Polytechnic Institute»
Zatsepin Alexey
Bachelor of security of information and communication systems
The National Technical University of Ukraine
«Igor Sikorsky Kyiv Polytechnic Institute»

ПОРІВНЯННЯ ORM-РІШЕНЬ ДЛЯ ОС ANDROID
СРАВНЕНИЕ ORM-РЕШЕНИЙ ДЛЯ ОС ANDROID
COMPARISON OF ORM SOLUTIONS FOR ANDROID OPERATING SYSTEM

Анотація: Проведено порівняння ORM-бібліотек для операційної системи Android.

Ключові слова: ORM, Android, SQLite, продуктивність.

Аннотация: Произведено сравнение ORM-библиотек для операционной системы Android.

Ключевые слова: ORM, Android, SQLite, производительность.

Summary: Comparison of ORM-libraries for the Android operating system.

Key words: ORM, Android, SQLite, performance.

Введення.

На даний момент існує чималоORM-бібліотек для ОС Android. Щоб вибрати з них ту, що найбільше підходить для вашого проекту, требати порівняти їх за різними параметрами.

В якості бібліотек для порівняння були вибрані найбільш популярні на момент написання рішення: *ORMLite, Sugar ORM, Freezer, DBFlow, requery, GreenDAO, ActiveAndroid, Room, Sprinkles*. Вони порівнювалися між собою та із вбудованним API для роботи з SQLite в Android, а також з популярним NoSql рішенням — *Realm*.

Основні поняття

Android — це мобільна операційна система, розроблена компанією Google Inc. Вона використовується більшістью смартфонами і планшетами (на момент написання статті). Приклади включають Sony Xperia, Samsung Galaxy і Google Nexus One.

SQLite [1] — це вбудований механізм СУБД SQL. На відміну від більшості інших баз даних SQL, SQLite не має окремого процесу сервера. SQLite читає і записує безпосередньо в звичайні файли на диску.ОС Android за замовчанням має вбудований інструментарій для підтримки sqlite3 в своїх додатках.

ORM (Object-Relational Mapping - об'єктно-реляційне відображення) — це технологія програмування, яка зв'язує бази даних з концепціями об'єктно-орієнтованих мов програмування, створюючи «віртуальну об'єктну базу даних».

Тобто це зручний спосіб програмно працювати з реляційною базою даних, (в більшості випадків) без необхідності використання SQL. Об'єктно-реляційне відображення — це відображення вигляду:

таблиця <-> клас, рядок таблиці <-> об'єкт класу.

Порівняння продуктивності.

Порівняння швидкості роботи проводилось з урахуванням чітких вимог.

Створення об'єктів відбувалося за допомогою спеціально реалізованого рандомайзера.

Всього було розглянуто 3 варіанти заповнення і для кожного з них кожна з CRUD-операцій — write, read, update і delete.

1) Перший варіант — *"simple"*:

```
public class Library{
        String address;
        String name;
}
public class Book{
        String author;
        String title;
        int pagesCount;
        int bookId;
        Library library;
}
```

Book має залежність багато до одного (many to one) до Library. 1 об'єкт Library на 1000 об'єктів Book. Для write, read, update і delete запитів параметри однакові — 1 об'єкт Library на 1000 об'єкт ів Book.

2) Другий варіант — *"complex"*:

Додається клас Person:

```
public class Person{
        String firstName;
        String secondName;
        Date birthdayDate;
        String gender;
        long phone;
        Library library;
}
```

Person має залежність багато до одного (many to one) до Library. 1 об'єкт Library на 500 об'єктів Book та 400 об'єктів Person. Для write, read, update і delete запитів параметри однакові — 5 об'єктів Library на 2500 об'єктів Book і 2000 об'єктів Person.

3) Третій варіант — *"balanced"*:

Залежності залишаються ті ж самі. 1 об'єкт Library на 50 об'єктів Book та 50 об'єктів Person. Для write, read, update і delete запитів параметри однакові — 50 об'єктів Library на 2500 об'єктів Book і 2500 об'єктів Person.

Вимоги та обмеження:

- для кожного об'єкта повинні існувати значення всіх полів (тобто значення полів не повинні бути порожніми); ця умова також поширюється на зв'язані (за допомогою foreign key constraint) об'єкти. Для перевірки виконання даної вимоги при кожній операції зчитування викликається функція, що виконує цю перевірку.
- кеш не бере участі у порівнянні. Щоб виключити можливість завантаження даних з кешу, після перевірки на запис додаток закривається і запускається заново вручну; після операції зчитування все зчитані дані відразу ж видаляються; кешування

відключається явно, якщо використовується бібліотекою за замовчуванням.

