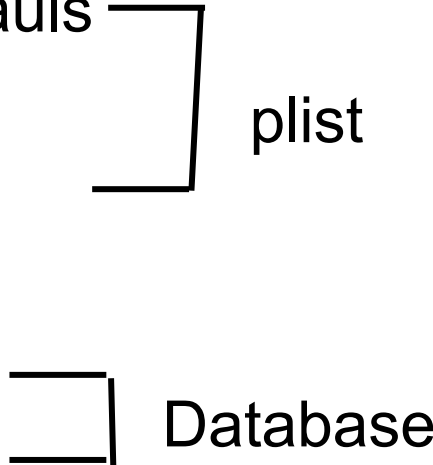


Tek Talk #2X

Lưu Trữ Dữ Liệu (Store Data)

Nguyễn Ngọc Lưu
xx/yy/2016

Nội Dung

1. Tổng Quan
 2. UserDefaults
 3. Keychain
 4. File Plist
 5. File
 6. Sqlite
 7. Core Data
- 
- plist
- Database

Qua mỗi kiểu lưu trữ, sẽ tìm hiểu về :

- Cách lưu trữ-bản chất
- Cách sử dụng(Coding)
- Hiệu năng (performance)
- Bảo mật(security)

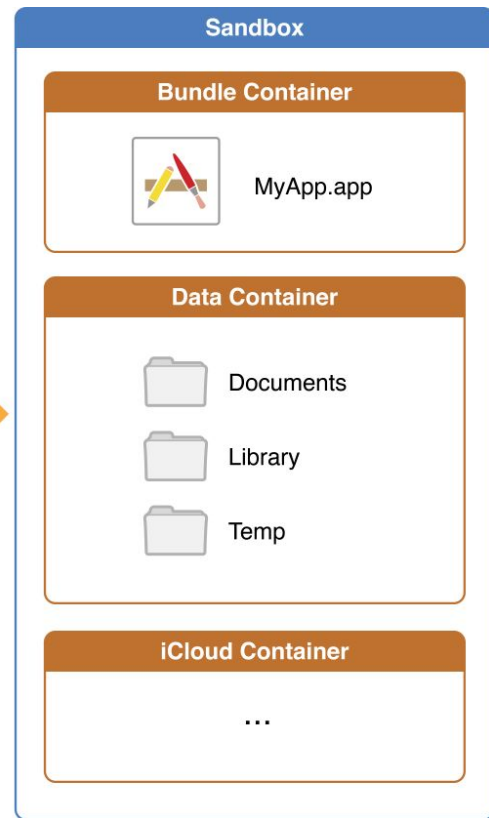
=> Nên sử dụng khi nào

1. Tổng Quan

- 99% ứng dụng có lưu trữ dữ liệu
- Dữ liệu : thông tin đã được số hóa
- Bộ nhớ trên mobile:
 - + Ram : tốc độ cực nhanh, dung lượng nhỏ, bị mất khi mở lại app
 - + Disk :
 - Internal Data Storage : NAND Flash(nhanh)
 - External Data Storage : SD card(android only)

1. Tổng Quan

- Cấu trúc thư mục 1 app iOS:
 - + Documents: dữ liệu liên quan tới user
 - + Library: dữ liệu không liên quan tới user, download, cache
 - + Temp: dữ liệu tạm
- Chú ý flag “Do not backup”



2. UserDefaults

- Lưu trữ kiểu : kiểu cơ bản, NSData, NSString, NSNumber, NSDate, NSArray, NSDictionary
- Bên dưới data lưu thông qua file Plist
- Cách dùng đơn giản(ưu điểm):

```
[[NSUserDefaults standardUserDefaults] setObject:valueToSave forKey:@"key"];  
NSString *savedValue = [[NSUserDefaults standardUserDefaults] valueForKey:@"key"];
```
- Tốc độ : nhanh => Ưu điểm
- Độ bảo mật không cao => Nhược điểm
- Nên sử dụng lưu dữ liệu setting ko quan trọng trong app.

