

Сервис Takeaway для запоминания информации

Takeaway: Service for Information Memorization

Исполнитель: Галимов Тимур Рафаэлевич, БПИ 132
Преподаватель: Мицюк Алексей Александрович, Преподаватель, Департамент
программной инженерии

План презентации

- Предметная область. Идея и функционал приложения.
- Реализация. Синхронизация данных.
- Демонстрация. Обзор реализованных и нереализованных задач.

Описание идеи



Flashcards











- 1) **Добавление информации и структуризация:** карточки добавляются в коллекции
- 2) **Запоминание:** нужно придумать, как разделить информацию для передней и задней стороны карточки











Решение



Flashcards

- 1) **Добавление информации и структуризация:** карточки заметки (короткие, 150 символов), к ним добавляются теги
- 2) **Запоминание:** нужно придумать, как разделить информацию для передней и задней стороны карточки 1) приложение присылает уведомления с заметками через определенные периоды времени (настраивается пользователем) 2) можно сделать “flashcards” из заметок 3) дополнительный функционал

											Take away
Запоминание с помощью флэш-карточек	+	+	+	+	+	+	+	-	+	-	+
Элементы тестов	+	-	+	-	-	-	+	+	+	-	-
Публичные коллекции	+	+	+	+	-	-	-	-	+	-	+
Пользователь оценивает знание заметки / карточки	-	+	-	-	-	-	-	-	+	-	+
Автоматизация / упрощение создания заметок	-	-	-	-	-	-	-	+	-	+	+
Теги, Нотификации	-	-	-	-	-	-	-	-	-	-	+

											Take away
Клиенты для других платформ (macOS, Chrome Extension)	+	+	+	-	-	-	-	+	+	+	+
API для создания заметок	-										+
Интеграция с онлайн платформами для обучения											+
Социальный функционал											+
Проверка знания карточки / заметки через период времени											+

Актуальность

- Подготовка к тестам, контрольным, экзаменам (студенты, школьники)
- Профессиональное развитие
- Расширение кругозора, эрудиция
- Преподаватели создают коллекции для студентов

