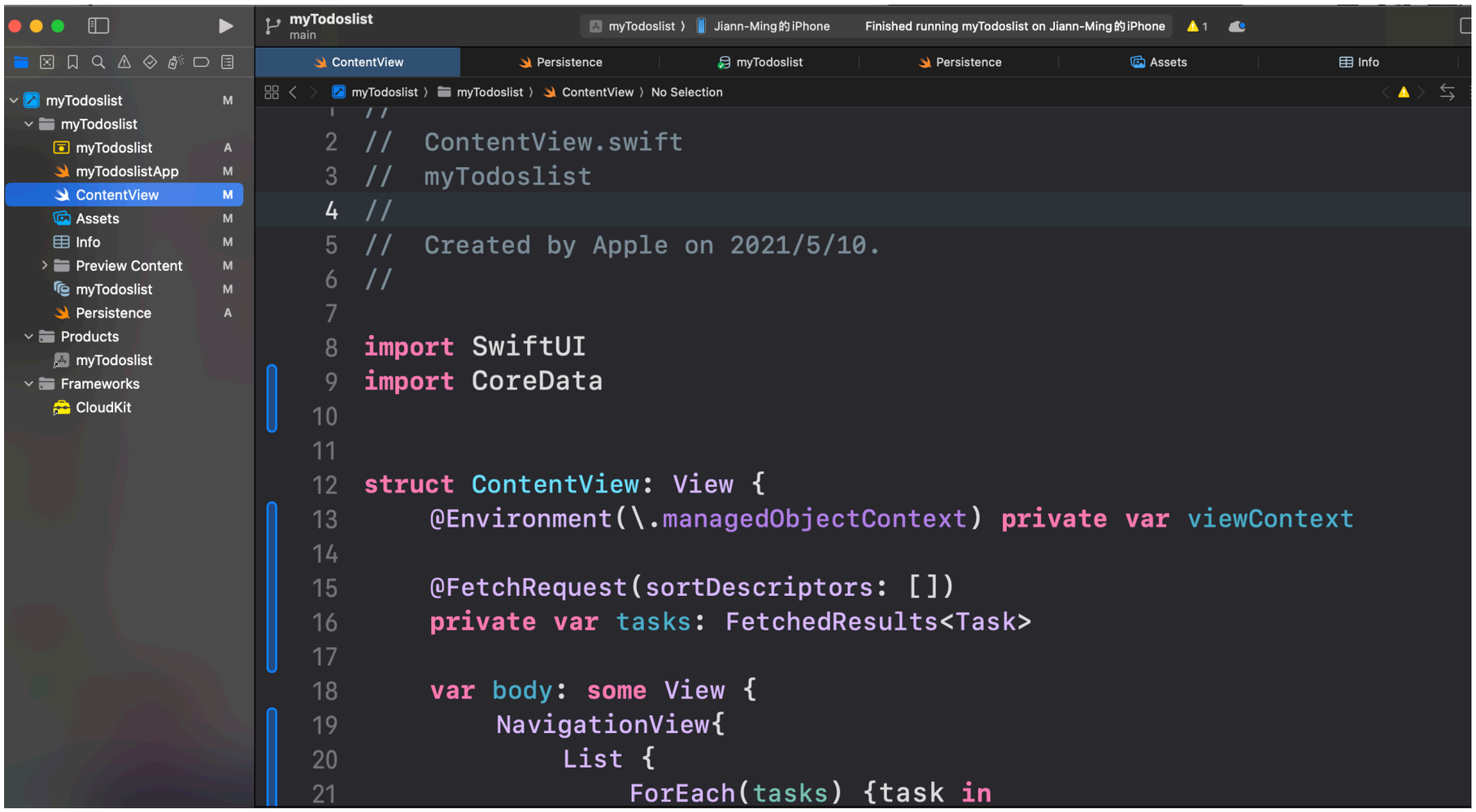
iCloud雲端APP

https://www.youtube.com/watch?v=BtotQmDVSRo&si=clQ4JmvLlXvuQyYW

iCloud雲端APP



How to sync SwiftData with iCloud

Paul Hudson @twostraws September 30th 2023

Updated for Xcode 15

SwiftData comes with built-in support for iCloud syncing using CloudKit. It's absurdly easy to use – literally zero code required in some situations – but it does *not* support the public or shared database.

Important: If you need to use the public or shared database, you need to use `NSPersistentCloudKitContainer` either by itself or with SwiftData coexistence.

If you want to sync your app's data to iCloud, go to the Signing & Capabilities settings for your app's target, then:

- Add the iCloud capability.
- Select CloudKit from its options.
- Press + to add a new CloudKit container, or select one of your existing ones.
- Add the Background Modes capability.

如何將SwiftData與iCloud同步

保羅·哈德森 @twostraws 2023年9月30日

為Xcode 15更新

SwiftData內建支援使用CloudKit進行iCloud同步。 它非常容易使用——在某些情況下簡直是零程式——但它不支援公共或共享資料庫。

重要資訊：如果您需要使用公共或共享資料庫，您需要單獨使用`NSPersistentCloudKitContainer`或與SwiftData共存使用。

如果您想將應用程式的資料同步到iCloud，請轉到應用程式目標的“簽名和功能”設定，然後：

- 新增iCloud功能。
- 從其選項中選擇CloudKit。

- 新增iCloud功能。
- 從其選項中選擇CloudKit。
- 按+新增新的CloudKit容器，或選擇現有容器之一。
- 新增背景模式功能。
- 從其選項中勾選“遠端通知”複選框。

...就是這樣：您的應用程式現在已配置為將其所有資料與iCloud同步。

提示：雖然您可以嘗試在模擬器中測試iCloud支援，但我發現它很少能很好地工作——在真實裝置上測試要好得多。

雖然配置已經完成，但*您可能*需要對SwiftData模型進行一些更改，因為CloudKit有一些非常具體的要求。 令人惱火的是，如果您不遵守這些要求，您的iCloud同步將默默地失敗，所以它們在這裡：

1. 您不能在要同步到CloudKit的任何屬性上使用`@Attribute(.unique)`
2. 所有屬性必須具有預設值，或者與其初始化器一起標記為可選。
3. 所有關係都必須標記為可選。 這是特別令人討厭™，但這是一個要求，所以我們無能為。

只要您對所有模型進行這些更改，它們就會自動同步到CloudKit。 如果您的使用者稍後從他們的裝置中刪除您的應用程式，然後重新安裝它，SwiftData將自動從iCloud中獲取他們的舊資料，並在本地同步。

使用CloudKit時，您可能會發現蘋果的CloudKit儀表板是檢視CloudKit如何儲存資料的有用資源。 您可以在https://icloud.developer.apple.com/dashboard找到它。


Core Data
SwiftUI 2.0
BeyondOnesAndZeros
imation
2HD IBM
formatted formatee
1.44MB
HD
imation
What is Core Data?
Understanding the Core Data Stack
How to create a Core Data (CRUD) App from Scratch?
SUBSCRIBE

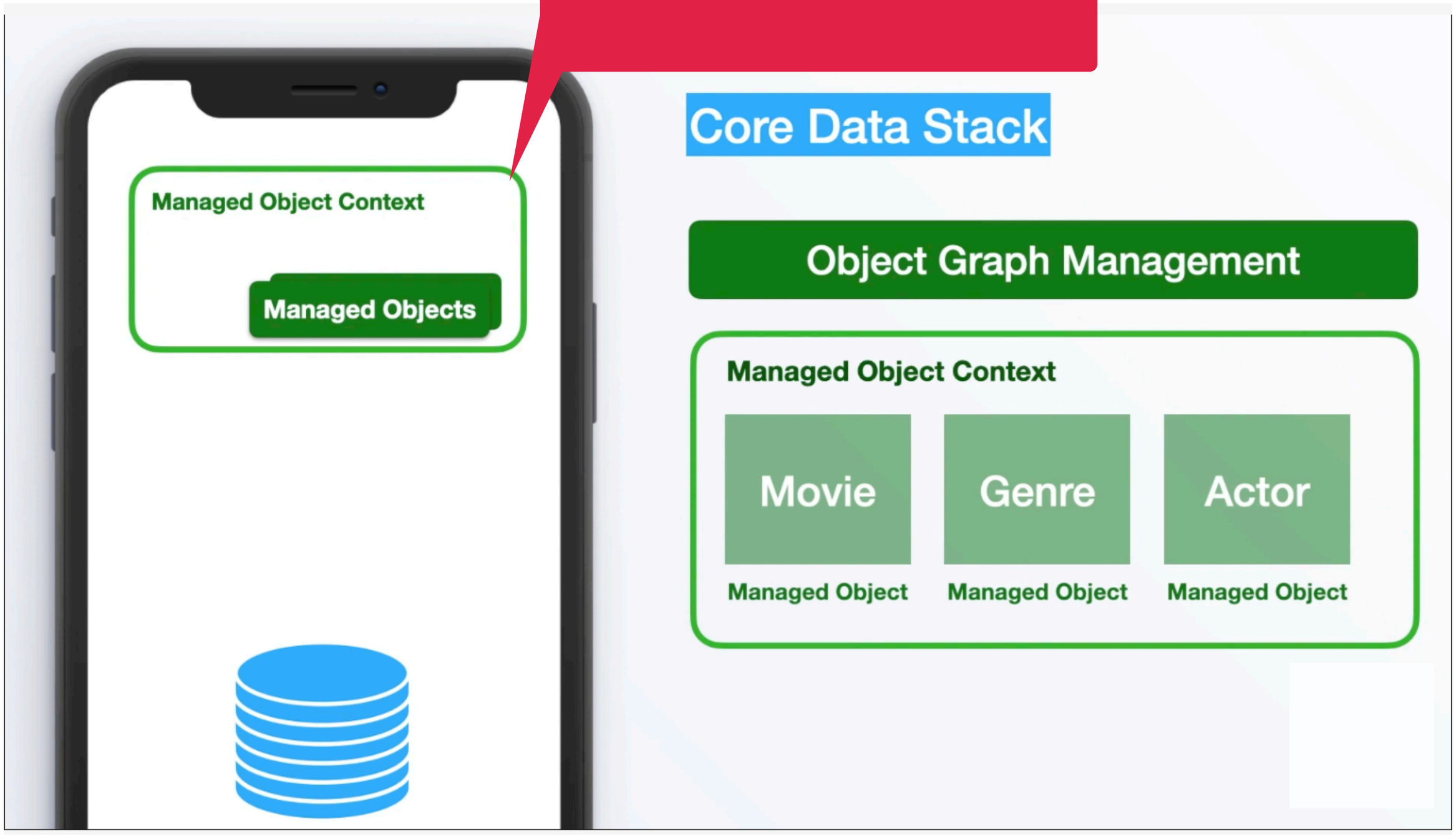
What is Core Data?

