

个人资料



chaoyang805

+

关注

✉

发私信

恒

访问：25210次

积分：626

等级：BLOG > 3

排名：千里之外

原创：34篇

转载：10篇

译文：2篇

评论：4条

文章搜索

Q

文章分类

android开发

(23)

Java

(5)

《Android群英传》读书笔记

(12)

Swift

(7)

iOS

(16)

ObjectiveC

(10)

Git

(1)

CoreData

(6)

文章存档

2017年01月

(1)

2016年12月

(5)

2016年09月

(2)

2016年08月

(4)

2016年06月

(2)

展开

⌵

阅读排行

Android适配华为手机虚拟按...

(4191)

💡

赠书 | AI专栏 (AI圣经! 《深度学习》中文版)

机器学习&数据挖掘 系统实训

【获奖公布】征文 | 你会为 AI 转型么?

原

CoreData 从入门到精通 （一） 数据模型 + CoreData 栈的创建

标签：

iOS

CoreData

2016-12-03 18:09

👁

1444人阅读

💬

评论(0)

☆



微信关注CSDN
获得无限技术资源

✎

快速回复

☆

我要收藏

⬆

返回顶部

！

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录(?)	[+]
目录(?)	[+]

概述

CoreData 是 Cocoa 平台上用来管理模型层数据和数据持久化的一个框架，说简单点，就是一个**数据库**存储框架。CoreData 里相关的概念比较多，而且初始化也非常繁琐，所以对初学者的学习还是有一些困难的。这篇文章将从头到尾详细地讲一遍 CoreData 的使用方法，从 CoreData 的初始化到简单的增删改查，再到批量处理，数据模型的版本更新以及和 TableView 的结合等，真正让你能彻底了解 CoreData。

一、CoreData 数据模型的创建

想要使用 CoreData ，第一部是是创建数据模型，它描述了数据的结构和关联关系等。可以理解为**数据库**中的表结构。在 Xcode 创建工程时，提供了创建 CoreData 的模板，只需要我们在创建时，勾选 CoreData 选项，Xcode 就会自动创建出数据模型文件：

Swift 3.0 中的新变化

UIButton 的 backgroundImage ...

CoreData 从入门到精通 （一...

使用ActionMode实现ListView...

...

小牛电动N1

创新智能电动踏板车

6月15日，震撼登场

电动车小牛

卡瓦依钢琴

...

* 深入剖析基于并发AQS的重入锁(ReentrantLock)及其Condition实现原理

* Android版本的"Wannacry"文件加密病毒样本分析(附带锁机)

* 工作与生活真的可以平衡吗?

* 《Real-Time Rendering 3rd》提炼总结——高级着色：BRDF及相关技术

* 《三体》读后思考-泰勒展开/维度打击/黑暗森林

最新评论

CoreData 从入门到精通（四）并发操作

qq_33059955 : 你好，你的文章写的很详细，但是我在学到并发操作的时候遇到了点问题:这个方法并没有将分线程操作合并到主...

使用ActionMode实现ListView的多选功能

Noob_U_Are : ViewHolder 是怎么定义的? customt_view.xml是指R.id.custon_vi...

使用ActionMode实现ListView的多选功能

Noob_U_Are : ViewHolder 是怎么定义的? customt_view.xml是指R.id.custon_vi...

使用ActionMode实现ListView的多选功能

Noob_U_Are : ViewHolder 是怎么定义的? customt_view.xml是指R.id.custon_vi...

Choose options for your new project:

Product Name: CoreDataDemo

Team: None

Organization Name: chaoyang805

Organization Identifier: me.chaoyang805

Bundle Identifier: me.chaoyang805.CoreDataDemo

Language: Objective-C

Devices: iPhone

☒ Use Core Data

☐ Include Unit Tests

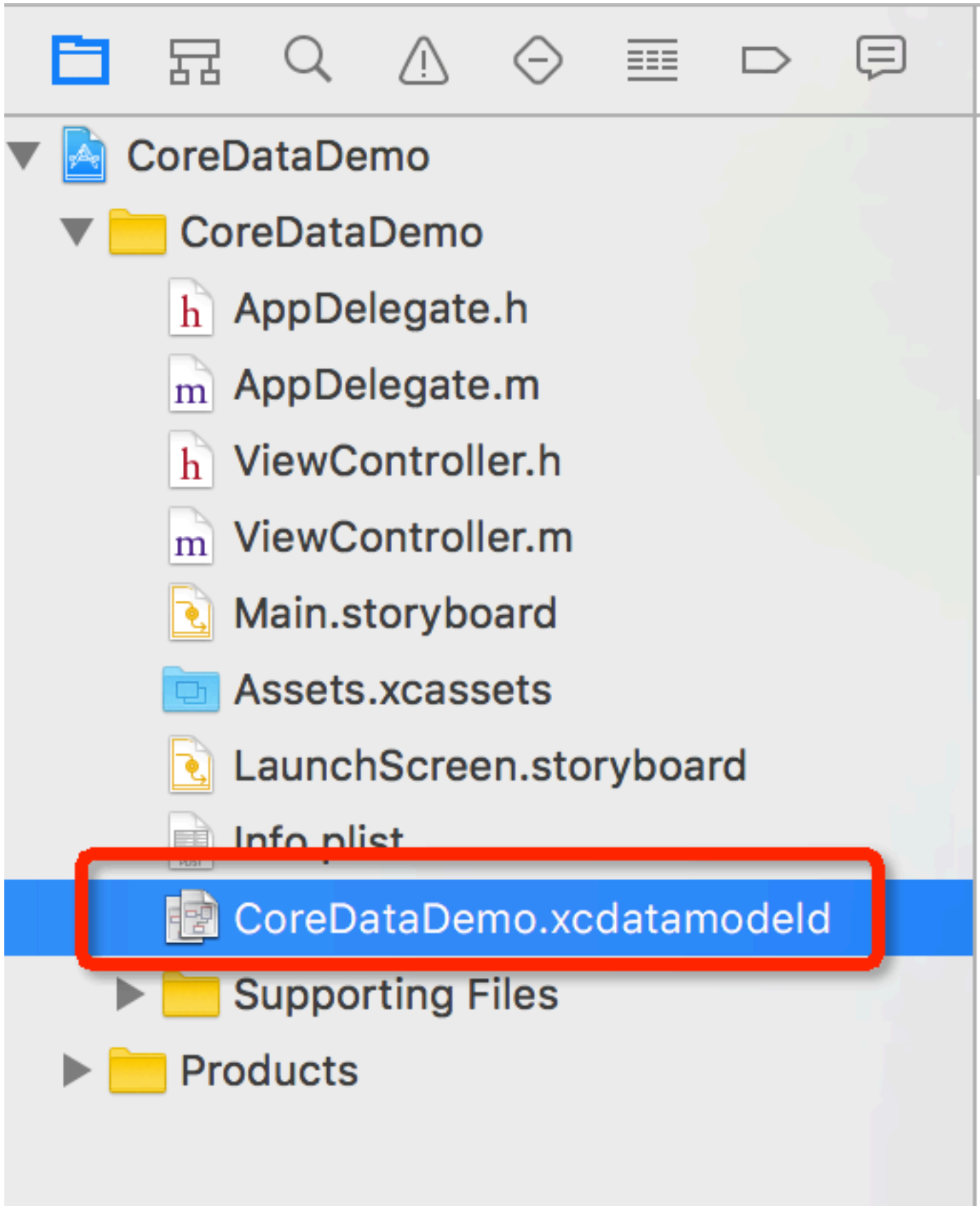