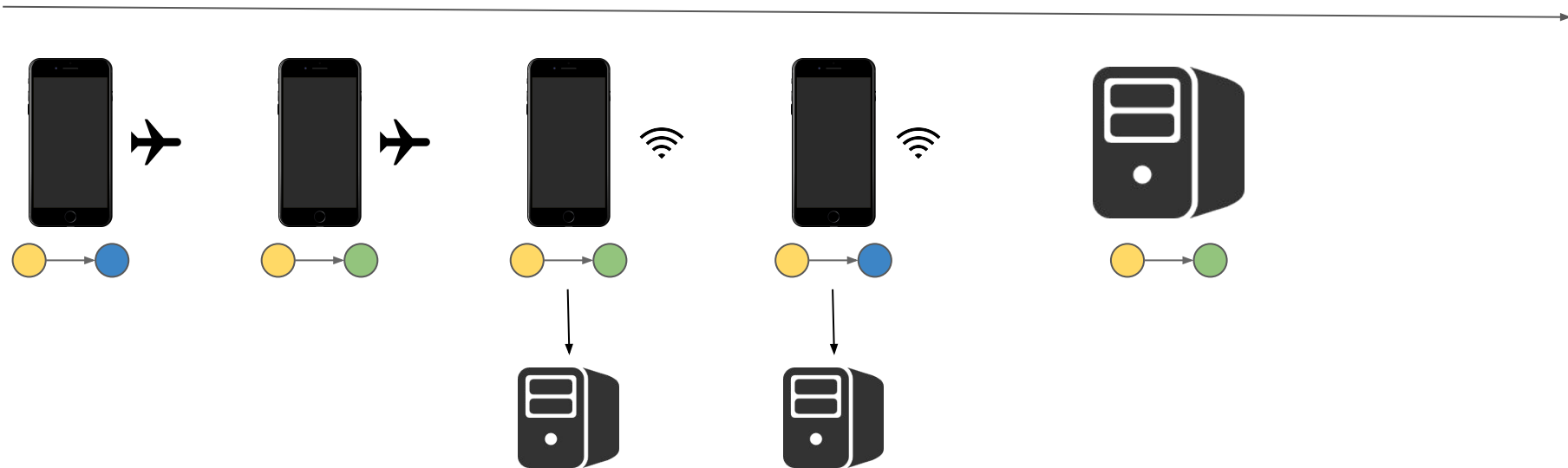
Цели и задачи

Цель: качественное, удобное в использовании мобильное приложение, содействующее в решении проблемы запоминания информации

Задачи:

- 1) Разработка модели данных и механизма автоматической синхронизации данных между сервером и клиентом
- 2) Проектирование и реализация удобного интерфейса мобильного приложения
- 3) Разработать и реализовать серверное API для синхронизации данных, деплой сервера
- 4) Тестирование (автоматизированное, а также ручное, интеграционное)