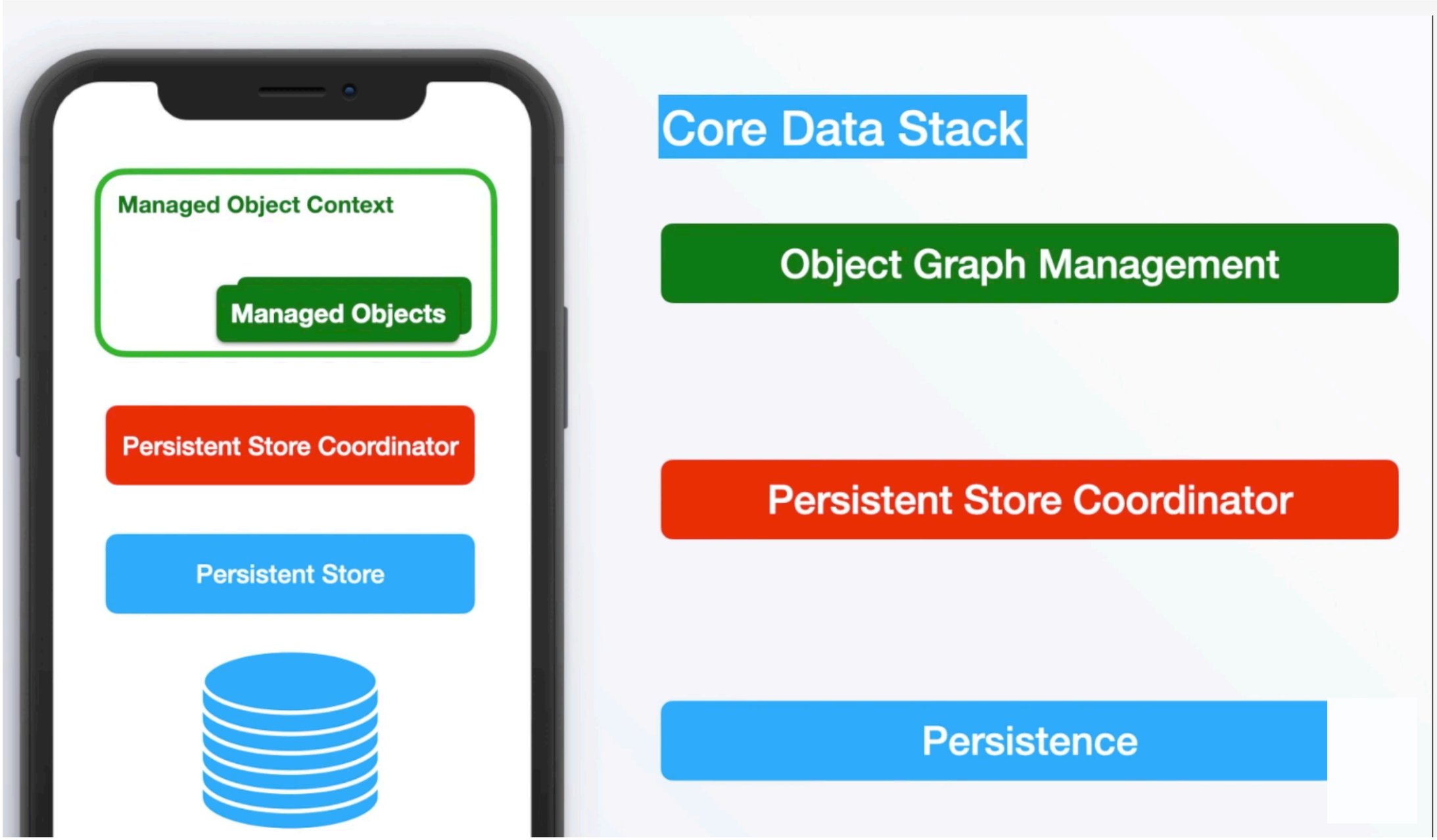
➡ **Native solution** to **store data** permanently

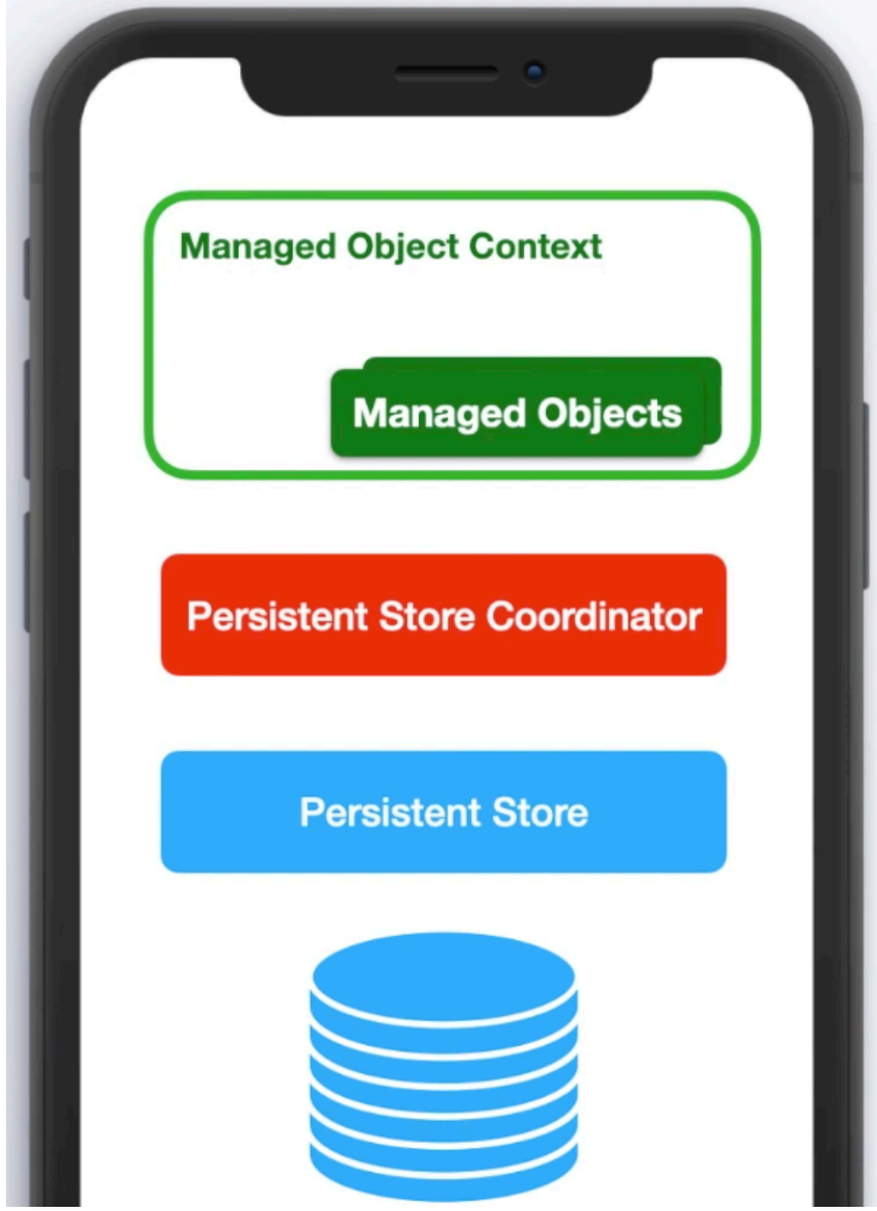
➡ Other use cases:

- **Cache data** temporary
- Support for **redo** and **undo**

受管理物件內容或內文


Managed Object Context
Managed Objects
Core Data Stack
Object Graph Management
Managed Object Context
Movie
Genre
Actor
Managed Object
Managed Object
Managed Object


Managed Object Context
Managed Objects
Persistent Store Coordinator
Persistent Store
Core Data Stack
Object Graph Management
Persistent Store Coordinator
Persistence



Core Data + SwiftUI

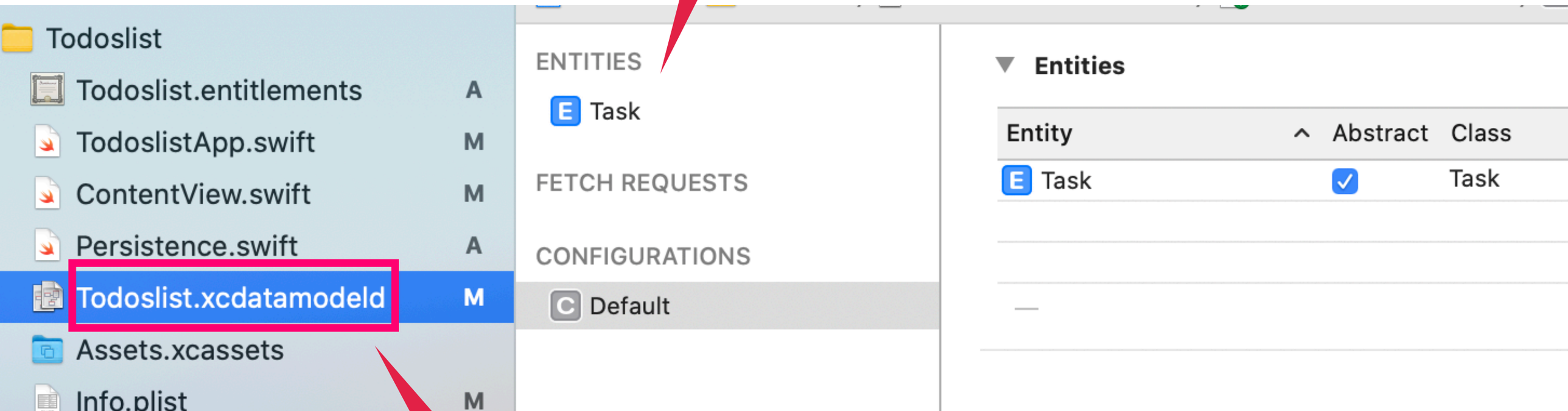
➡ **NSManagedObject** conforms to **ObservableObject**

➡ **@FetchRequest** makes fetching and displaying results very easy!

