Разработка приложений для iOS

Лекция 8

Работа с данными

Глеб Тарасов
gleb34@gmail.com

Вспомним прошлое занятие

Что нужно сделать, чтобы загрузить данные в фоновом потоке, а потом обновить интерфейс?

```
dispatch_async(dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0), ^{
    // background
    dispatch_async(dispatch_get_main_queue(), ^{
        // main
    });
});
```

Как проще всего подключать сторонние библиотеки к нашему проекту?

через CocoaPods

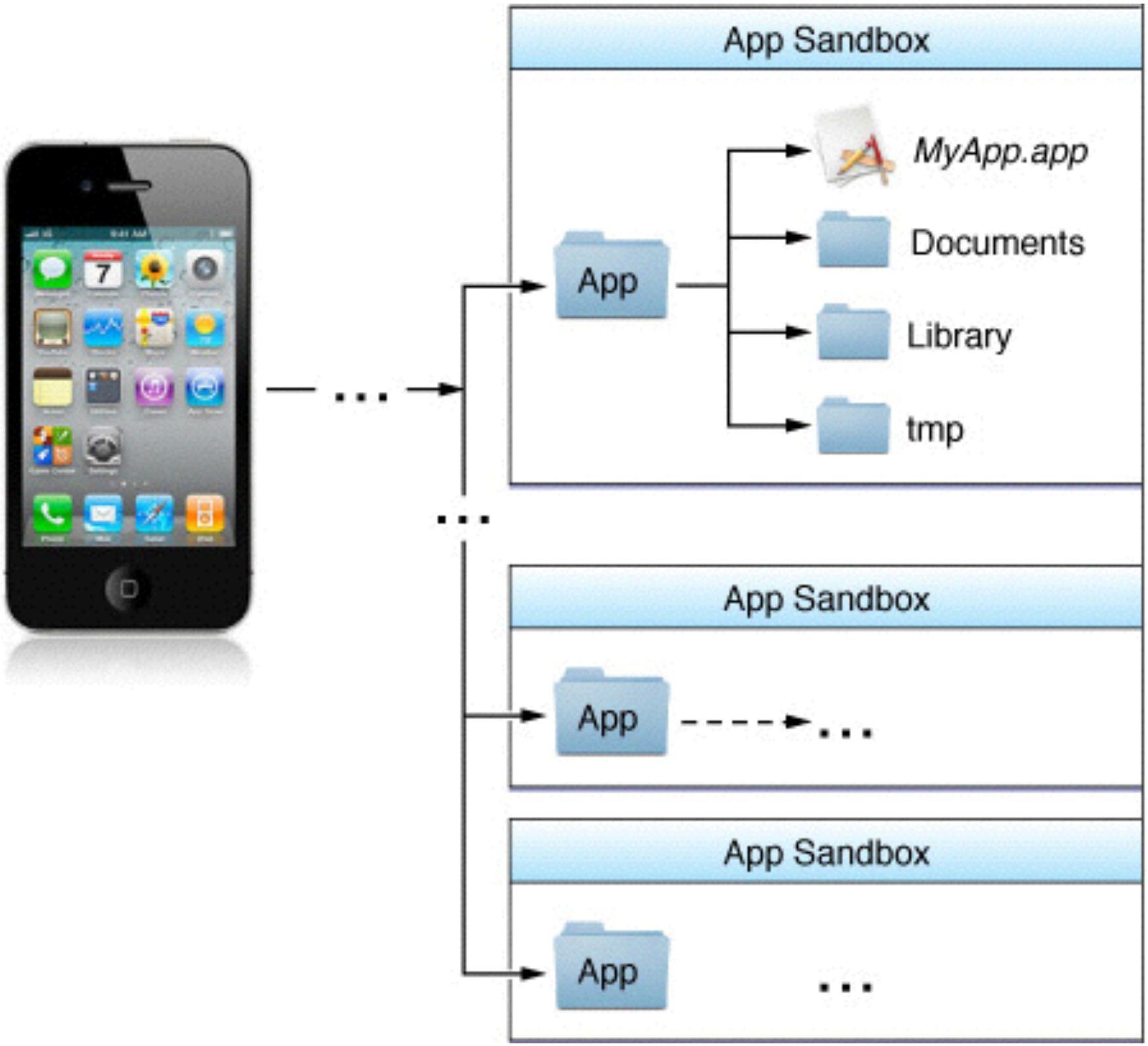
Как называется самая популярная библиотека для отправки GET и POST запросов?

