Хранение данных

Занятие №8

Дмитрий Тараев

Не забыть

МГТУ им. Н.Э. Баумана
ТЕХНОПАРК
Mail.Ru Group

Отметиться

Оставить отзыв

План

Изучим, как можно хранить данные и немного попишем код

Хранение данных

- Кеширование
- Оффлайн работа
- Быстродействие

Простое хранение

UserDefaults

- примитивы
- URL

Разделяется по доменам для кросс-доменной работы (их можно делить между extensions и вообще между приложениями)

Простое хранение

Файловая система

NSFileManager

- список файлов
- удаление
- перемещение
- копирование

Простое хранение

NSSearchPathForDirectoriesInDomains

- документы
- кеши

Всё должно лежать на своим месте, в документах только невосстановимые данные.

CoreData

Хранение деревьев объектов

- стор
- не является реляционной базой
- несколько бекендов
  - Sqlite
  - Xml
  - память

CoreData

- **NSManagedEntity** — хранимые объекты
- **NSManagedObject** — базовый класс для объектов
- **NSFetchRequest** — запрос на выборку объектов
- Все операции делаются через контекст
- Объекты нельзя передавать между потоками напрямую
  - Можно только через ид объекта

CoreData: основные понятия

- **NSManagedObjectModel** — описание структуры данных (загружается в память при старте)
- **NSPersistentStore** — конкретная разновидность хранилища (**SQLite**, Binary, In-Memory + свои)
- **NSPersistentStoreCoordinator**
  - отвечает за создание/извлечение существующих из persistent store;
  - передаёт их запросившему контексту.
- **NSManagedObjectContext**
  - при извлечении объектов из persistent store создаётся объектный граф;
  - все изменения выполняются в контексте;
  - если контекст сохраняют, то изменения сохраняются в persistent store.

CoreData

NSFetchedResultsController

- Выбирает данные для представления в виде UITableView
- Поддерживает секции
- Реагирует на изменения NSFetchRequest и вызывает делегат с описанием этих изменений

NSPredicate

Класс, предназначенный для задания каких-то условий фильтрации

Его можно создать

- из строки с условием
- блоком кода, который отдаёт Bool в зависимости от того, подходит объект или нет
- как логическую комбинацию других предикатов
- Подробнее см. ссылки

Realm

- Тоже объектное хранилище
- Использует свой собственный высокопроизводительный бекенд хранения
- Есть для ObjectiveC, Swift, Java, JS (если у вас swift + objc, то надо использовать objc версию)

Realm

- **Object** — базовый класс для моделей данных
- Связи моделируют отдельным классом **List**
- Условия в запросах делаются тоже через **NSPredicate**
- Объекты realm не должны передаваться между потоками
- Аналог **NSFetchedResultController** это нотификации

SQLite

- низкий уровень
- C-API
- Реляционная база
- есть проблемы с i18n

Для облегчения работы есть несколько библиотек

Ссылки про хранение данных

Ссылки из презентации

Apple CoreData Guide: https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/CoreData/index.html

Realm: https://realm.io

NSPredicate cheat sheet: https://news.realm.io/news/nspredicate-cheatsheet/

SQLite obj-c wrapper: https://github.com/ccgus/fmdb

SQLite swift wrapper: https://github.com/stephencelis/SQLite.swift

Другие базы

YapDatabase: https://github.com/yapstudios/YapDatabase

Pencil: https://github.com/naru-jpn/pencil