➡ **managedObjectContext** is now included in the **environment**

步驟一、新增資料模型

實體名稱



資料模型的檔案名稱

欄位date

Todoslist

Todoslist.entitlements A

TodoslistApp.swift M

ContentView.swift M

Persistence.swift A

Todoslist.xcdatamodeld M

ENTITIES

Task

FETCH REQUESTS

CONFIGURATIONS

Default

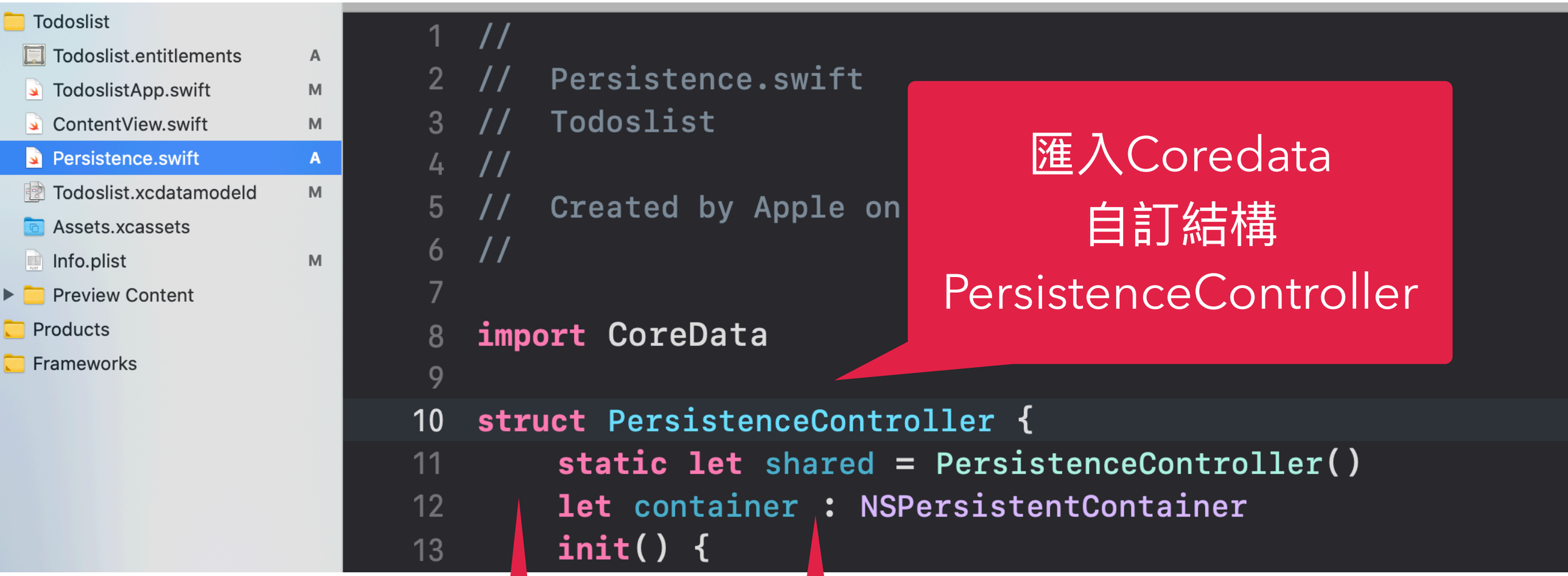
Attributes

Attribute	Type
date	Date
title	String

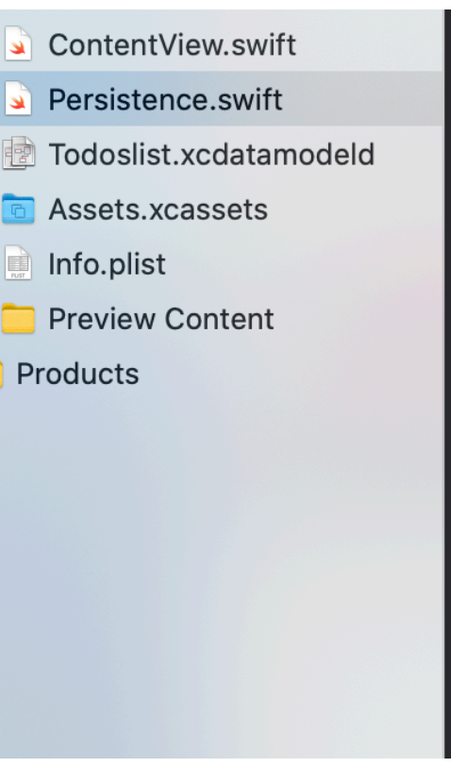
欄位title

步驟二、新增

Persistence.swift

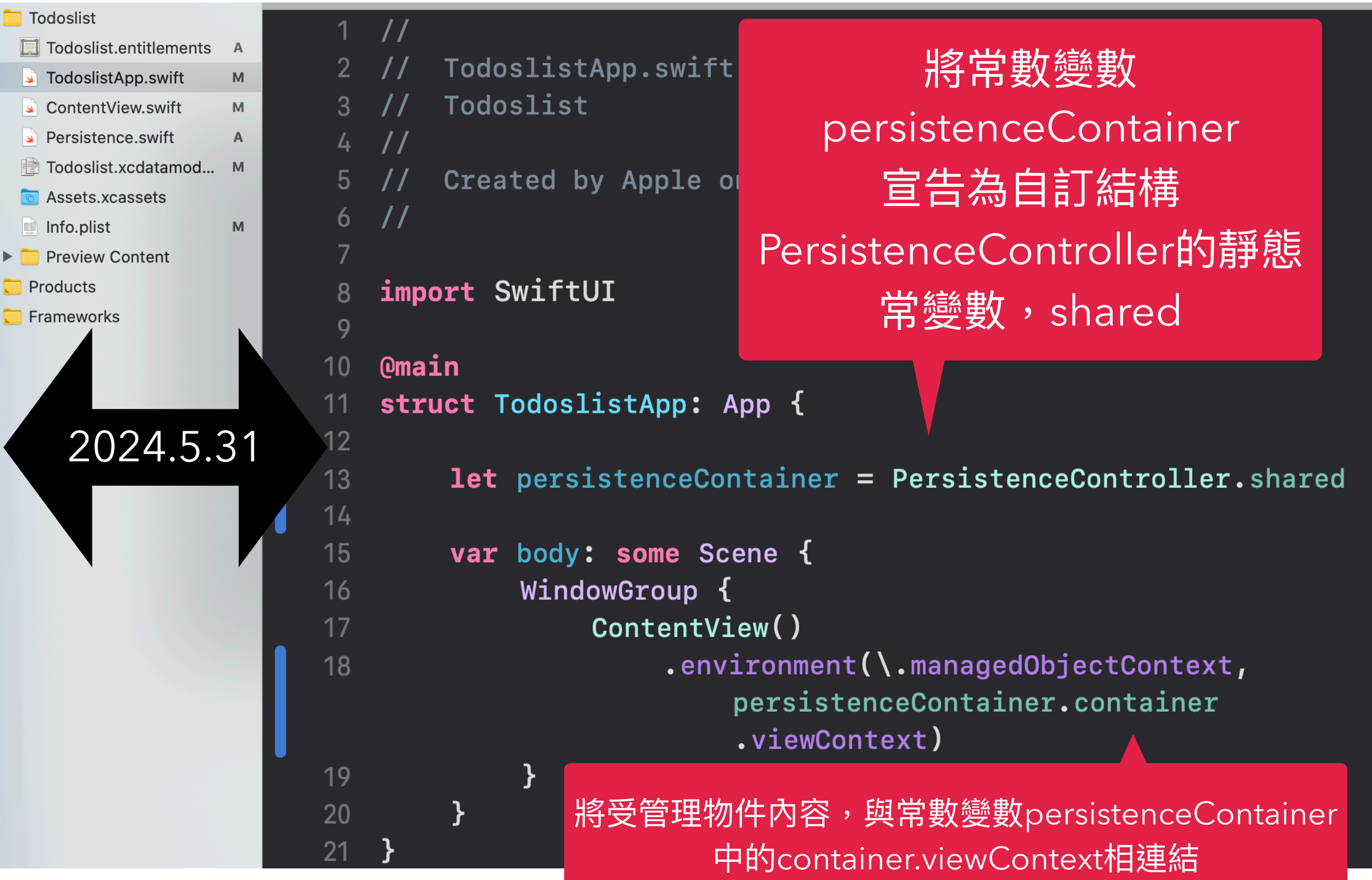

Todoslist
Todoslist.entitlements A
TodoslistApp.swift M
ContentView.swift M
Persistence.swift A
Todoslist.xcdatamodeld M
Assets.xcassets
Info.plist M
Preview Content
Products
Frameworks
1 //
2 // Persistence.swift
3 // Todoslist
4 //
5 // Created by Apple on
6 //
7
8 import CoreData
9
10 struct PersistenceController {
11 static let shared = PersistenceController()
12 let container : NSPersistentContainer
13 init() {
匯入Coredata
自訂結構
PersistenceController
宣告靜態的常數變數 shared
宣告的常數變數container，型態為PersistentContainer

將container初始化為，NSPersistenetContainer，資料模型名稱為 “Todoslist"

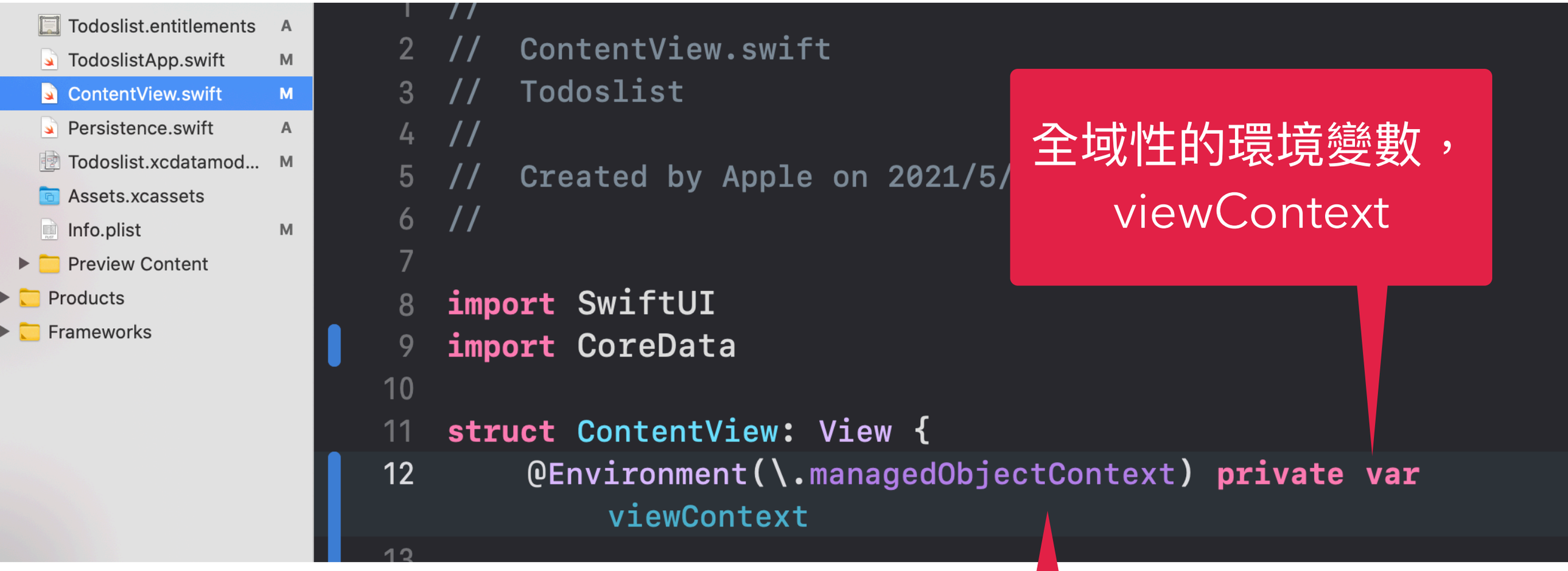



```
12      let container : NSPersistentContainer
13      init() {
14          container = NSPersistentContainer(name: "Todoslist")
15          container.loadPersistentStores { (storeDescription,
                error) in
16              if let error = error as NSError? {
17                  fatalError("Unsolved error: \(error)")
18              }
19
20          }
21      }
```

步驟三、修改main主程式


Todoslist
Todoslist.entitlements A
TodoslistApp.swift M
ContentView.swift M
Persistence.swift A
Todoslist.xcdatamod... M
Assets.xcassets
Info.plist M
Preview Content
Products
Frameworks
2024.5.31
1 //
2 // TodoslistApp.swift
3 // Todoslist
4 //
5 // Created by Apple o
6 //
7
8 import SwiftUI
9
10 @main
11 struct TodoslistApp: App {
12
13   let persistenceContainer = PersistenceController.shared
14
15   var body: some Scene {
16   WindowGroup {
17   ContentView()
18   .environment(\.managedObjectContext,
persistenceContainer.container
.viewContext)
19   }
20   }
21 }
將常數變數
persistenceContainer
宣告為自訂結構
PersistenceController的靜態
常變數，shared
將受管理物件內容，與常數變數persistenceContainer
中的container.viewContext相連結

步驟四A、
在ContentView中宣告
全域性的環境變數


Todoslist.entitlements A
TodoslistApp.swift M
ContentView.swift M
Persistence.swift A
Todoslist.xcdatamod... M
Assets.xcassets
Info.plist M
Preview Content
Products
Frameworks
1 //
2 // ContentView.swift
3 // Todoslist
4 //
5 // Created by Apple on 2021/5/
6 //
7
8 import SwiftUI
9 import CoreData
10
11 struct ContentView: View {
12 @Environment(\.managedObjectContext) private var
viewContext
全域性的環境變數，
viewContext
受管理物件內容

步驟四B、使用FetchRequest抓取並顯示結果

```
struct ContentView: View {
    @Environment(\.managedObjectContext) private var
        viewContext

    @FetchRequest(sortDescriptors: [])

    private var tasks: FetchedResults<Task>


    var body: some View {
        NavigationView {
            List {
                ForEach(tasks) {task in
                    Text(task.title ?? "Untitled")

                }.onDelete(perform: deleteTasks)
            }
```

使用FetchRequest
抓取並顯示結果

宣告private變數tasks為
資料實體Task的抓取結果

使用NavigationView
使用List

執行deleteTasks方法


Todoslist
Todoslist.entitlements A
TodoslistApp.swift M
ContentView.swift M
Persistence.swift A
Todoslist.xcdatamod... M
Assets.xcassets
Info.plist M
Preview Content
Products
Frameworks
var body: some View {
NavigationView {
List {
ForEach(tasks) {task in
Text(task.title ?? "Untitled")
}.onDelete(perform: deleteTasks)
}
.navigationTitle("Todo List")
.navigationBarItems(trailing: Button("Add Task"){
addTask()
})
}
}
執行addTask()方法

步驟四C、定義saveContext、deleteTasks、addTasks

```
32      private func saveContext(){
33          do {
34              try viewContext.save()
35          } catch {
36              let error = error as NSError
37              fatalError("Unsolved Error: \(error)")
38          }
39      }
40      private func deleteTasks(offsets: IndexSet){
41          withAnimation{
42              offsets.map{
                    tasks[$0]}.forEach(viewContext.delete)
43              saveContext()
44          }
45      }
46      private func addTask(){
47          withAnimation{
48          let newTask = Task(context: viewContext)
49          newTask.title = "New Task \(Date())"
50          newTask.date = Date()
51          saveContext()
52          }
53      }
```