AFNetworking

Какой самый простой способ загрузить текстовый файл по ссылке?

```
NSURL *url = [NSURL URLWithString:@"http://server.com/file.txt"];
NSString *str = [[NSString alloc] initWithContentsOfURL:url
                                               encoding:NSUTF8StringEncoding
                                                  error:nil];
```

Работа с файлами



https://developer.apple.com/library/ios/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html

<Application_Home>/AppName.app

readonly
Бандл приложения

<Application_Home>/Documents/

read/write

Критичные данные, которые мы не можем перегенерировать программно, чаще всего те, которые создал сам пользователь

<Application_Home>/tmp/

read/write

Временные файлы, которые очищаются после перезапуска программы

<Application_Home>/Library/

read/write

Важные файлы, которые нужны программе, но создал не пользователь.

<Application_Home>/Library/Application Support

read/write

лучше хранить все тут, а не в Library

<Application_Home>/Library/Caches

read/write

храним тут данные, которые можно потом перегенерировать или перезагрузить

com.apple.MobileBackup

Атрибут файла, который предотвращает backup файла в iCloud и iTunes

```
- (BOOL)addSkipBackupAttributeToItemAtURL:(NSURL *)URL
{
    assert([[NSFileManager defaultManager] fileExistsAtPath: [URL path]]);

    NSError *error = nil;
    BOOL success = [URL setResourceValue: [NSNumber numberWithBool: YES]
                                  forKey: NSURLIsExcludedFromBackupKey error: &error];
    if(!success){
        NSLog(@"Error excluding %@ from backup %@", [URL lastPathComponent], error);
    }
    return success;
}
```

https://developer.apple.com/library/ios/qa/qa1719/_index.html

Получения пути к папке

```
NSArray *paths = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory,
                                                     NSUserDomainMask,
                                                     YES);
NSString *path = paths[0];
NSLog(@"%@", path);
```

NSDocumentDirectory

NSCachesDirectory

NSLibraryDirectory

NSApplicationSupportDirectory

Путь к папке tmp

```
NSString *path = NSTemporaryDirectory();
NSLog(@"%@", path);
```

Работа с путями

```
NSString *path = NSTemporaryDirectory();
NSString *folderPath = [path stringByAppendingPathComponent:@"folder"];
NSString *filePath = [folderPath stringByAppendingPathComponent:@"file.txt"];

NSString *folderPath2 = [filePath stringByDeletingLastPathComponent];
NSString *path2 = [folderPath2 stringByDeletingLastPathComponent];
```

```
NSURL *url = [NSURL fileURLWithPath:path];
NSLog(@"%@", url.path);
```

NSFileManager

```
NSFileManager *m = [NSFileManager defaultManager];

[m removeItemAtPath:path error:nil];
[m removeItemAtURL:url error:nil];

[m copyItemAtPath:pathFrom toPath:pathTo error:nil];
[m copyItemAtURL:urlFrom toURL:urlTo error:nil];


[m moveItemAtPath:pathFrom toPath:pathTo error:nil];
[m moveItemAtURL:urlFrom toURL:urlTo error:nil];


    [m createDirectoryAtPath:path
 withIntermediateDirectories:YES
                  attributes:nil
                       error:nil];
```

Сохранение в файл

```
NSString *path = ...
NSString *filePath = [path stringByAppendingPathComponent:@"file.txt"];

NSString *str = @"test";
    [str writeToFile:filePath
          atomically:YES
            encoding:NSUTF8StringEncoding
               error:nil];
```

```
NSData *data = ...
[data writeToFile:path atomically:YES];
```