Синхронизация данных



Синхронизация данных

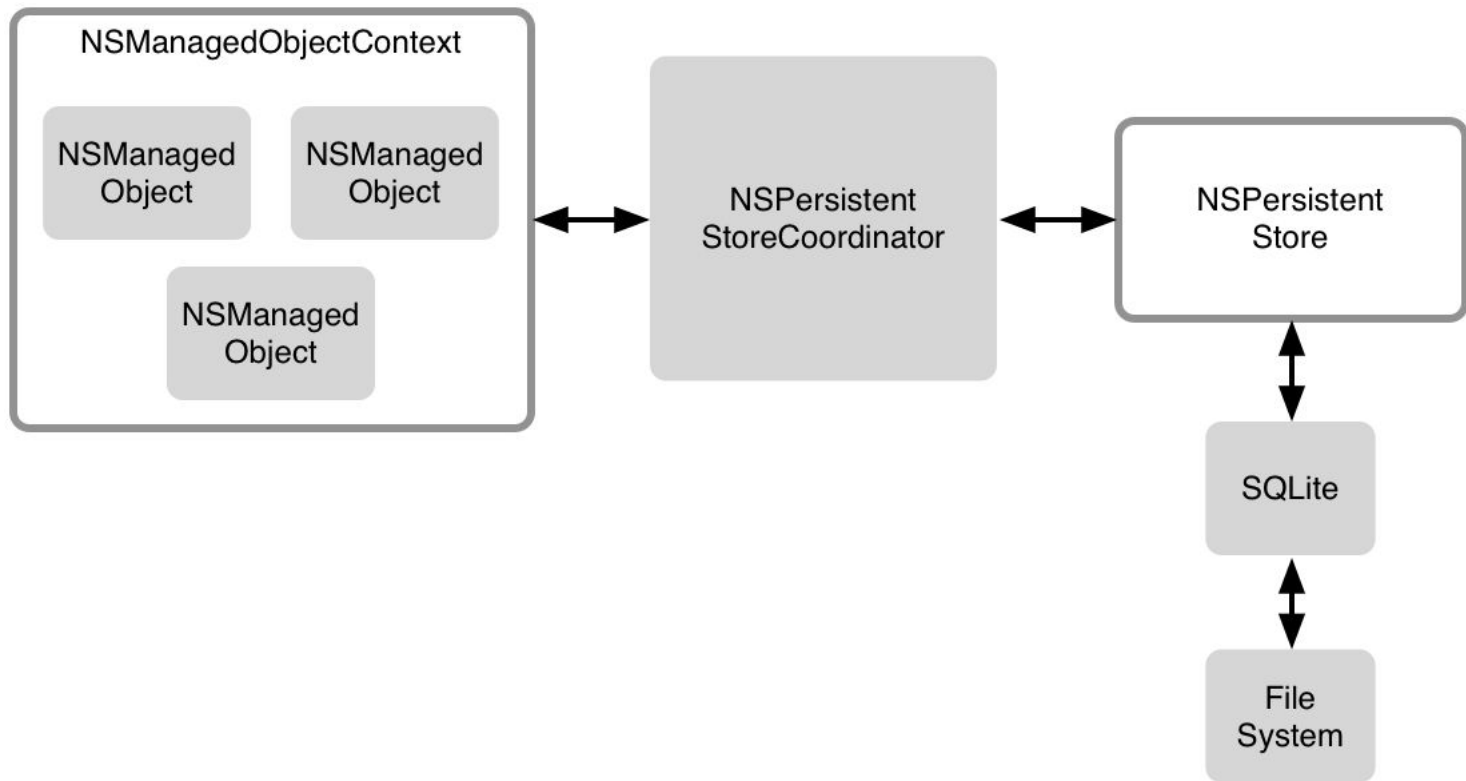


Change:

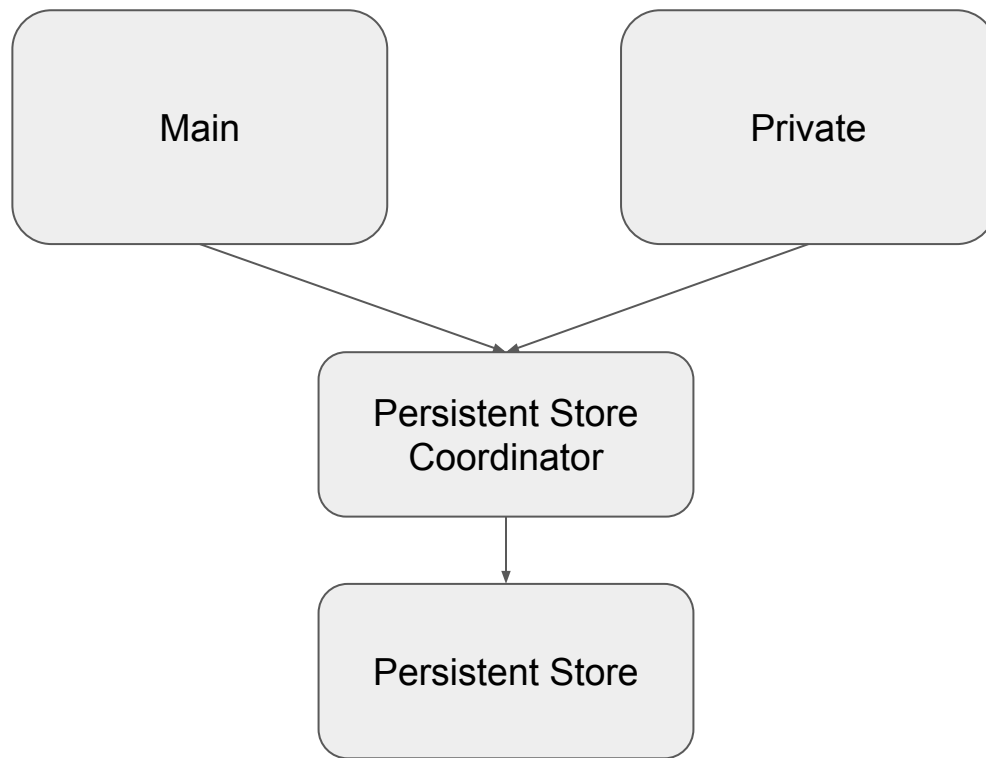
- value
- fieldName
- changedObjectID
- timestamp
- token*