程式碼 saveContext

```
    private func saveContext(){
        do {
            try viewContext.save()
        } catch {
            let error = error as NSError
            fatalError("Unsolved Error: \(error)")
        }
    }
```

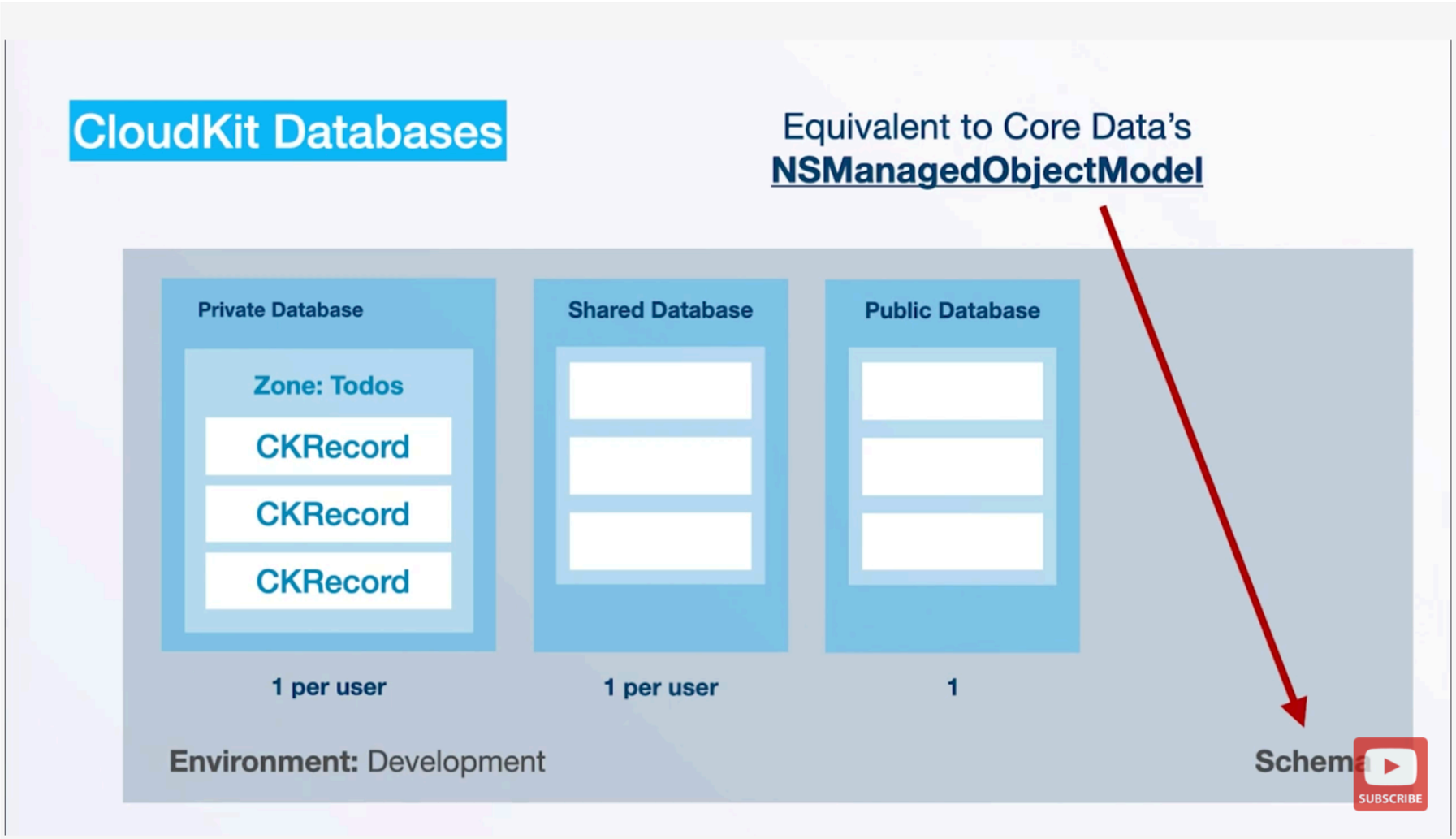
程式碼 deleteTasks

```
    private func deleteTasks(offsets: IndexSet){
        withAnimation{
            offsets.map{ tasks[$0]}.forEach(viewContext.delete)
            saveContext()
        }
    }
```