Чтение из файла

```
NSString *str = [[NSString alloc] initWithContentsOfFile:path
                                                encoding:NSUTF8StringEncoding
                                                   error:nil];
```

```
NSData *data = [[NSData alloc] initWithContentsOfFile:path];
```

NSUserDefaults

```
NSUserDefaults *d = [NSUserDefaults standardUserDefaults];

[d setObject:@"qwerty" forKey:@"key"];
[d setInteger:123 forKey:@"key2"];
[d setObject:@[ @"123", @(12), @"321" ] forKey:@"key3"];

[d synchronize];



NSUserDefaults *d = [NSUserDefaults standardUserDefaults];

NSString *str = [d objectForKey:@"key"];
NSInteger num = [d integerForKey:@"key2"];
NSArray *arr = [d objectForKey:@"key3"];
```

NSCoding

```
@interface User : NSObject<NSCoding>

@property (strong, nonatomic) NSString *firstName;
@property (strong, nonatomic) NSString *lastName;
@property (nonatomic) NSInteger age;

@end
```

```objc
@implementation User

- (void)encodeWithCoder:(NSCoder *)aCoder
{
    [aCoder encodeObject:self.firstName forKey:@"firstName"];
    [aCoder encodeObject:self.lastName forKey:@"lastName"];
    [aCoder encodeInteger:self.age forKey:@"age"];
}

- (id)initWithCoder:(NSCoder *)aDecoder
{
    self = [super init];
    if (self)
    {
        self.firstName = [aDecoder decodeObjectForKey:@"firstName"];
        self.lastName = [aDecoder decodeObjectForKey:@"lastName"];
        self.age = [aDecoder decodeIntegerForKey:@"age"];
    }
    return self;
}

@end
```

Сохранение произвольных объектов

```
NSUserDefaults *d = [NSUserDefaults standardUserDefaults];

User *u = ...
NSData *data1 = [NSKeyedArchiver archivedDataWithRootObject:u];

NSArray *arr = @[ user1, user2, user3 ];
NSData *data2 = [NSKeyedArchiver archivedDataWithRootObject:arr];

[d setObject:data1 forKey:@"data1"];
[d setObject:data2 forKey:@"data2"];
```