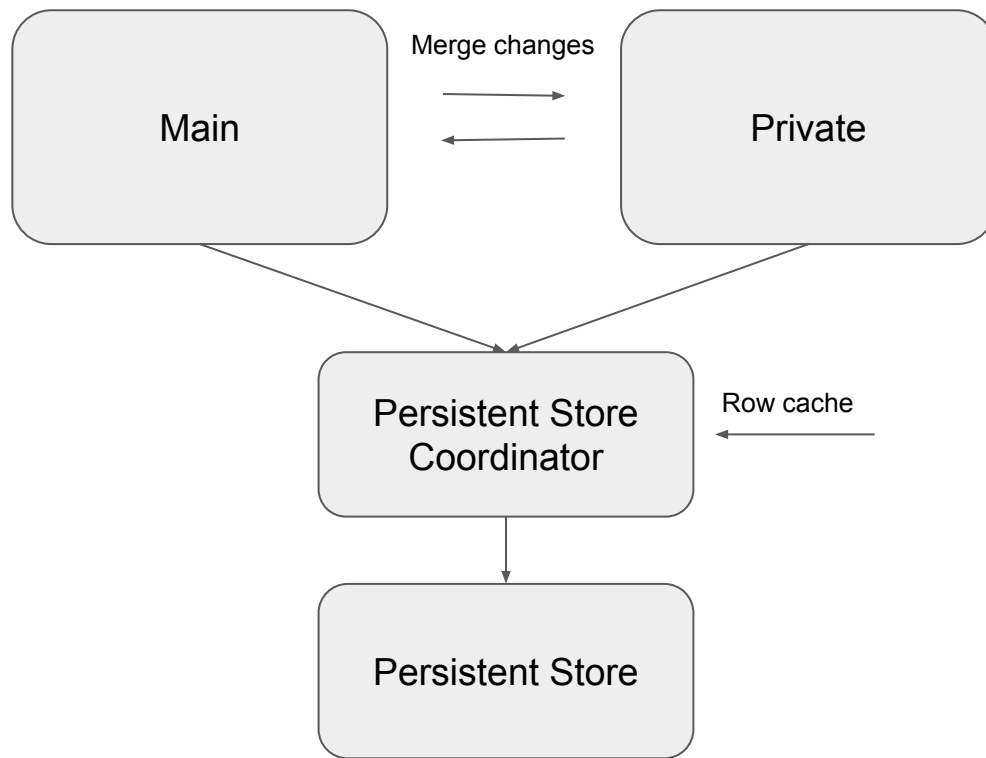
Core Data



Core Data Sync






Core Data Sync







Демонстрация

Технологии и инструменты

Сервер:

- Google App Engine 
- Язык программирования Python 
- IDE: PyCharm 

Клиент:

- iOS 
- Язык программирования Swift 3 
- XCode 
- Crashlytics 

- Git 
- Github 
- Trello 

Сделано

- Базовый функционал, синхронизация
- Серверное API (90%)
- Документация (20%)

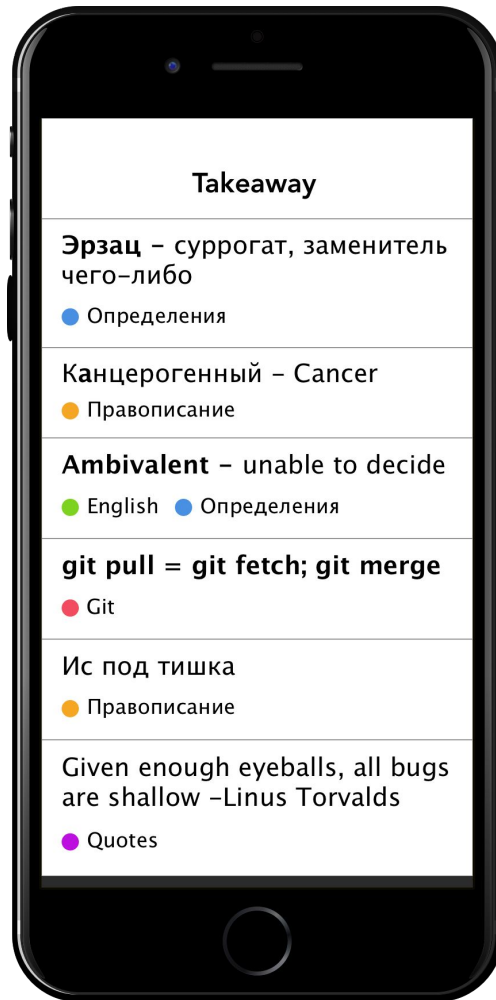
До защиты

- Доработать коллекции
- Доработать работу с карточками
- Нотификации
- Расширение для выделения текста
- Интерфейс (дизайн)
- Доработать документацию