3. Keychain

- Lưu password hoặc dữ liệu cần bảo mật
- Api bằng C, khó sử dụng => có thể sử dụng lib wrapper
- Dữ liệu lưu đã được mã hóa
- Không bị mất khi xóa app
- Có thể chia sẻ giữa các app cùng Access Group
- Cách sử dụng:<Demo>
- Thường sử dụng lưu password, token, uuid, credit cards

4. Plist

- Lưu/load Dictionary hoặc Array vào file plist
- Dictionary/Array được encode xml và lưu xuống file, decode khi load lên
- Tốc độ nhanh với tập dữ liệu không nhiều(< 100 bản ghi)
- dữ liệu ko được mã hóa
- Cách sử dụng:

Array:

```
[arr writeToFile:path atomically:YES];
```

```
NSArray *arr = [[NSArray alloc] initWithContentsOfFile:path];
```

Dictionary:

```
[dict writeToFile:path atomically:YES];
```

```
NSDictionary *dict = [[NSDictionary alloc] initWithContentsOfFile:filePath];
```

4. Plist - NSCoder

- Đơn giản để lưu object model
- Bản chất serialized object với xml và lưu vào file
- serialized qua `NSKeyedArchiver`, deserialized qua `NSKeyedUnarchiver`
- Tốc độ nhanh với tập dữ liệu ít, độ bảo mật không cao
- Cách sử dụng:
 - + conform to NSCoder: `class Book: NSObject, NSCoder`
 - + viết code ở: `-initWithCoder:` và `encodeWithCoder:`
 - + serialized qua `NSKeyedArchiver`
 - + deserialized qua `NSKeyedUnarchiver`
- Thường dùng lưu trữ object với số lượng không lớn (< 100)

5. File

- Lưu trữ và load trực tiếp file vào Disk
- Thường lưu file image, sound, text, zip,... Có thể tự tùy chỉnh cấu trúc file
- Cho phép lưu với dung lượng lớn
- Nên tổ chức thư mục, file thuận tiện và hợp lý
- NSFileManager quản lý hệ thống file
- Lưu data: **[data writeToFile:atomically:]**
[data writeToFile:options:error:]
- Load: **[NSData dataWithContentsOfFile:]**
[NSData dataWithContentsOfFile:options:error:]
- Tốc độ tùy thuộc vào kích thước file
- Lưu/đọc file dung lượng lớn nên chạy async
- Bảo mật ko cao => nên mã hóa data trước khi lưu với dữ liệu quan trọng

6. Sqlite

- Là hệ csdl dạng nhỏ, lưu dữ liệu dạng bảng.
- Hỗ trợ thao tác cơ bản(query, insert, update, delete) và transaction
- Thường được sử dụng trên thiết bị phần cứng yếu, ứng dụng nhỏ
- Có thư viện trên nhiều ngôn ngữ

Trên IOS :

- IOS đã có sẵn lib sqlite(libsqlite3.dylib)
- Thường để lưu dữ liệu nhiều(trăm, nghìn record)
- Api bằng C, khó sử dụng => dùng wrapper
- Nhẹ, nhỏ gọn, tốc độ truy cập phụ thuộc vào độ lớn + cấu trúc bảng
- Độ bảo mật ko cao => sqlcipher

6. Sqlite-Syntax

- CREATE TABLE: tạo bảng
- SELECT: truy vấn
- INSERT: thêm mới dữ liệu
- UPDATE: cập nhật dữ liệu
- DELETE: xóa dữ liệu
- BEGIN/COMMIT: transaction
- VD:
 - + **CREATE TABLE COMPANY(id INT PRIMARY KEY NOT NULL, name TEXT NOT NULL);**
 - + **SELECT id, name FROM COMPANY WHERE name LIKE "%tom%";**
 - + **INSERT INTO COMPANY (id, name) VALUES (1, 'Paul');**
 - + **UPDATE COMPANY SET name="Jimmy" where id=7;**
 - + **DELETE FROM COMPANY where id=9;**

[Link sqlite.org](http://link.sqlite.org)

6. Sqlite

- Cách sử dụng:
- mở db: `sqlite3_open([sqliteDb UTF8String], &_database)`
- thực hiện câu lệnh: `sqlite3_prepare_v2(_database, [query UTF8String], -1, &statement, nil)`
- fetch dữ liệu từ truy vấn:
`int uniqueId = sqlite3_column_int(statement, 0);`
`char *nameChars = (char *) sqlite3_column_text(statement, 1);`
- Kết thúc fetch dữ liệu: `sqlite3_finalize(statement);`
- Đóng kết nối: `sqlite3_close(_database);`