Считывание произвольных объектов

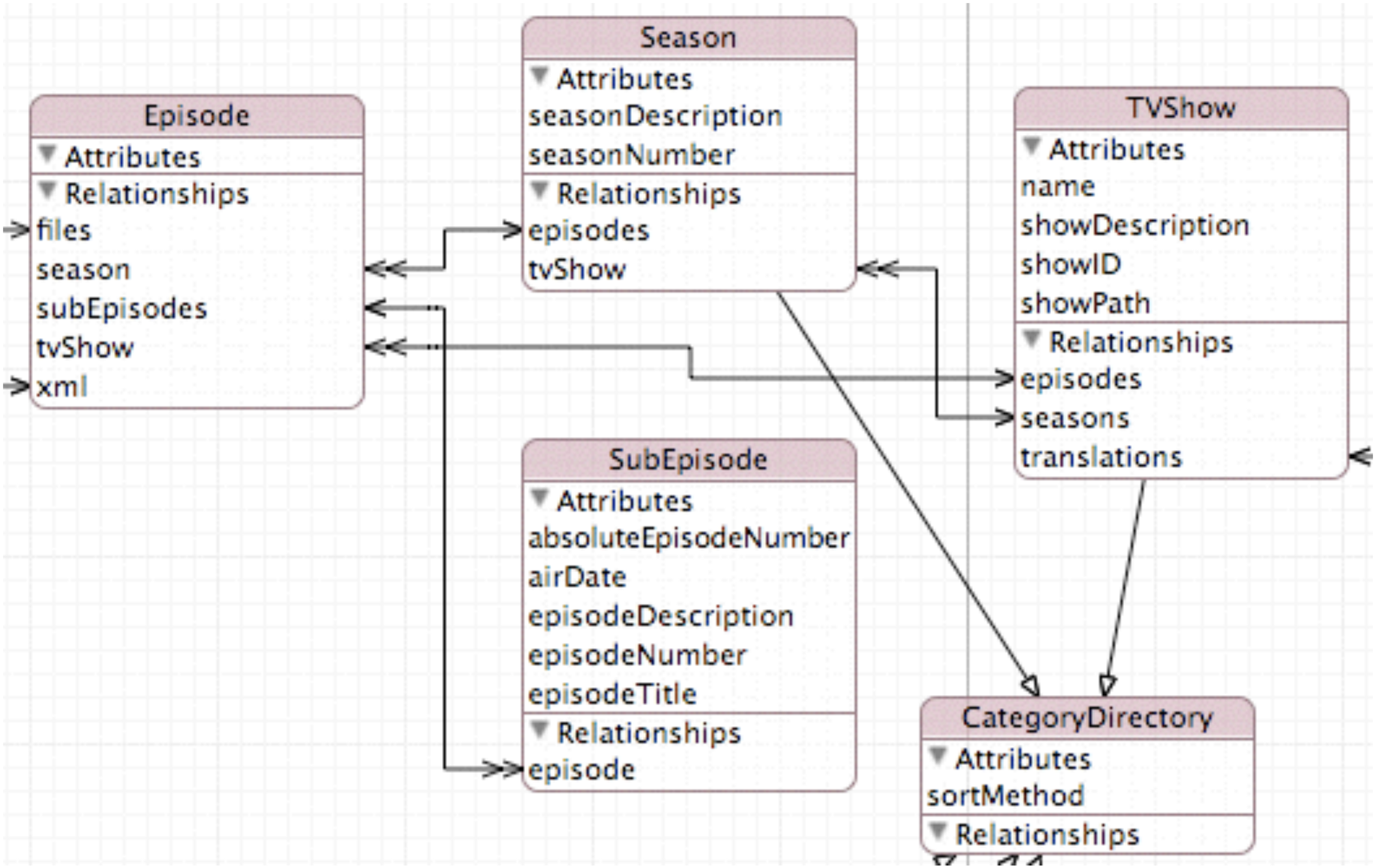
```
NSUserDefaults *d = [NSUserDefaults standardUserDefaults];

NSData *data1 = [d objectForKey:@"data1"];
User *u = [NSKeyedUnarchiver unarchiveObjectWithData:data1];

NSData *data2 = [d objectForKey:@"data2"];
NSArray *arr = [NSKeyedUnarchiver unarchiveObjectWithData:data2];
```

sqlite

Core Data


Episode
Attributes
Relationships
files
season
subEpisodes
tvShow
xml
Season
Attributes
seasonDescription
seasonNumber
Relationships
episodes
tvShow
TVShow
Attributes
name
showDescription
showID
showPath
Relationships
episodes
seasons
translations
SubEpisode
Attributes
absoluteEpisodeNumber
airDate
episodeDescription
episodeNumber
episodeTitle
Relationships
episode
CategoryDirectory
Attributes
sortMethod
Relationships