☐ Include UI Tests

Cancel

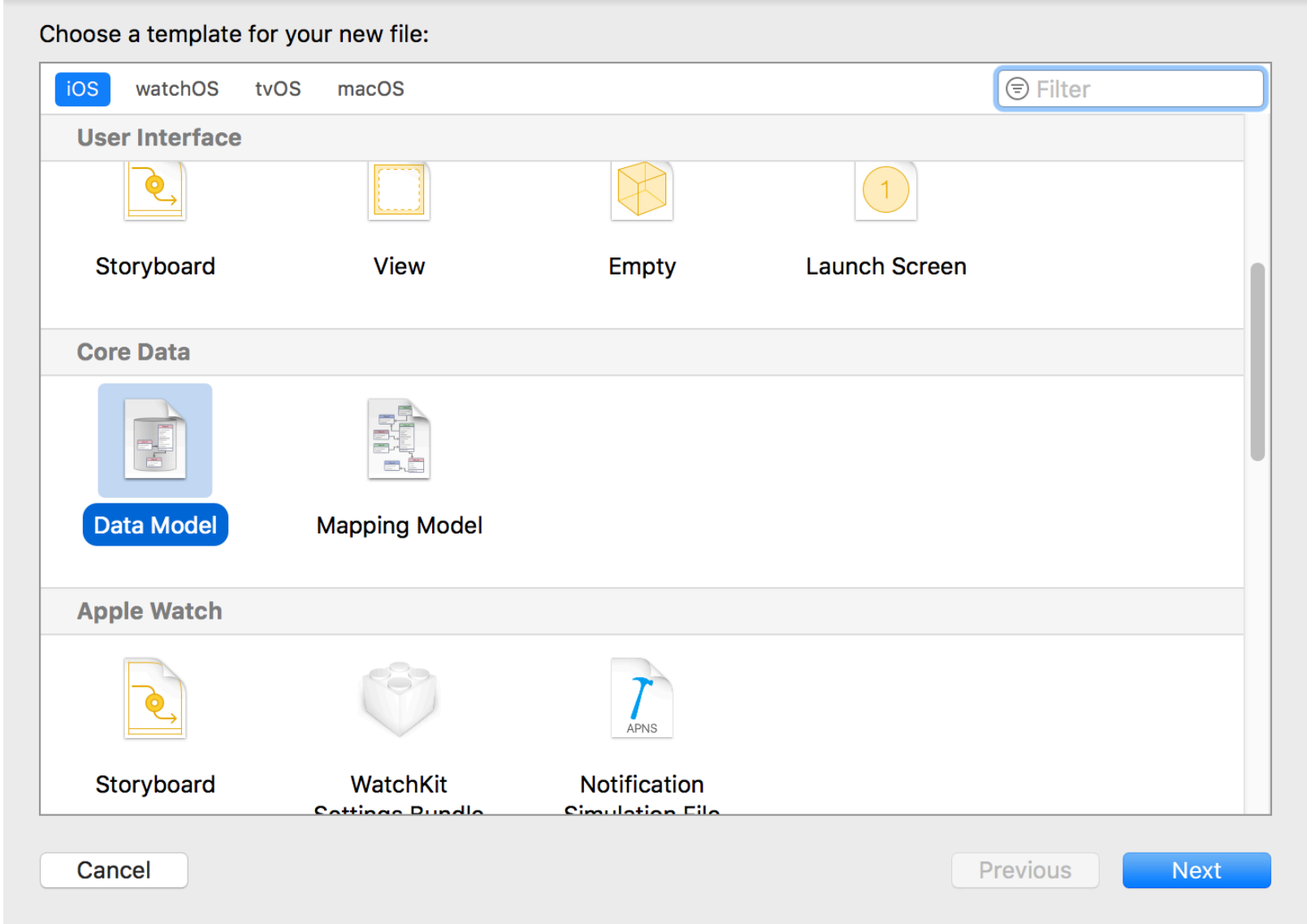
Previous

Next

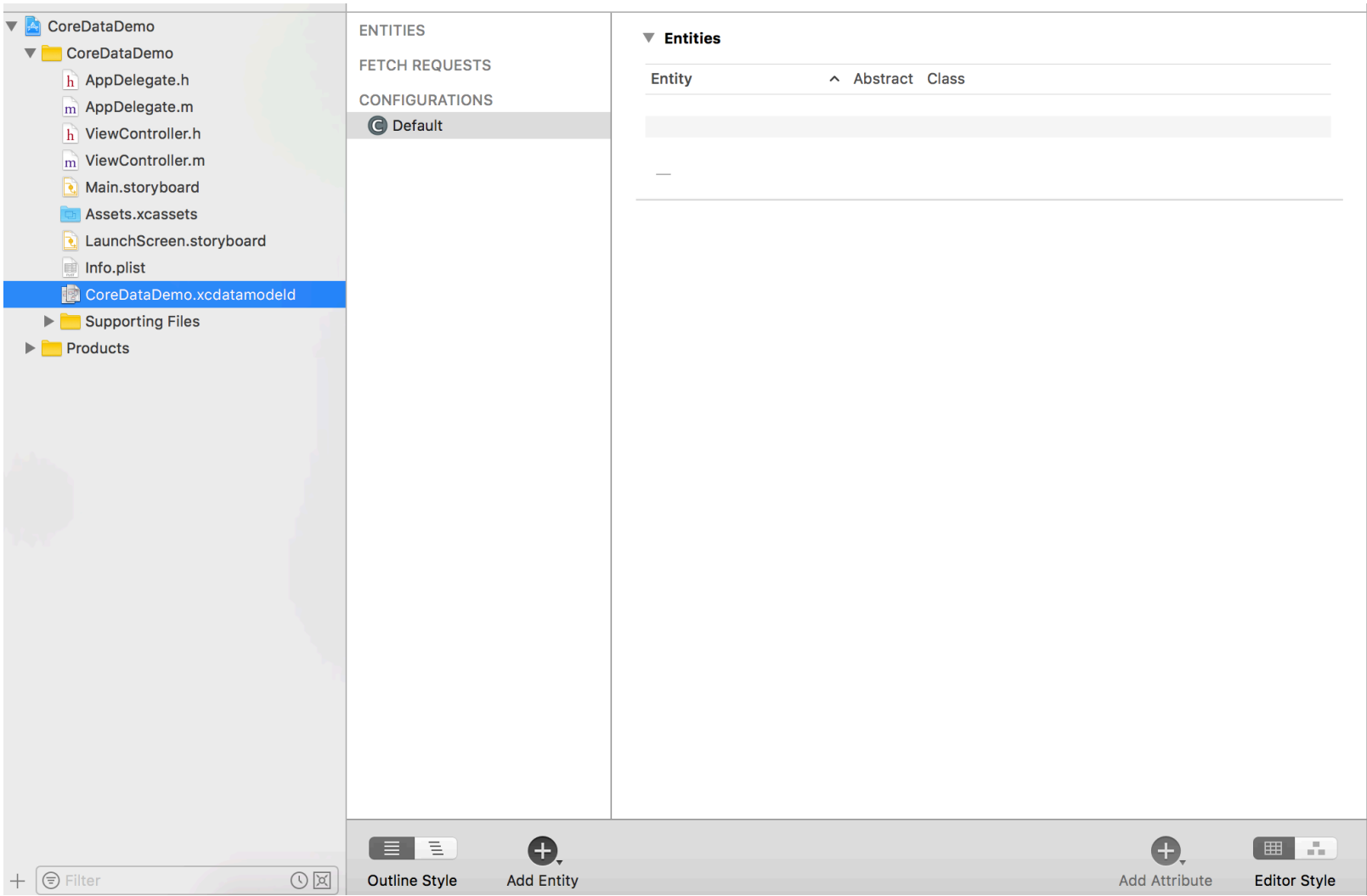
它是一个 .xcdatamodeld 格式的文件：



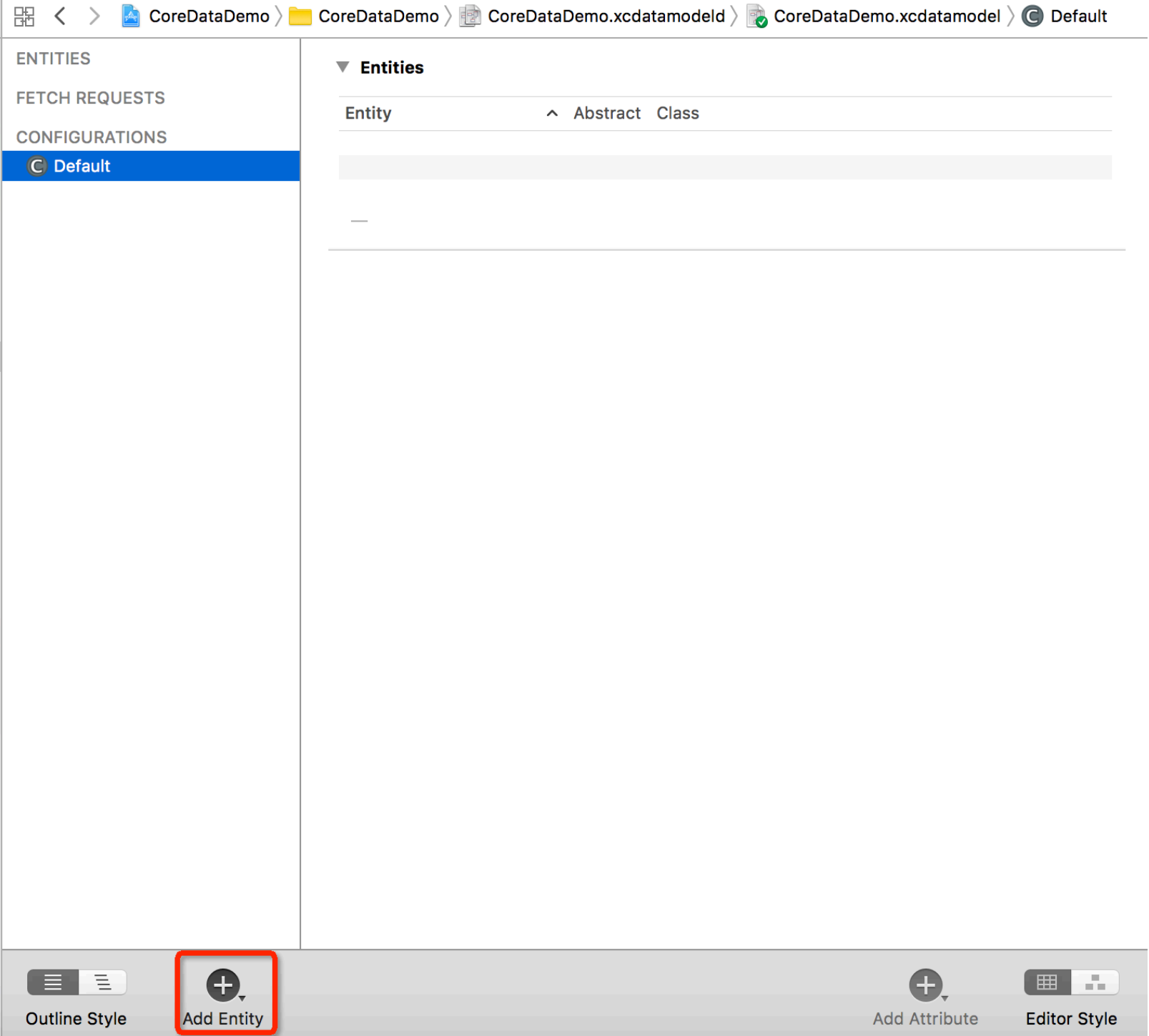
如果创建时没有勾选 CoreData，当然也可以在 File -> new -> file 里手动添加这个文件：