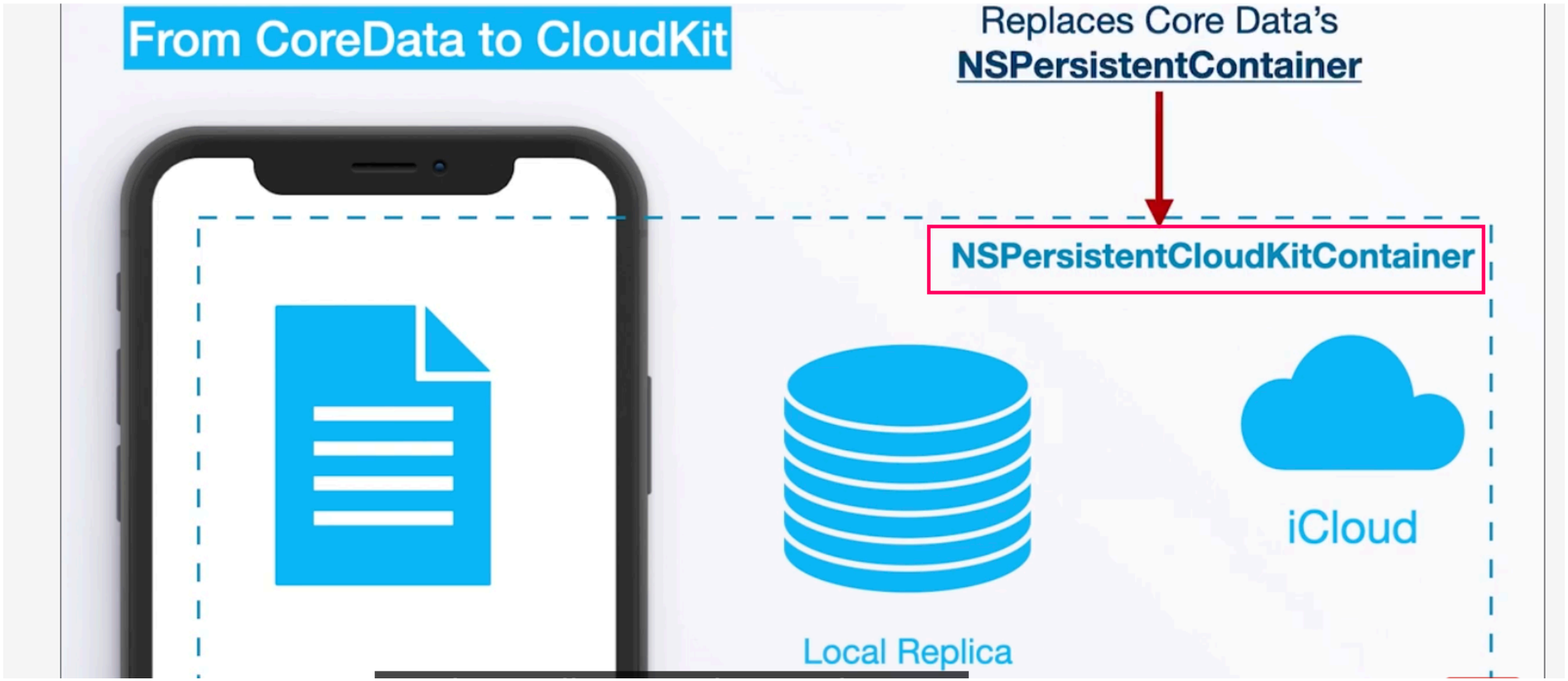
程式碼 addTasks

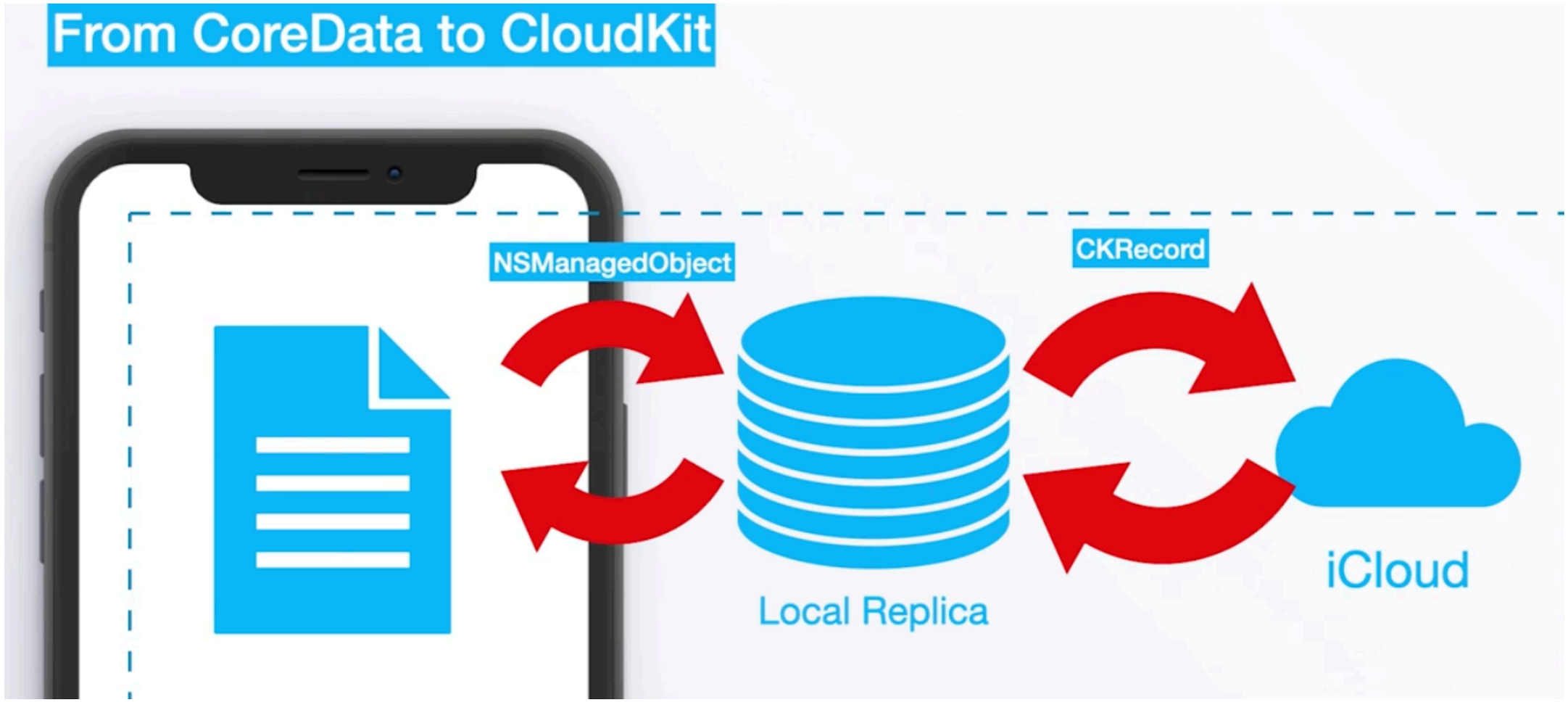
```
    private func addTask(){
        withAnimation{
        let newTask = Task(context: viewContext)
        newTask.title = "New Task \(Date())"
        newTask.date = Date()
        saveContext()
        }
    }
```

步驟五、在setting中使用iCloud



SwiftUI 2.0: CloudKit & Core Data - Getting Started


From CoreData to CloudKit
Replaces Core Data's
NSPersistentContainer
NSPersistentCloudKitContainer
Local Replica
iCloud


From CoreData to CloudKit
NSManagedObject
CKRecord
Local Replica
iCloud


iPhone 11 Pro – 14.4
2:56
Settings
JW
Jiann-Ming Wu
Apple ID, iCloud, Media & Purchases
General
Accessibility
Privacy
Passwords
Safari
News
Maps
Shortcuts
Health
Siri & Search
Photos

步驟六、使用 automatically managing singing

Product Debug Source Control Window Help 100% 週六 下午

lloCoreData 〉 iPhone 11 Pro | Finished running HelloCoreData on iPhone 11 Pro 1

HelloCoreData | ContentView.swift | *HelloCoreData.xcodeproj* | CoreDataManager.swift | 0 mach_msg_tra

HelloCoreData

General | Signing & Capabilities | Resource Tags | Info | Build Settings | Build Phases | Build Rules

PROJECT
HelloCoreData

TARGETS
HelloCoreData

+ Capability | All | Debug | Release

▼ **Signing**

Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.

Team: None

Bundle Identifier: ItLab-AMNDHU.HelloCoreData

Provisioning Profile: Xcode Managed Profile

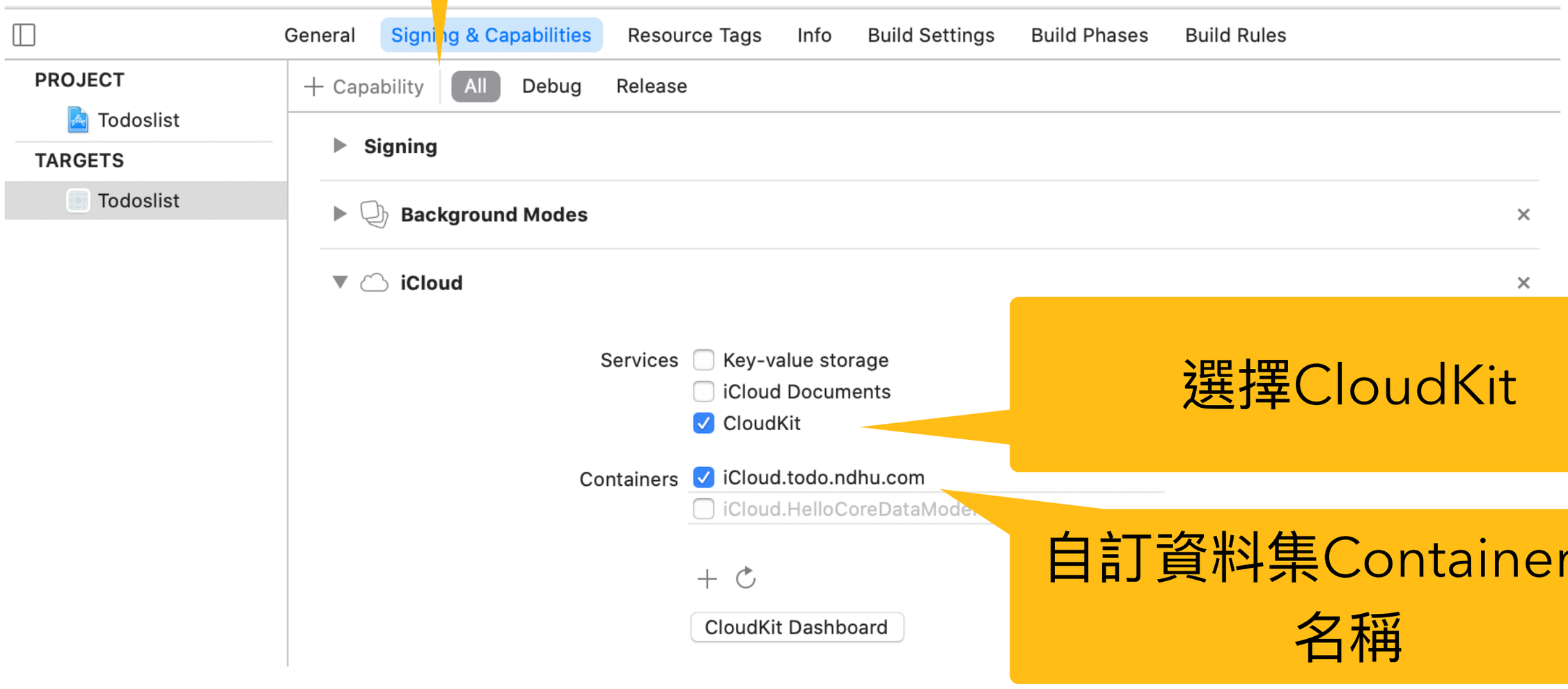
Signing Certificate: Apple Development

Status: ⚠ Signing for "HelloCoreData" requires a development team.
Select a development team in the Signing & Capabilities editor.

Add capabilities by clicking the "+" button above.

步驟七、在capability中增加使用iCloud

capability



選擇CloudKit

自訂資料集Containers的名稱

步驟八、在background mode中增加使用remote notification

General | Signing & Capabilities | Resource Tags | Info | Build Settings | Build Phases | Build Rules

ility | All | Debug | Release

gning

iCloud

Push Notifications

backg

Background Modes

Background Modes

Background Modes specifies that the app provides specific background services and must be allowed to continue running while in the background. These keys should be used sparingly and only by apps providing the indicated services. Where alternatives for running in the background exist, those alternatives should be used instead.