ENTITIES

- E City
- E Comment
- E Country
- E Shop
- E User

FETCH REQUESTS

CONFIGURATIONS

- C Default

Attributes

Attribute ▲	Type
S name	String

+ −

Relationships

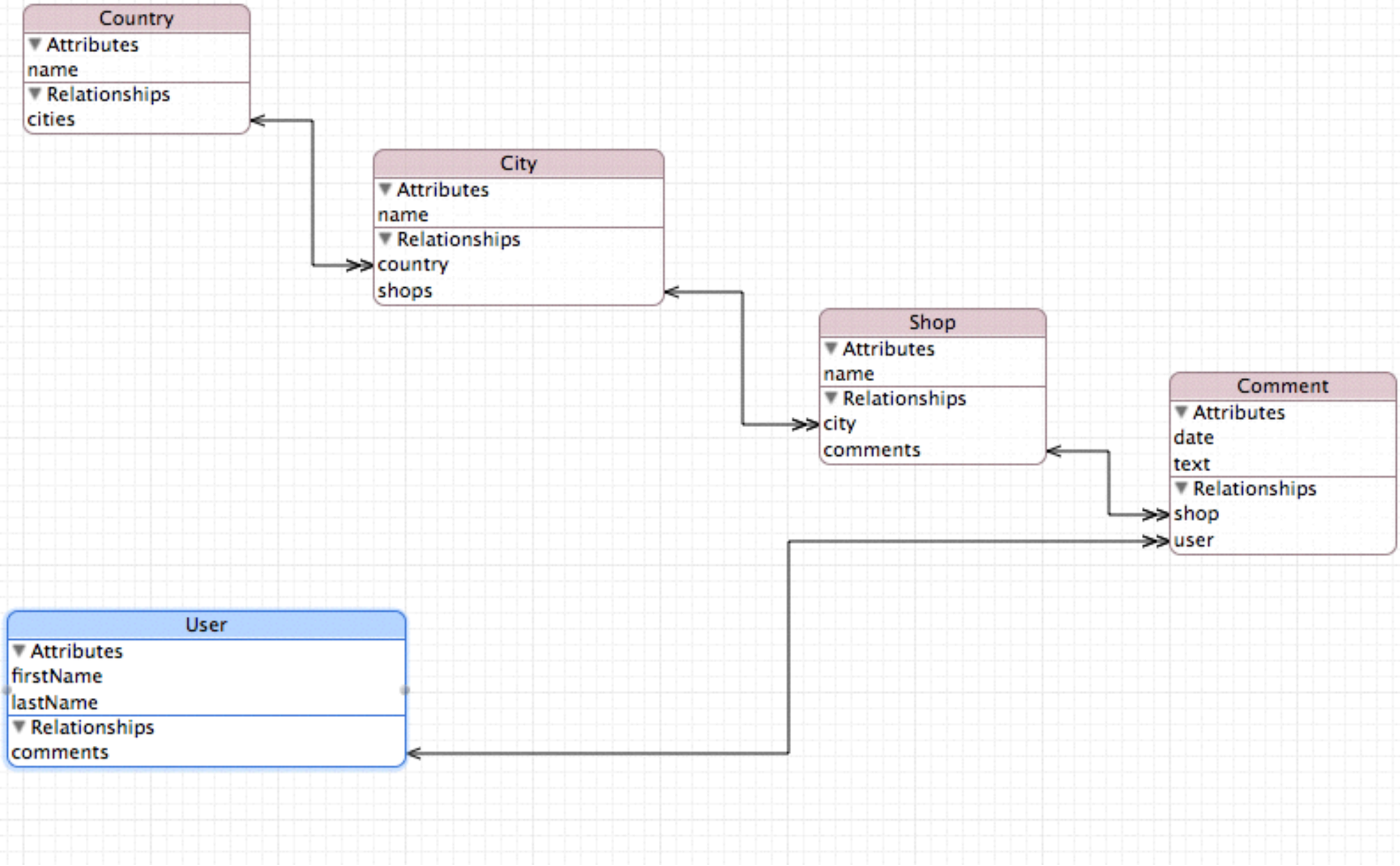
Relationship ▲	Destination	Inverse
O country	Country	cities
M shops	Shop	city

+ −

Fetched Properties

Fetched Property ▲	Predicate

+ −


Country
Attributes
name
Relationships
cities
City
Attributes
name
Relationships
country
shops
Shop
Attributes
name
Relationships
city
comments
Comment
Attributes
date
text
Relationships
shop
user
User
Attributes
firstName
lastName
Relationships
comments

NSManagedObjectModel

(аналог схемы базы данных)

```
- (NSManagedObjectModel *)createManagedObjectModel
{
    NSURL *modelURL = [[NSBundle mainBundle] URLForResource:@"Test" withExtension:@"momd"];
    NSManagedObjectModel *m = [[NSManagedObjectModel alloc] initWithContentsOfURL:modelURL];
    return m;
}
```