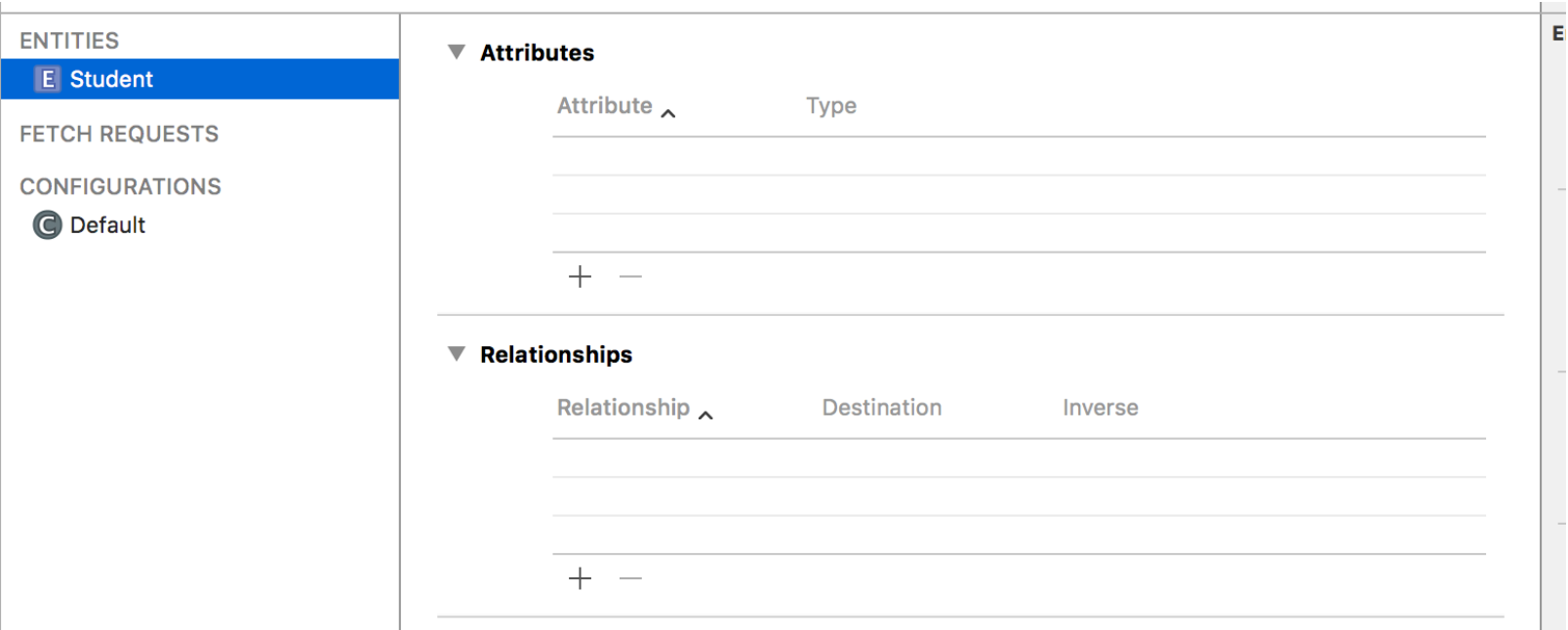
然后打开这个文件，是这样的：



点击下面的 Add Entity 按钮可以添加一个Entity，也就是一个数据实体，相当于数据库中的一张表：



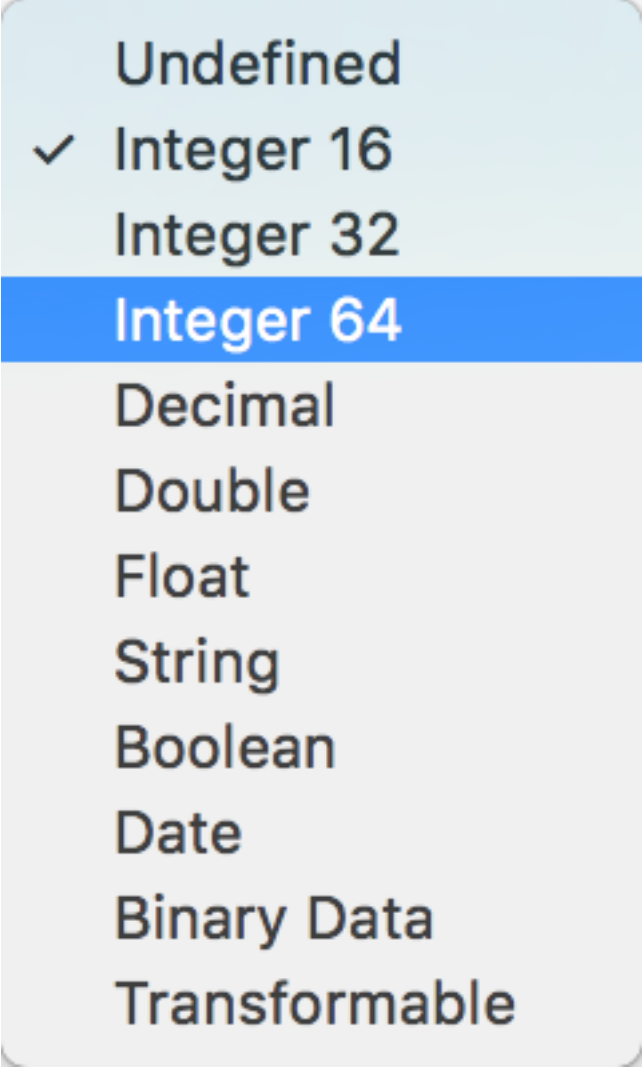
点击添加一个 Student 的 Entity：



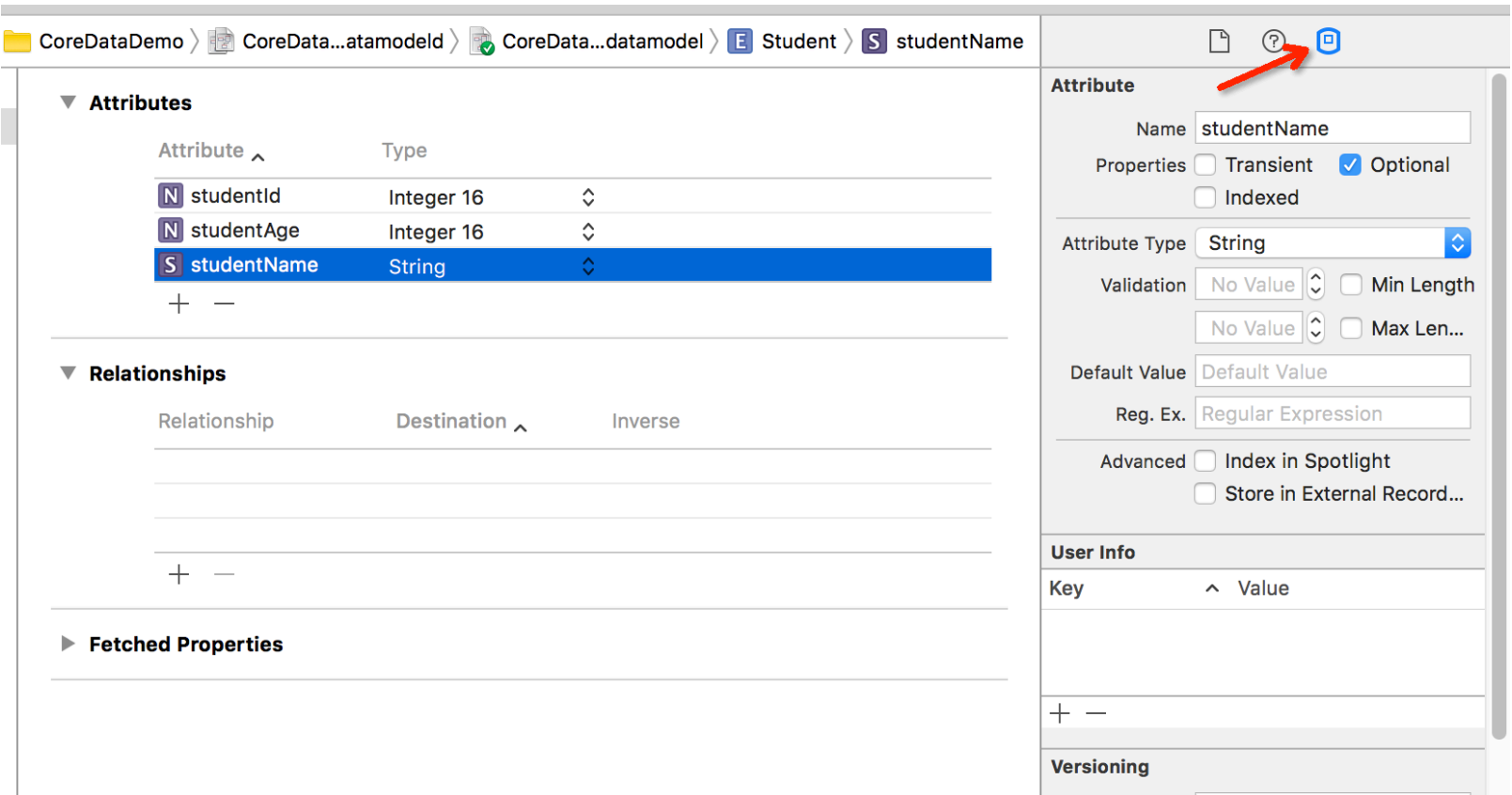
图中的 Attributes 是定义属性的地方，Relationships 是定义关联关系的地方，点击加号可以添加。下面来给 Student 添加三个字段：studentName, studentAge, studentId：

▼ Attributes			
Attribute ^		Type	
N	studentId	Integer 16	↕
N	studentAge	Integer 16	↕
S	studentName	String	↕
+		-	

下面是 CoreData 里支持的数据类型：



选中一个字段，可以在右侧的面板中对它做一些自定义：



例如在 validation 里对数据做一些限制，字符串的长度，数字类型的最大最下值；设置索引、默认值等。不同的数据类型可以设置不同的内容，一般维持默认就可以。

另外对于每一个 entity 实体类，Build 过后 Xcode 都会自动帮我们生成相应的实体类代码，生成的代码不会在工程目录中显示出来，但是可以通过导入头文件索引到；当然也可以配置成手动生成的，选中对应的 Entity 然后点击右侧面板的 Codegen，把 ClassDefinition 修改成 Manual/None，然后 Xcode 就不会再自动生成了。

Entity

Name

Student

☐

Abstract Entity

Parent Entity

No Parent Entity

Class

Name

Student

Module

Global namespace

Codegen

Manual/None

Indexes

No Content

+ —

Constraints

No Content

+ —

User Info




Key

^

Value

+ —

另外，Xcode 自动生成的代码都是 **Swift** 语言的，如果想改成Objc，可以在这里改：



Identity and Type

Name CoreDataDemo.xcdatamodel

Type Default - Core Data Model

Location Relative to Group

CoreDataDemo.xcdatamodel

Full Path /Users/chaoyang805/Desktop/CoreDataDemo/CoreDataDemo/CoreDataDemo.xcdatamodel
CoreDataDemo.xcdatamodel

On Demand Resource Tags

Add to a target to enable tagging

Core Data Model

Identifier Model Version Identifier


Tools Version

Minimum Automatic (Xcode 8.0)

