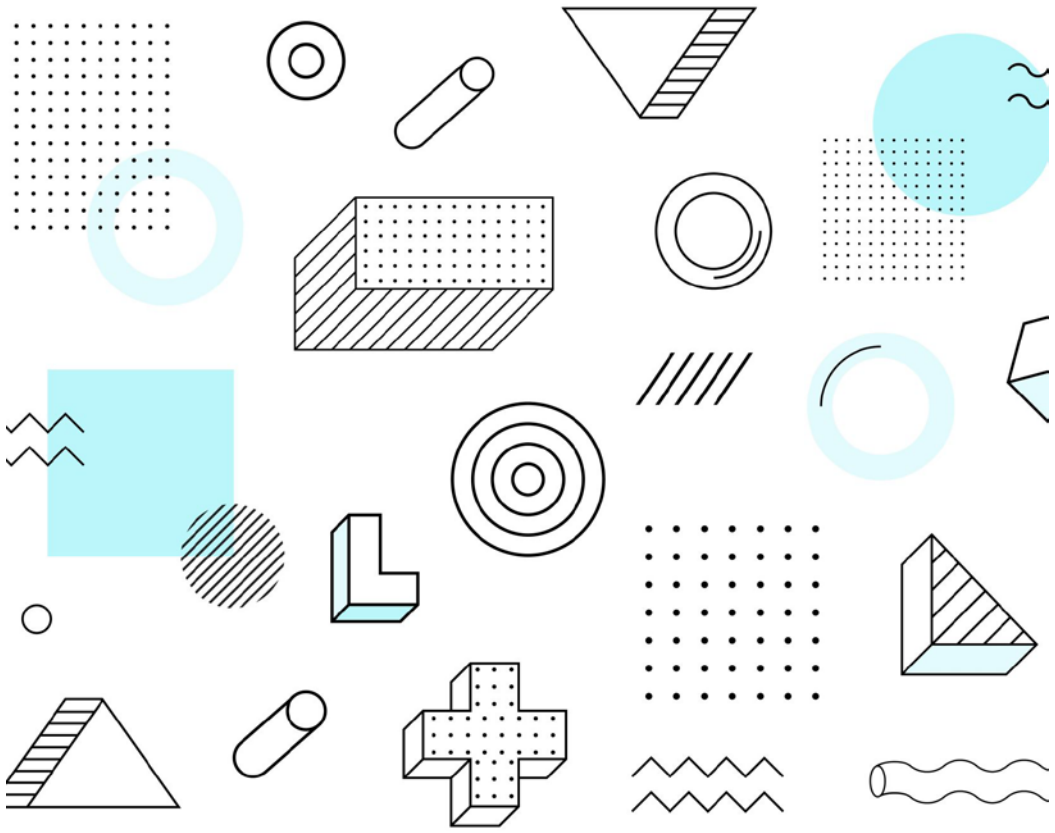


Використання LMDB для роботи з великими обсягами даних в iOS застосунках

Слайди до доповіді

Виконав:

Студент БП–2

”Інженерія програмного забезпечення”

Тарасенко М.С.

Науковий керівник:

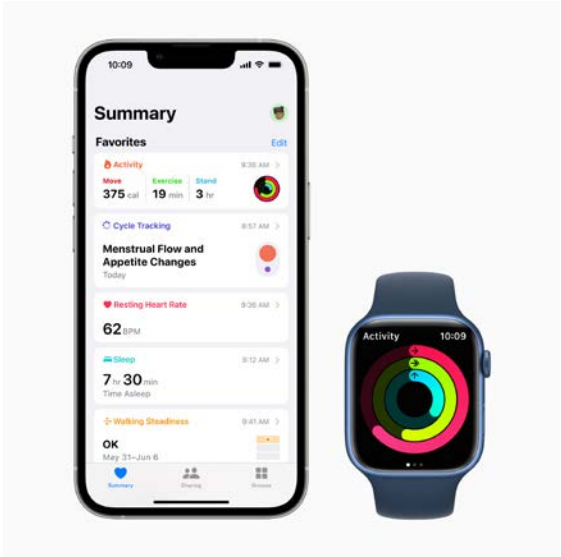
Старший викладач

Франків О.О.

Мета: дослідити можливість створення власної імплементації NSIncrementalStore для досягнення максимальної швидкості отримання інформації з бази даних.