ИСТОЧНИКИ

1. Florian Kugler and Daniel Eggert. Core Data. -Kugler, Eggert und Eidhof GbR, 2015. - 309 с.
2. Chris Eidhof, Ole Begemann, and Airspeed Velocity. Advanced Swift. -Airspeed Velocity, Chris Eidhof.
3. REST; [Online]: <https://ru.wikipedia.org/wiki/REST>. [Accessed: 12.11.16].
4. The Swift Programming Language (Swift 3.0.1); [Online]; https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift_Programming_Language/. [Accessed: 12.11.16].
5. Google App Engine Python Standard Documentation; [Online]: <https://cloud.google.com/appengine/docs/python/>. [Accessed: 12.11.16].
6. Core Data; [Online]: <https://developer.apple.com/reference/coredata>. [Accessed: 13.11.16].

Спасибо за внимание



Takeaway

Эрзац – суррогат, заменитель чего-либо

● Определения

Канцерогенный – Cancer

● Правописание

Ambivalent – unable to decide

● English ● Определения

git pull = git fetch; git merge

● Git

Ис под тишка

● Правописание

Given enough eyeballs, all bugs are shallow –Linus Torvalds

● Quotes

Flashcard

Эрзац

Заменитель чего-либо,
суррогат

Создание Flashcard из заметки

Эрзац



Эрзац - заменитель
чего-либо, суррогат

Э _____ - з _____
ч _____-л _____, с _____



Эрзац - заменитель
чего-либо, суррогат

```

public override func awakeFromInsert() {
    super.awakeFromInsert()
    primitiveDate = NSDate()
}
@NSManaged private var primitiveDate: NSDate
// ...

```

`awakeFromInsert()` is only invoked once in an object's lifetime, namely when the object is first created. After having called the superclass's implementation, we initialize the date. You might have noticed that we're using the primitive variant of the date property again. The reason for this is that we don't want this change to be tracked as a change in the managed object; we simply want to be the default state of the object.

Summary

In this chapter, we explored Core Data's default data types and possibilities for storing other custom data types. For custom types, you always need to convert your custom data to one of the supported basic types in order to persist the data. To achieve this, you can just let your type conform to `NSCoding` and use a transformable attribute, you can specify your own value transformer, or you can implement custom accessor methods.

Which approach you should take depends a lot on your use case. Don't make it more complicated than it has to be, and do some real profiling to verify any worries about performance you might have.

Takeaways

- Core Data supports a wide range of basic types out of the box. Choose the one that fits your use case without unnecessarily wasting space.
- When storing binary data, always enable the option to allow external storage, especially if the binary data is large.
- The easiest way to persist custom data types is to simply make them conform to `NSCoding`. Use this option if you don't have valid concerns about storage size or performance.
- If you need to implement a more efficient storage format, consider whether or not the transformation should happen lazily. If so,

- implement your custom accessor methods. Otherwise, you can provide a custom value transformer.
- When implementing custom accessors, always remember to wrap access to primitive properties in `willAccess.../didAccess...` or `willChange.../didChange...` calls.
 - Mark attributes as non-optional by default and only make exceptions if you have to. This applies especially to numeric attributes.
 - Next to the static default values you can specify in the model editor, you can set default values at runtime by overriding `awakeFromInsert()`.

Функционал

- Заметки, тэги, markdown текста
- Синхронизация данных
- Нотификации (push, local, настройка, отправка)
- Sharing (отправить пользователю напрямую, поделиться ссылкой)
- Поиск (Spotlight, локальный, публичный)
- Создание коллекций заметок, публичные коллекции, explore
- Статистика по созданным заметкам
- Оценка насколько хорошо запомнил
- Создание Flashcards из заметок
- Сохранение фильтров