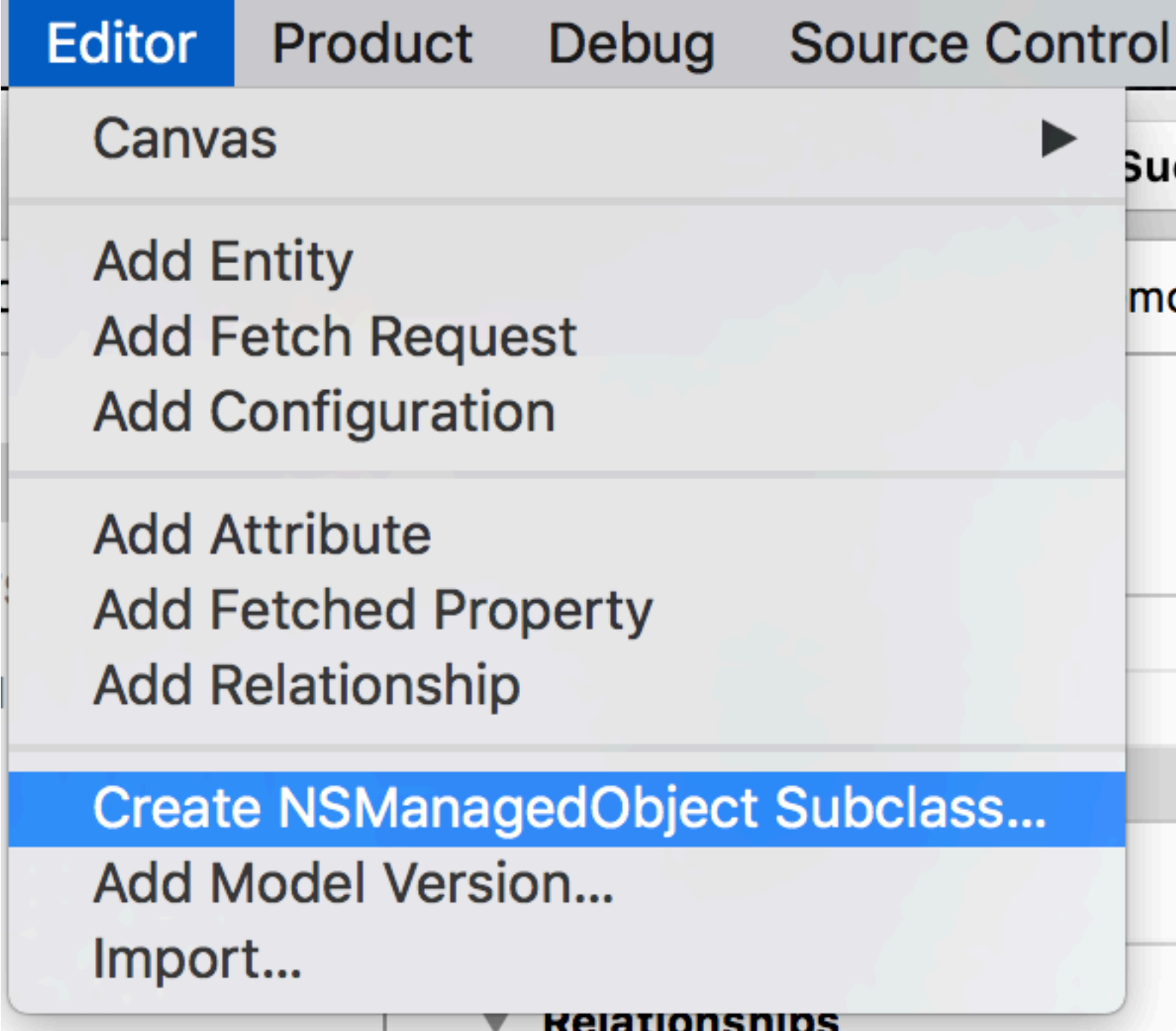
Code Generation

Language Swift

Target Membership

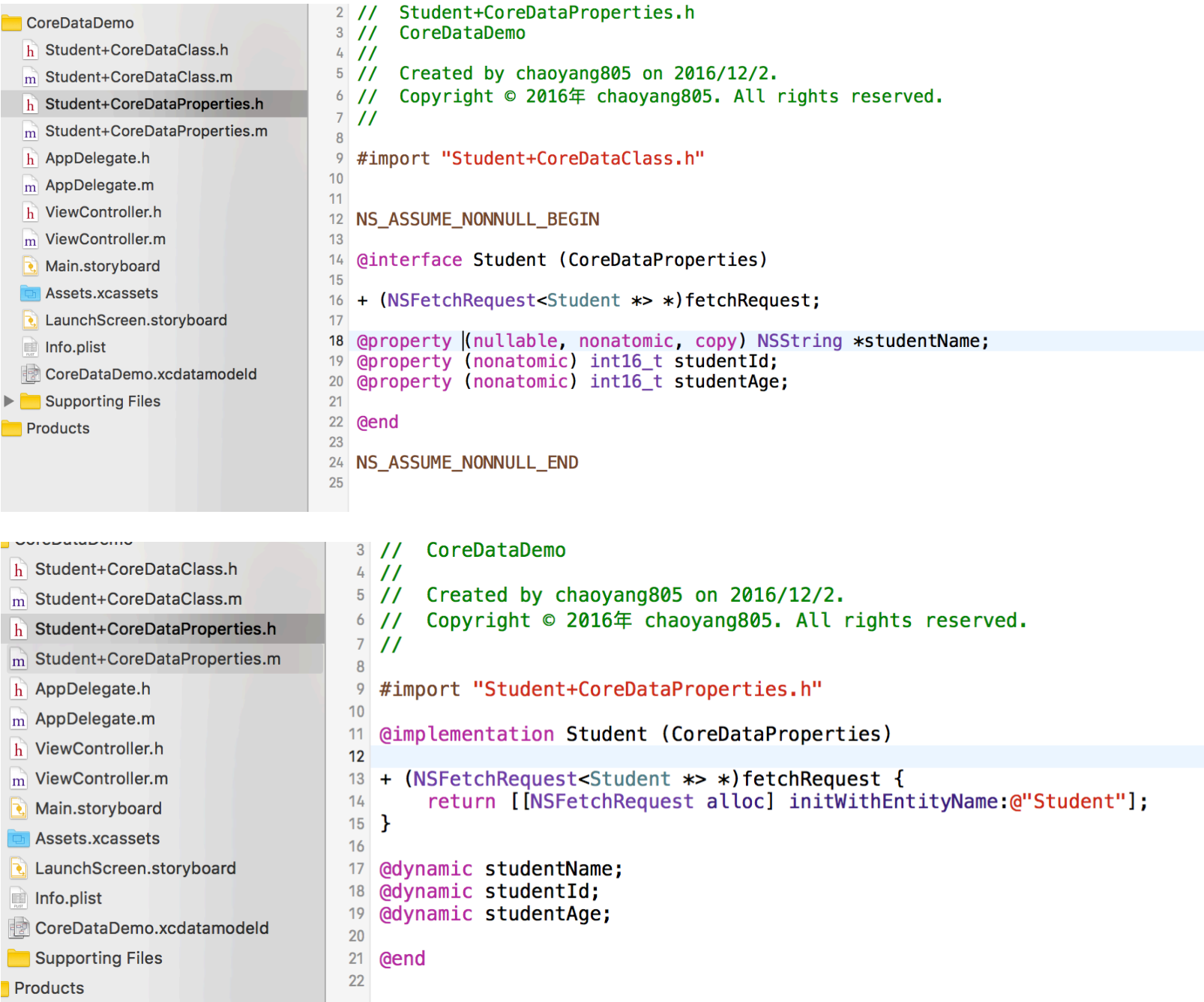
☒  CoreDataDemo

这个时候也可以通过 Editor -> Create NSManagedObject Subclass 来生成相应的实体类：



需要注意的是，如果前面有自动生成过这些类文件，手动生成后可能会编译出错，因为工程里会索引到两份同样的代码，这个时候需要 Clean 一下工程再 Build 即可。

下面是自动生成的实体类：



到此为止，CoreData 的数据模型就创建好了。