NSPersistentStoreCoordinator

(информация о типе БД и месте хранения данных)

```
- (NSPersistentStoreCoordinator *)loadPersistentStoreCoordinator
{
    NSString *docs = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory,
                                                         NSUserDomainMask,
                                                         YES)[0];
    NSString *filePath = [docs stringByAppendingPathComponent:@"db.sqlite"];
    NSURL *storeURL = [NSURL fileURLWithPath:filePath];

    NSDictionary *options = @{ NSMigratePersistentStoresAutomaticallyOption : @YES };

    NSPersistentStoreCoordinator *p = [[NSPersistentStoreCoordinator alloc]
                                       initWithManagedObjectModel:[self managedObjectModel]];

    if (![p addPersistentStoreWithType:NSSQLiteStoreType
                         configuration:nil
                                   URL:storeURL
                               options:options
                                 error:nil])
    {
        NSLog(@"error");
    }

    return p;
}
```

NSManagedObjectContext

(аналог соединения с базой данных)

```
- (NSManagedObjectContext *)loadManagedObjectContext
{

    NSPersistentStoreCoordinator *coordinator = [self persistentStoreCoordinator];
    NSManagedObjectContext *c = [[NSManagedObjectContext alloc] init];
    [c setPersistentStoreCoordinator:coordinator];

    return c;
}
```

Вставка объектов

```
NSManagedObjectContext *m = [self managedObjectContext];
NSManagedObject *city = [NSEntityDescription insertNewObjectForEntityForName:@"City"
                                                       inManagedObjectContext:m];
[city setValue:@"Москва" forKey:@"name"];
[m save:nil];
```

Загрузка объектов

```
NSFetchRequest *request = [[NSFetchRequest alloc]
                           initWithEntityName:@"User"];

NSSortDescriptor *sort = [NSSortDescriptor sortDescriptorWithKey:@"name"
                                                       ascending:YES];

NSPredicate *predicate = [NSPredicate predicateWithFormat:@"age > 12"];

request.predicate = predicate;
request.sortDescriptors = @[ sort ];

NSArray *users = [[self managedObjectContext] executeFetchRequest:request
                                                            error:nil];

for (NSManagedObject *user in users)
{
    NSLog(@"%@", [user valueForKey:@"name"]);
}
```

NSPredicate

http://nshipster.com/nspredicate/

Базовые сравнения

- =, == (age == 13)
- >=, => (age >= 13)
- <=, =< (age <= 13)
- > (age > 13)
- < (age < 13)
- !=, <> (age != 13)
- BETWEEN (age BETWEEN { 0 , 33 })

Базовые булевы операции

- AND, && (age > 13 && age < 27)
- OR, || (age > 13 || name != nil)
- NOT, ! (!(age > 13))

Строковые сравнения

[cd] - case insensitive и diacritic insensitive

- `BEGINSWITH (name BEGINSWITH[c] 'алекс')`
- `CONTAINS (name CONTAINS[cd] 'алекс')`
- `ENDSWITH (name ENDSWITH 'андр')`
- `LIKE: (name LIKE 'Але*нд?')`
- `MATCHES (name MATCHES 'Александр+')`

Функции агрегирования

- `ANY, SOME (ANY children.age < 18)`
- `ALL (ALL children.age < 18)`
- `NONE (NONE children.age < 18)`
- `IN: (name IN { 'Ben', 'Melissa', 'Nick' })`

Удаление объектов

```
for (NSManagedObject *user in users)
{
    [[user managedObjectContext] deleteObject:user];
}
```

Многопоточность в Core Data

- каждый контекст привязан к одному потоку
- каждая сущность привязана к одному контексту
- удобная многопоточная работа: библиотека MagicalRecord

Демонстрация

- создание модели данных
- создание собственных классов для сущностей
- миграция версий

Домашнее задание

- добавить работу с файлами и базой данных

Всё!

Глеб Тарасов
gleb34@gmail.com
twitter.com/pilot34