General | Signing & Capabilities | Resource Tags | Info | Build Settings | Build Phases | Build Rules

+ Capability | All | Debug | Release

▶ **Signing**

▼ **Background Modes**

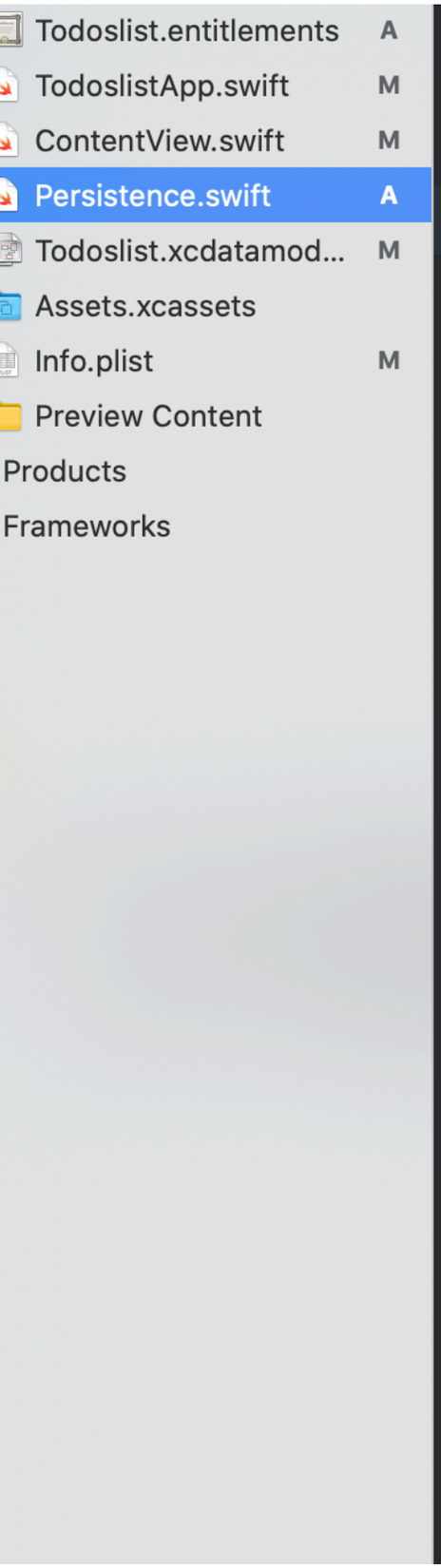
Modes
- [ ] Audio, AirPlay, and Picture in Picture
- [ ] Location updates
- [ ] Voice over IP
- [ ] External accessory communication
- [ ] Uses Bluetooth LE accessories
- [ ] Acts as a Bluetooth LE accessory
- [ ] Background fetch
- [x] Remote notifications
- [ ] Background processing

▼ **iCloud**

Services
- [ ] Key-value storage
- [ ] iCloud Documents
- [x] CloudKit

Containers
- [x] iCloud.HelloCoreDataModel

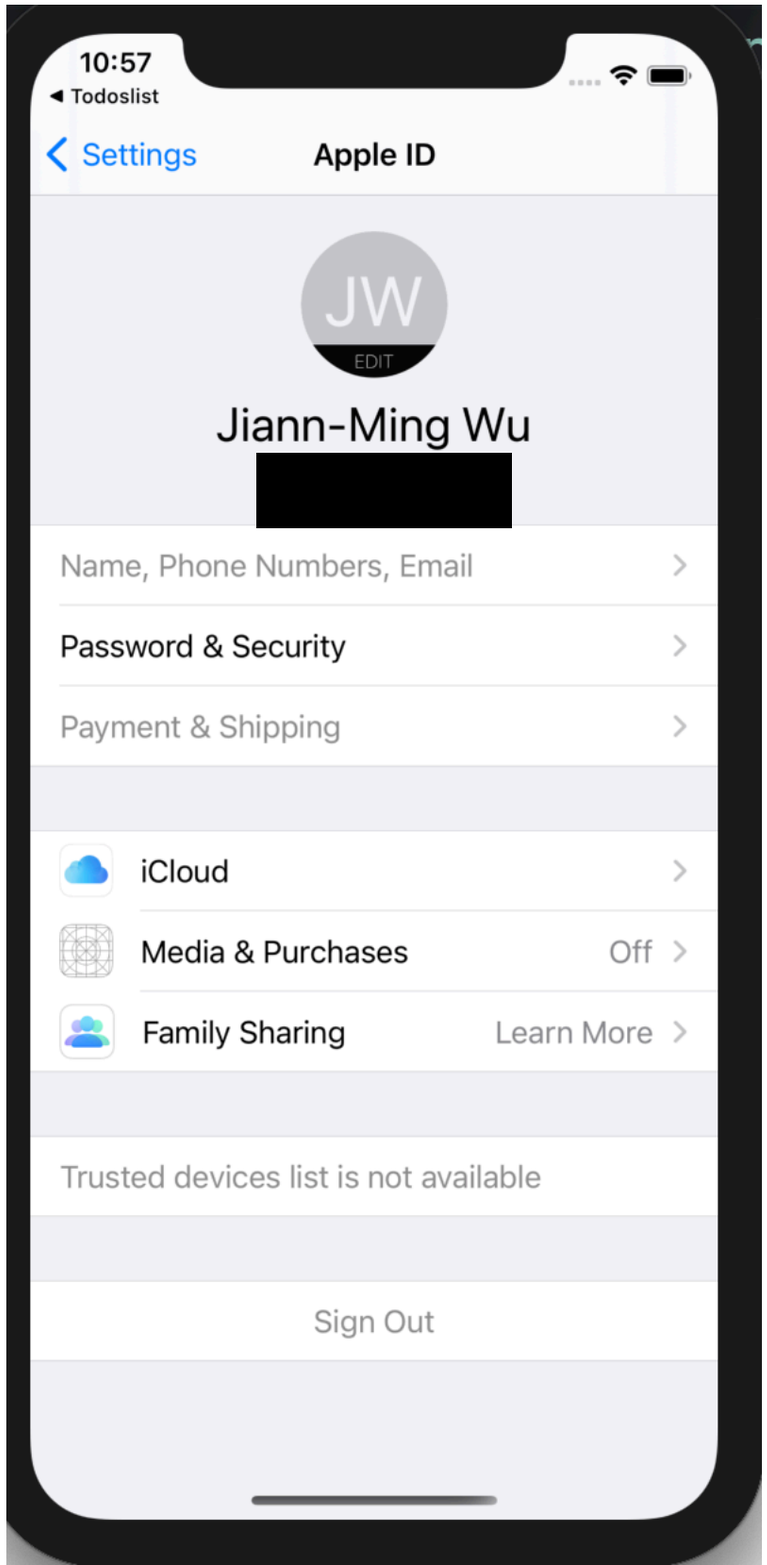
步驟九、修改



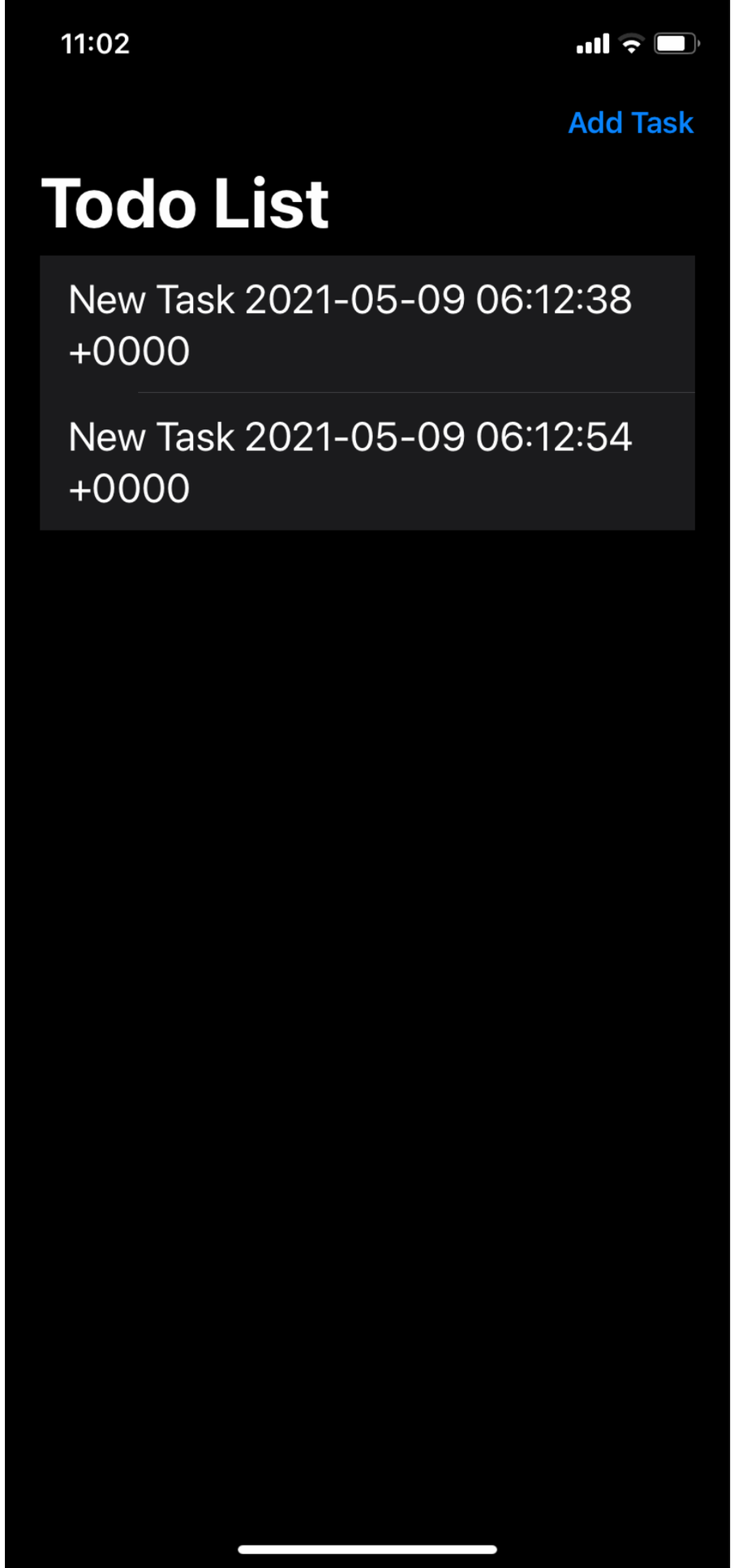

```swift
import CoreData

struct PersistenceController {
    static let shared = PersistenceController()
    let container : NSPersistentContainer
    init() {
        container = NSPersistentCloudKitContainer(name:
            "Todoslist")
        container.loadPersistentStores {
            (storeDescription, error) in
            if let error = error as NSError? {
                fatalError("Unsolved error: \(error)")
            }

        }
        container.viewContext
            .automaticallyMergesChangesFromParent = true
        container.viewContext.mergePolicy =
            NSMergeByPropertyObjectTrumpMergePolicy
    }

}
```