- для того, щоб операції ініціалізації бази даних не впливали на результат операції записи, проводиться "розігрів" (warming up) - згенеровані рандомний дані записуються, зчитуються і видаляються з бази даних.

Результати порівняння знаходяться в таблиці 1.

Таблиця 1. Порівняння продуктивності

Бібліотека	write (s)	read (s)	update (s)	delete (s)	write (c)	read (c)	update (c)	delete (c)	write (b)	read (b)	update (b)	delete (b)
ORMLite	151	**666**	122	105	445	3836	857	811	1563	**3426**	724	728
SugarORM	245	842	252	152	1402	4129	1467	1003	2204	4397	1702	1197
Freezer	248	5430	240	4797	1337	78982	2221	22104	3255	134942	1887	29515
DBFlow	97	757	459	186	360	3534	3124	1044	1129	4653	5204	1268
Requery	87	1501	147	129	461	8057	861	802	1368	8002	886	763
Realm	151	29	1079	723	698	688	19666	9180	1522	210	21129	10006
GreenDAO	**81**	1238	**117**	**97**	**357**	5552	**455**	**274**	**598**	5905	**504**	**315**
ActiveAndroid	3123	930	2293	2423	14671	4165	15958	13023	17213	4653	19303	14642
Sprinkles	5766	1050	6364	605	25978	4334	65579	2428	27774	4526	37705	2519
Room	131	699	170	109	562	**3201**	717	403	1330	3532	790	507
SQLite	**50**	**436**	**63**	**80**	**386**	**2155**	**192**	**284**	**1146**	**2313**	**213**	**318**

У таблиці 1(s) означає simple, (b) — balanced та (c) — complex.

Діаграми можна знайти за посиланням:

https://github.com/AlexeyZatsepin/Android-ORM-benchmark/tree/master/results

Порівняння за іншими параметрами показано в таблиці 2.

Таблица 2. Порівняння за іншими параметрами

Бібліотека	Зручність використання (суб'єктивно, від 1 до 5)	Розмір бібліотеки	Кешування (+/-)	Відкладена ініціалізація (+/-)	Підтримка RXJava (+/-)	Документація	Дата останнього оновлення
ORMLite	3	122 KB	+	+	-	5	Mar 16, 2017
SugarORM	5	718 KB	-	-	-	4	Nov 9, 2017
Freezer	5	300 KB	-	-	+	4	Apr 18, 2017
DBFlow	5	178 KB	+	+	+	5	May 12, 2017
Requery	4	800 KB	+	+	+	3	May 15, 2017
Realm	5	2500 KB	+	+	+	5	May 13, 2017
GreenDAO	4	56 KB	+	+	+	5	Apr 28, 2017
ActiveAndroid	4	17 KB	+	-	ActiveAndroid Rx	5	Dec 2, 2016
Sprinkles	3	28 KB	-	-	-	2	Apr 18, 2017
Room	5	30 KB	-	-	+	5	June, 2017
SQLite	3	_	-	-	+	5	_

Опишем більш детально параметри з таблиці 2:

- Зручність використання — суб'єктивний параметр, який будується на тому, чи є необхідність писати SQL-запити; наскільки просто реалізовані ті чи інші операції; скільки коду необхідно, щоб описати БД; чи потрібно писати свої DAO-класи та інше.
- Розмір бібліотеки перевірявся щодо розміру отриманого .apk, в порівнянні з .apk, отриманим без використання ORM.

- Під кешуванням розуміється збереження даних при записі / зчитуванні в окремому кеші, для того, щоб не виконувати зайвих запитів до БД в майбутньому.
- Відкладена ініціалізація - прийом в програмуванні, коли деяка ресурсномістка операція (створення об'єкта, обчислення значення) виконується безпосередньо перед тим, як буде використаний її результат. Таким чином, ініціалізація виконується «на вимогу», а не завчасно. Таким, наприклад, є завантаження пов'язаного ставленням один до багатьох об'єкта тільки при зверненні до нього, або навіть завантаження значень полів об'єкта при першому зверненні до самого поля.
- RxJava (RxJava2) на даний момент використовується багатьма Android-розробниками, тому її підтримка є актуальним вимогою до бібліотеки.
- Оцінка документації проводилася за такими критеріями:
    - наявність документації
    - розглянуті прості запити, операції створення простої моделі
    - розглянуті складні запити, операції створення комплексної моделі
    - присутні повні приклади використання
- Останнє оновлення бібліотеки на момент написання статті (19.05.2017, виключаючи Room).

Висновки

ORMLite. Не сильно рятує від boilerplate коду, проте одна з найшвидших бібліотек з представлених, присутнє вбудоване кешування і відкладена ініціалізація, якісна документація.