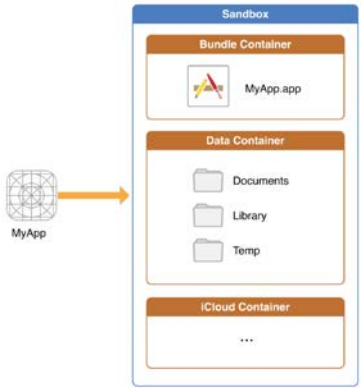
Результатом стала розробка проєкту CustomStore та iStoreBenchmark – для тестування і проведення порівнянь продуктивності порівняно з базовим рішенням

Поняття "Великі обсяги даних" та джерела їх виникнення в iOS застосунках



Огляд стандартних механізмів зберігання даних в iOS та їх обмеження

- Файлова система
- SQLite
- Realm
- Core Data



База даних LMDB:

- Зберігає дані у форматі ключ-значення
- ACID-транзакції
- Надшвидке читання
- Легке для використання Api
- Малий розмір бібліотеки (32 кб)

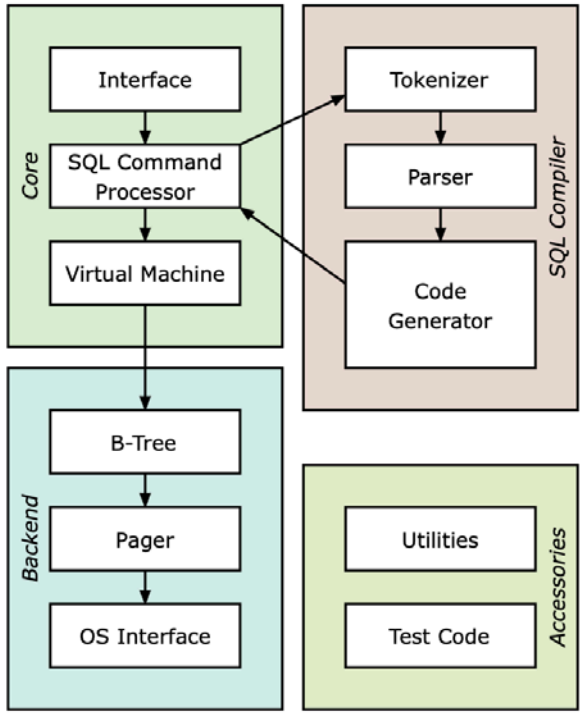
Архітектура та принципи роботи LMDB

Файли відображенні у пам’ять (memory-mapped-files)

Паралельне керування багатьма версіями (MVCC)

Append only B-tree

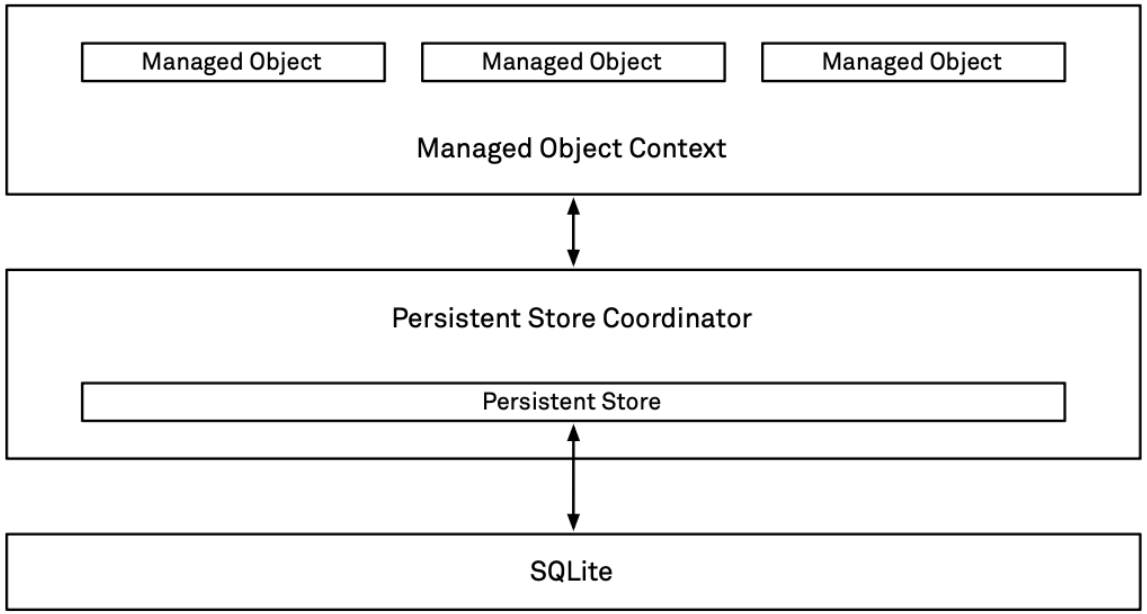
Принцип інтеграції SQLite та LMDB, потенційні переваги



	SQLite	SQLightning
Operation times in microseconds, lower is better		
Sync Seq Write	8175.371	6171.233
Sync Rand Write	8308.706	6231.249
Seq Write	25.587	31.778
Batch Seq Write	7.402	7.087
Rand Write	33.235	32.902
Batch Rand Write	18.847	13.754
Rand Read	22.645	7.685
Seq Read	7.557	1.551
Rev Seq Read	7.456	1.531