二、CoreData 栈的创建

数据模型创建好之后，想要使用 CoreData 进行数据持久化，下一步就是初始化 CoreData 栈了。下面

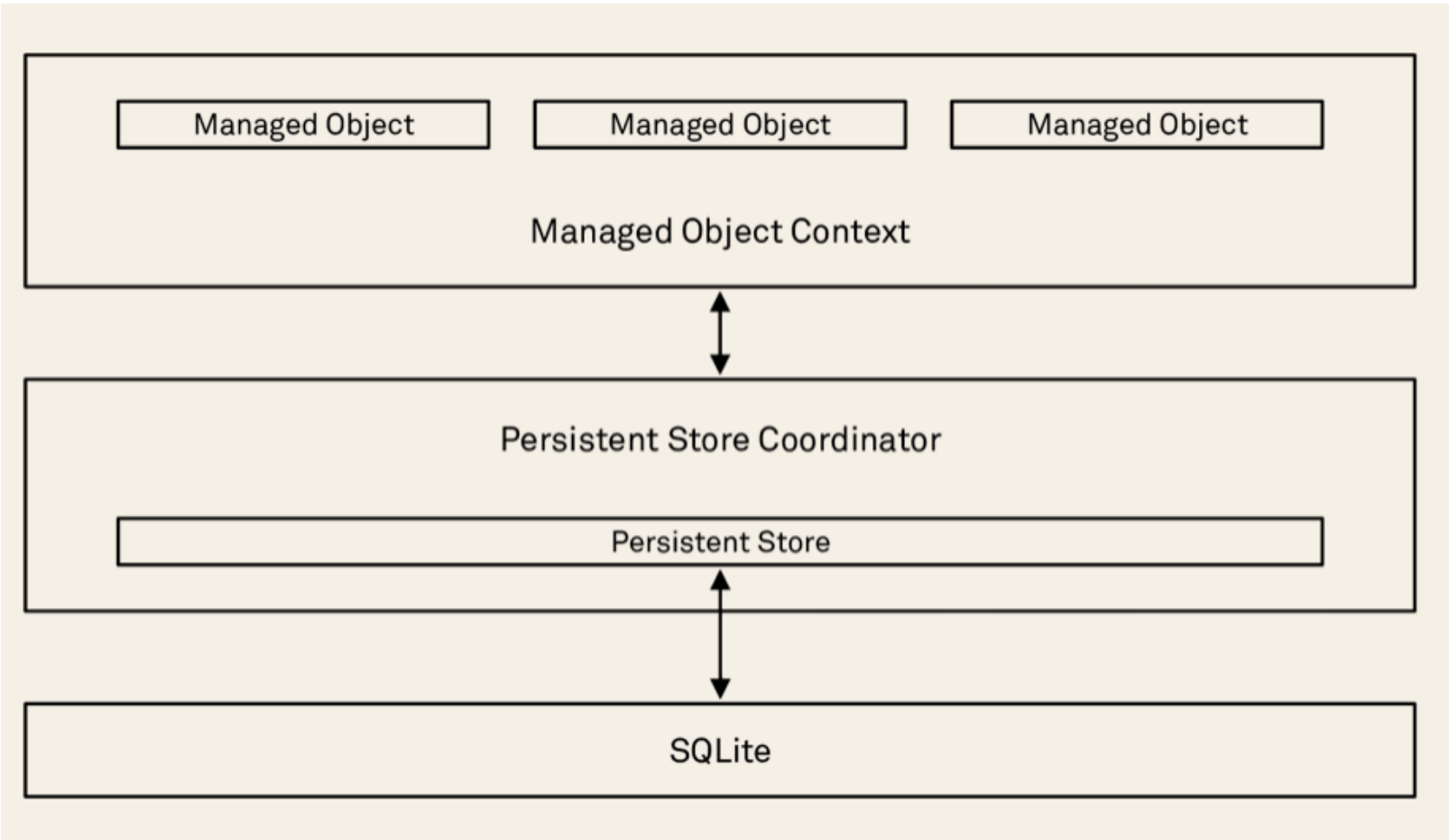
是苹果文档里对 CoreData 栈的介绍：

The Core Data stack is a collection of framework objects that are accessed as part of the initialization of Core Data and that mediate between the objects in your application and external data stores.