SugarORM. З переваг можна виділити зручність використання, якого не вистачає в ORMLite, але на цьому все.

Має окремий недолік, пов'язаний з підходом до реалізації — існують проблеми з одночасним використанням з Instant Run.

Freezer. Відносно молода бібліотека, мало поширена. Зручна, але будь-яких складних маніпуляцій з нею не зробиш. Крім цього, швидкість роботи дуже низька.

DBFlow. Одна з кращих за параметрами з представлених в цій статті. Одна з кращих за швидкістю операцій write / read, середня по швидкості операцій update / delete. За параметрами, наведеними в таблиці 2, є найкращою, нарівні з Realm.

Requery. Досить спірне рішення, здебільшого через проблеми з установкою та документацією (якщо використовувати без RXJava). Якщо розібратися, робота з нею стає зручною.

Підтримує різні БД, багатий функціонал.

Realm. Відносно швидка і зручна бібліотека, просто реалізуються будь-які зв'язки, що пов'язано з об'єктною орієнтованістю БД, при читанні виграє в швидкості навіть у "чистого" SQLite (навіть з вручну прописаним для нього кешем). Відмінна документація. Є, мабуть, одним з кращих варіантом зберігання даних на мобільному пристрої на даний момент, мінусом може бути тільки зростання розміру apk-файлу на 2.5 Мб.

Для всіх ORM, крім Realm, вбудоване кешування об'єктів було відключено, звідси такі показники при читанні.

GreenDao. Гнучко і зручно використовувати зв'язок один до багатьох, але присутня кодогенерація, в результаті якої класи наповнюються великою кількістю методів і коментарів. Крім цього, потрібно підключати Gradle плагін, який сильно збільшує час збирання.

ActiveAndroid. Не дуже швидке, але досить зручне рішення, що зарекомендувало себе з часом. Досить популярна, а тому можна знайти багато статей, рішень питань та іншого, що зводить появу проблем, що складно вирішити, до мінімуму.

Sprinkles. Кожен запис супроводжує непотрібна вибірка (SELECT * FROM %s WHERE %s LIMIT 1). Потрібно писати sql-запити. Показники швидкості роботи є гіршими з представлених.

Отже, немає причин використовувати її для своїх проектів.

Room. Цікаве рішення, представлене на Google I / O 2017 як оптимальне для роботи з БД в Android OS. Незважаючи на те, що необхідно використати наявні sql-запити, бібліотека виявилась досить зручною. За продуктивністю знаходиться в лідерах. Так як це рішення, представлене Google, воно швидко стане популярним, а, значить, з пошуком рішень задач, що попутно виникають при її використанні, проблем не буде.

SQLite. Багато коду, довше за часом написання коду, але за швидкістю і гнучкості все ж немає рівних.

Інше. Підводячи підсумки, "найшвидшими" ORM виявилися Realm, GreenDAO, ORMLite і Room, але, якщо продуктивність критична для проекту, "чистий" SQLite з вручну розробленим кешем все ще є кращим вибором.

За іншими параметрами, кращими є Realm, DBFlow і GreenDAO.

Таким чином, для невеликих проектів і проектів середньої складності краще використовувати DBFlow або GreenDAO, якщо розмір apk не важливий — Realm. Для великих проектів підходять Realm і Room. Якщо їх можливостей не вистачає, або ORM-підхід — це не для вас, використовуйте вбудоване API для SQLite.

Крім розглянутих в статті, спочатку також брали участь в розгляді такі ORM, як AndrORM, OHibernate, MemoryORM, ORMDroid, StromIO, які були відкинуті з тих чи інших причин.

Всі виміри проводилися на пристрої Samsung S7.

Код проекту можна знайти на Github за цим посиланням: https://github.com/AlexeyZatsepin/Android-ORM-benchmark/

Література:

1. About SQLite [Електронний ресурс] / SQLite — Режим доступу до ресурсу: https://www.sqlite.org/about.html.
2. OrmLite - Lightweight Object Relational Mapping (ORM) Java Package [Електронний ресурс] / OrmLite — Режим доступу до ресурсу: http://ormlite.com/.
3. DBFlow [Електронний ресурс] / Raizlabs — Режим доступу до ресурсу: https://github.com/Raizlabs/DBFlow.
4. greenDAO: Android ORM for your SQLite database [Електронний ресурс] / GreenRobot — Режим доступу до ресурсу: http://greenrobot.org/greendao/.
5. The Realm Mobile Platform [Електронний ресурс] / Realm — Режим доступу до ресурсу: https://realm.io/docs/.