6. Sqlite-FMDB

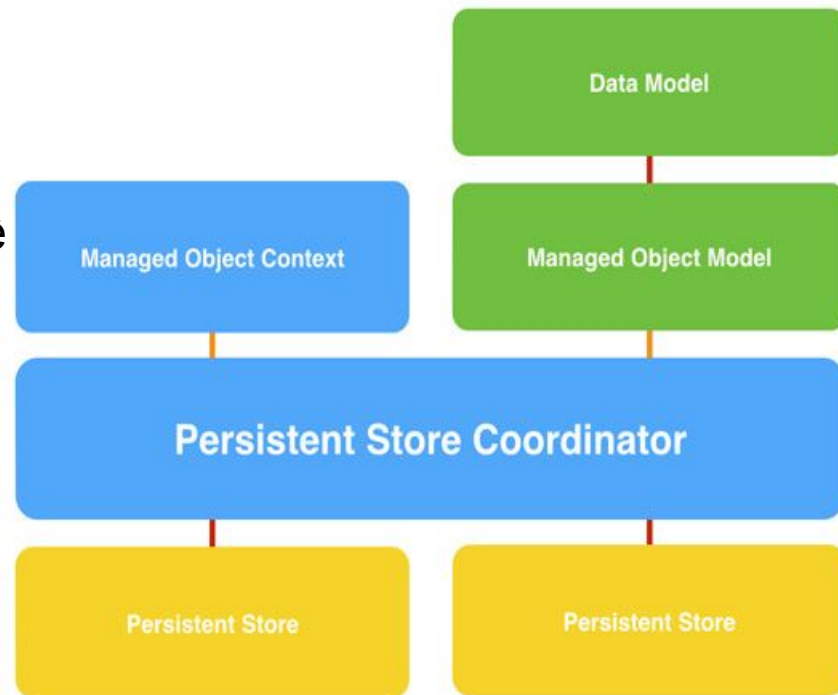
- Là một Objective-C wrapper cho sqlite
- Đơn giản và thuận tiện sử dụng sqlite
- Hỗ trợ thread Safe với FMDatabaseQueue
- Cách sử dụng:
 - + Tạo object db: `FMDatabase *db = [FMDatabase databaseWithPath:path];`
 - + mở db, tạo kết nối: `[db open]`
 - + truy vấn: `FMResultSet *s = [db executeQuery:@"SELECT * FROM myTable"];`
 - + đóng kết nối: `[db close];`

7. Core Data

- Là một framework do Apple cung cấp để quản lý(lưu trữ) model object trong app
- Bên dưới dữ liệu được lưu với sqlite(default), xml hoặc binary file
- Viết code ít hơn, có template khi tạo mới project: Use Core Data
- Core data ở tầng trên của sqlite, nhiều ưu điểm:
 - + Giao diện trực quan khi thiết kế model
 - + Api đơn giản để quản lý(truy vấn, sửa, xóa)
 - + Thuận tiện và tối ưu khi sử dụng với UITableView : sử dụng **NSFetchedResultsController**
 - + Có hỗ trợ liên kết giữa các model
 - + ...
- Sử dụng bộ nhớ + size db lớn hơn sqlite

7. Core Data Stack

- NSPersistentStoreCoordinator: lưu, load, cache data
- NSManagedObjectModel: giữ kiến trúc models. Thông tin, thuộc tính, mối liên hệ giữa các model
- NSManagedObjectContext: quản lý các object model đã được tạo ra



7. Core Data

- Cách sử dụng:
 - + Tạo project với template có sẵn của Xcode
 - + Tạo các model trong file .momd
 - + Sử dụng NSFetchRequest để query dữ liệu
 - + Tạo mới object: `Item* item = [NSEntityDescription insertNewObjectForEntityForName:name inManagedObjectContext:managedObjectContext];`
 - + Xóa object : `[managedObjectContext deleteObject:item]`
 - + Lưu dữ liệu đã thay đổi xuống db: `[managedObjectContext save:&error]`

Kết Thúc

Câu hỏi và chia sẻ