CoreData 栈是 CoreData 初始化被访问的框架对象的集合，以及应用中数据对象和外部数据存储的媒介。CoreData 的初始化需要一步步地初始化 CoreData 栈上的三个对象结构，它们分别是：

- 1. NSManagedObjectContext — 描述了数据模型的结构信息
- 2. NSPersistentStoreCoordinator — 数据持久层和内存对象模型的协调器
- 3. NSManagedObjectContext — 内存中 managedObject 对象的上下文

下图是 CoreData 栈的结构，图片来自 objc.io 的图书 《CoreData》：



下面来用代码演示 CoreData 栈的初始化过程：

1、加载 ManagedObjectModel

第一步是创建 NSManagedObjectContext 对象，它需要通过上文中讲的数据模型文件来创建：

```
1  @interface AppDelegate ()
2
3  @property (nonatomic, readwrite, strong) NSManagedObjectContext *managedObjectContext
4
5  @end
6
7  @implementation
8      // 使用懒加载的方式初始化
9  - (NSManagedObjectContext *)managedObjectContext {
10      if (!_managedObjectContext) {
11          // url 为CoreDataDemo.xcdatamodeld, 注意扩展名为 momd, 而不是 xcdatamodeld
12          NSURL *modelURL = [[NSBundle mainBundle] URLForResource:@"CoreDataDemo" withExtension:@"momd"];
13          _managedObjectContext = [[NSManagedObjectContext alloc] initWithConfiguration:nil];
14      }
15      return _managedObjectContext;
16  }
17
18  @end
```

2、创建 PersistentStoreCoordinator

创建好 `managedObjectModel` 后就可以来创建 `persistentStoreCoordinator` 了，因为它的创建需要用到 `managedObjectModel`，`managedObjectModel` 告诉了 `persistentStoreCoordinator` 数据模型的结构，然后 `persistentStoreCoordinator` 会根据对应的模型结构创建持久化的本地存储。

```
1  @interface AppDelegate ()
2
3  @property (nonatomic, readwrite, strong) NSManagedObjectModel *managedObjectModel;
4  @property (nonatomic, readwrite, strong) NSPersistentStoreCoordinator *persistentStoreCoordinator;
5
6  @end
7
8  @implementation AppDelegate
9
10 - (NSManagedObjectModel *)managedObjectModel {
11     if (!_managedObjectModel) {
12         NSURL *modelURL = [[NSBundle mainBundle] URLForResource:@"CoreData" withExtension:@"mom"];
13         _managedObjectModel = [[NSManagedObjectModel alloc] initWithContentsOfURL:modelURL];
14     }
15     return _managedObjectModel;
16 }
17 // 同样使用懒加载创建
18 - (NSPersistentStoreCoordinator *)persistentStoreCoordinator {
19     if (!_persistentStoreCoordinator) {
20         // 创建 coordinator 需要传入 managedObjectModel
21         _persistentStoreCoordinator = [[NSPersistentStoreCoordinator alloc] initWithManagedObjectModel:_managedObjectModel];
22         // 指定本地的 sqlite 数据库文件
23         NSURL *sqliteURL = [[self documentDirectoryURL] URLByAppendingPathComponent:@"sqlite.sqlite"];
24         NSError *error;
25         // 为 persistentStoreCoordinator 指定本地存储的类型，这里指定的是 SQLite
26         [_persistentStoreCoordinator addPersistentStoreWithType:NSSQLiteStoreType
27                                                         configuration:nil
28                                                         URL:sqliteURL
29                                                         options:nil
30                                                         error:&error];
31         if (error) {
32             NSLog(@"failed to create persistentStoreCoordinator %@", error);
33         }
34     }
35     return _persistentStoreCoordinator;
36 }
37
38 // 用来获取 document 目录
39 - (nullable NSURL *)documentDirectoryURL {
40     return [[NSFileManager defaultManager] URLsForDirectory:NSDocumentDirectory
41                                     inDomains:NSDocumentDirectoryInMyDomain];
42 }
43 @end
```

3、创建 ManagedObjectContext

上面两步都完成之后，下面来创建 `managedObjectContext`，这也是平时操作 CoreData 主要会用到的对象：

```
1  @interface
2  ...
3  @property (nonatomic, readwrite, strong) NSManagedObjectContext *managedObjectContext;
4  ...
5  @end
6  @implementation
7  ...
8
```



```
9 - (NSManagedObjectContext *)context {
10     if (!_context) {
11         // 指定 context 的并发类型： NSMainQueueConcurrencyType 或 NSPrivateQueueConcurrencyType
12         _context = [[NSManagedObjectContext alloc ] initWithConcurrencyType:NSMainQueueConcurrencyType];
13         _context.persistentStoreCoordinator = self.persistentStoreCoordinator;
14     }
15     return _context;
16 }
17 ...
18 @end
```

至此，CoreData 栈的初始化就创建完成了。以后操作 CoreData 就可以通过 context 属性来完成，操作完之后调用 context 的 save 方法就可以数据持久化到本地。

顶

0

踩

0

+

☆

👤

🔗

👤

💬

上一篇

Swift 3.0 中的新变化

下一篇

CoreData 从入门到精通 二 数据的增删改查

相关文章推荐

CoreData 从入门到精通 二 数据的增删改查

objc.io 4.2 一个简单的CoreData (转)

iOS： WKWebView与UIWebView的区别

coredata 最简单例子

CoreData

objc.io系列文章中文翻译汇总

iphone数据存储之- - Core Data的使用（一）

UITableView，UICollectionView和CoreData完美结...

Core Data入门

iOS 持久化存储之CoreData VS 直接SQLite



botox瘦腿



肚脐减肥法



电动摩托车



本科录取人数



mac系统升级



无限流量卡



到烟台机票



一年拿本科文



卖狗网



电动车电池价



特价机票



买本科...



无限流量卡



电动摩托车



小型四轮电动



到烟台机票



夜大



肚脐减肥法



卖狗网

猜你在找

- 【直播】机器学习&数据挖掘7周实训--韦玮

【直播】3小时掌握Docker最佳实战-徐西宁

【直播】计算机视觉原理及实战--屈教授

【直播】机器学习之矩阵--黄博士

【直播】机器学习之凸优化--马博士
- 【套餐】系统集成项目管理工程师顺利通关--徐朋

【套餐】机器学习系列套餐（算法+实战）--唐宇迪

【套餐】微信订阅号+服务号Java版 v2.0--翟东平

【套餐】微信订阅号+服务号Java版 v2.0--翟东平

【套餐】Javascript 设计模式实战--曾亮

查看评论

暂无评论

您还没有登录,请[登录](#)或[注册](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

 [网站客服](#)  [杂志客服](#)  [微博客服](#)  webmaster@csdn.net  400-660-0108 | 北京创新乐知信息技术有限公司 版权所有

江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2017, CSDN.NET, All Rights Reserved 

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

 [网站客服](#)  [杂志客服](#)  [微博客服](#)  webmaster@csdn.net  400-660-0108 | 北京创新乐知信息技术有限公司 版权所有

江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2017, CSDN.NET, All Rights Reserved 

关闭



瘦腿的快方法