- Архітектура SQLite
- Принцип інтеграції
- Переваги від поєднання

Огляд ахітектури Core Data



- керовані об’єкти (NSManagedObject)
- контекст керованих об’єктів (NSManagedObjectContext)
- координатор постійного сховища (NSPersistentStoreCoordinator)
- постійне сховище (Persistent Store)
- NSIncrementalStore

```swift
import CoreData

class CustomIncrementalStore: NSIncrementalStore {

    override func loadMetadata() throws {
        code
    }

    override func execute(_ request: NSPersistentStoreRequest, with context: NSManagedObjectContext?)
        throws -> Any {
        code
    }

    override func newValuesForObject(with objectID: NSManagedObjectID, with context:
        NSManagedObjectContext) throws -> NSIncrementalStoreNode {
        code
    }

    override func obtainPermanentIDs(for array: [NSManagedObject]) throws -> [NSManagedObjectID] {
        code
    }
}

```

Механізм NSIncrementalStore: Інтерфейс для власного сховища

Написання власного NSIncrementalStore з використанням SQLightning

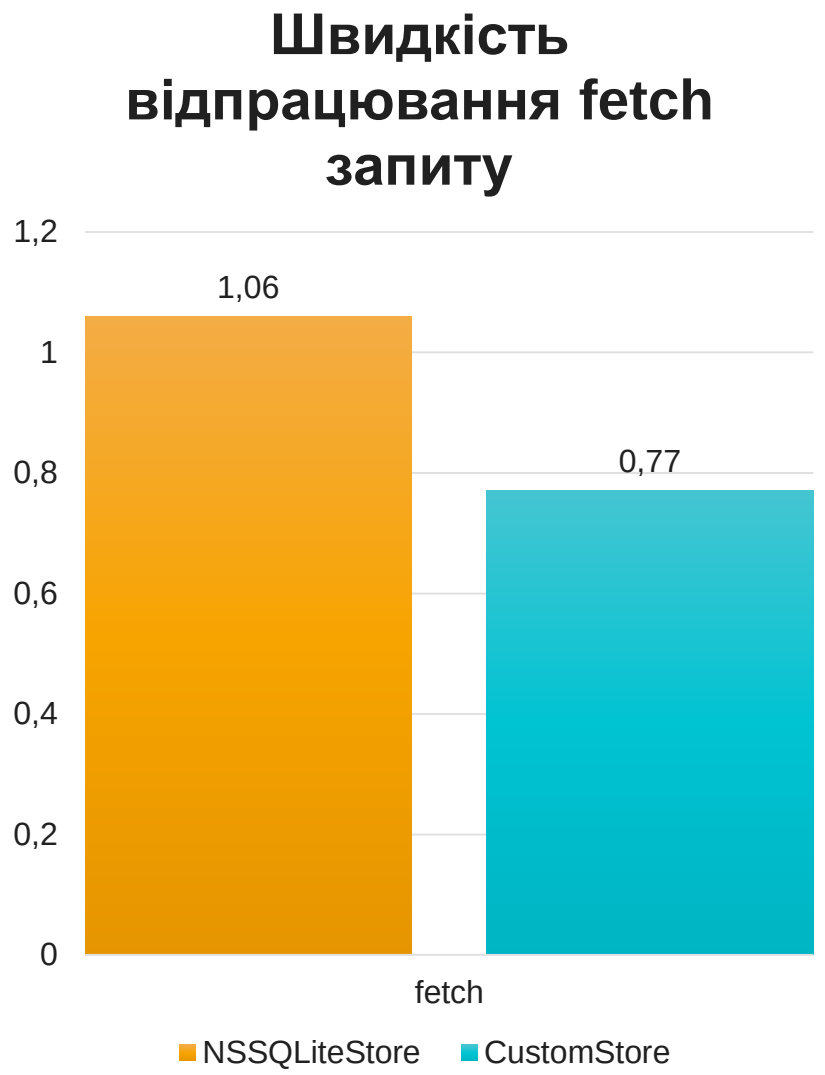
Проблеми при розробці

Загальна логіка постійного сховища

Деталі реалізації

Порівняння результатів власного рішення зі стандартним NSSQLiteStore та аналіз результатів

- Створення програми для тестування
- Методика проведення тестувань
- Порівняння результатів
- Аналіз проблем та недоліків



Висновки

- В ході даної роботи було досліджено механізми зберігання даних в iOS застосунках
- Детально розібрана архітектура та принципи роботи LMDB та Core Data
- Розроблено власний NSIncrementalStore
- Проведено порівняльний аналіз зі стандартним рішенням NSSQLiteStore

Дякую за увагу!