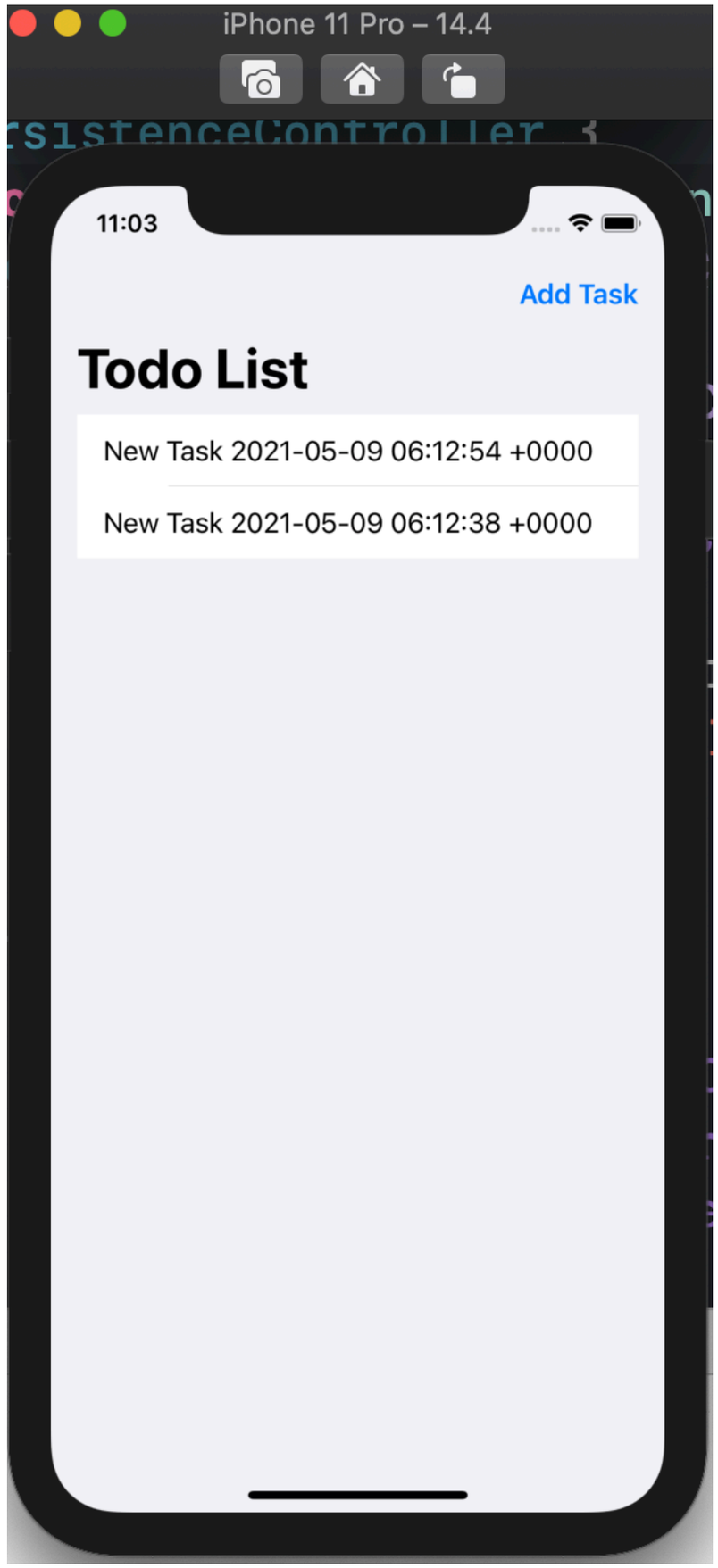
測試步驟A：手機與模擬器都登入iCloud


10:59
設定
搜尋
JW
Jiann-Ming Wu
Apple ID、iCloud、媒體與購買項目

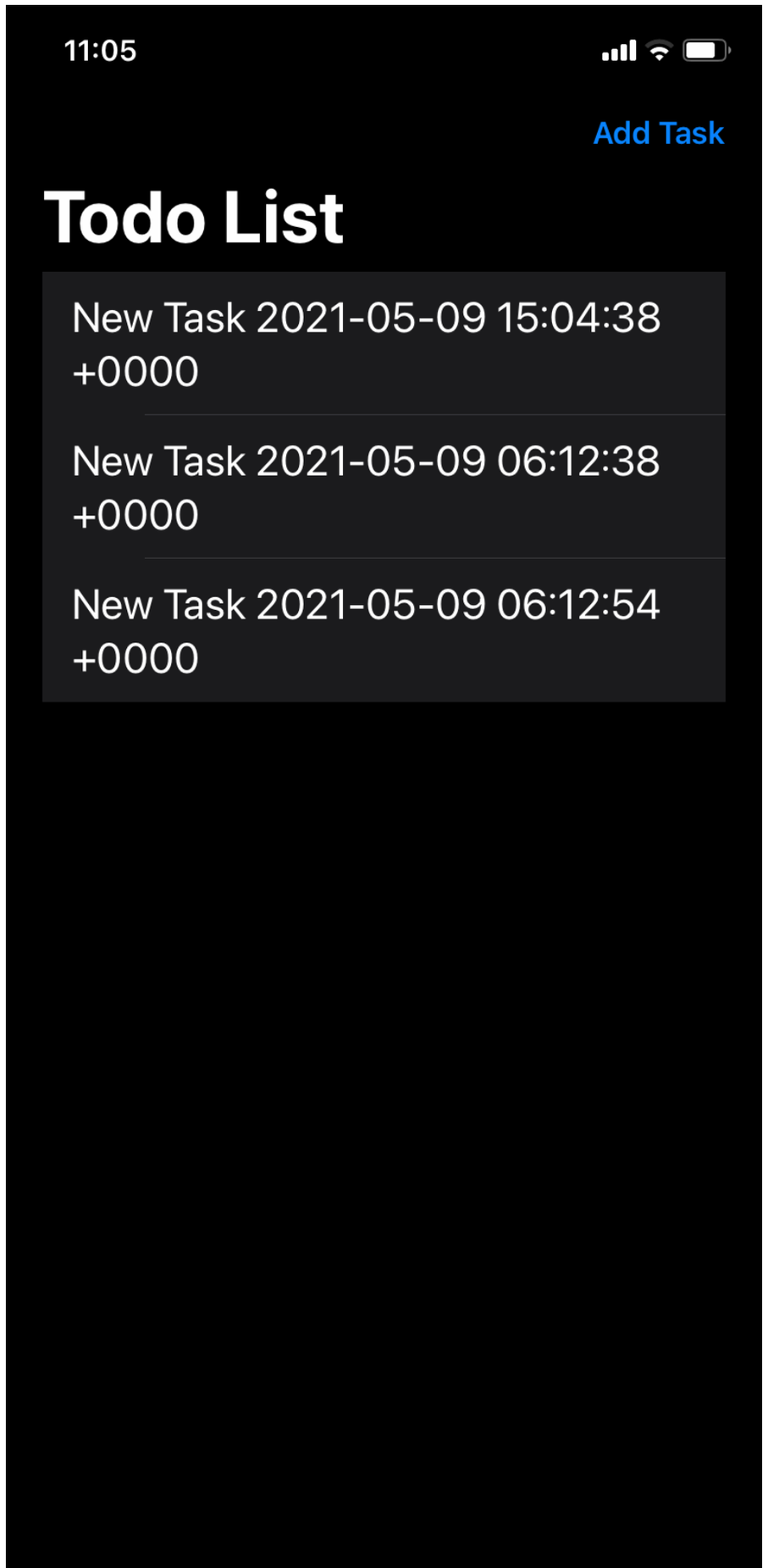

10:57
Todoslist
Settings
Apple ID
JW
EDIT
Jiann-Ming Wu
[redacted]
Name, Phone Numbers, Email
Password & Security
Payment & Shipping
iCloud
Media & Purchases
Off
Family Sharing
Learn More
Trusted devices list is not available
Sign Out

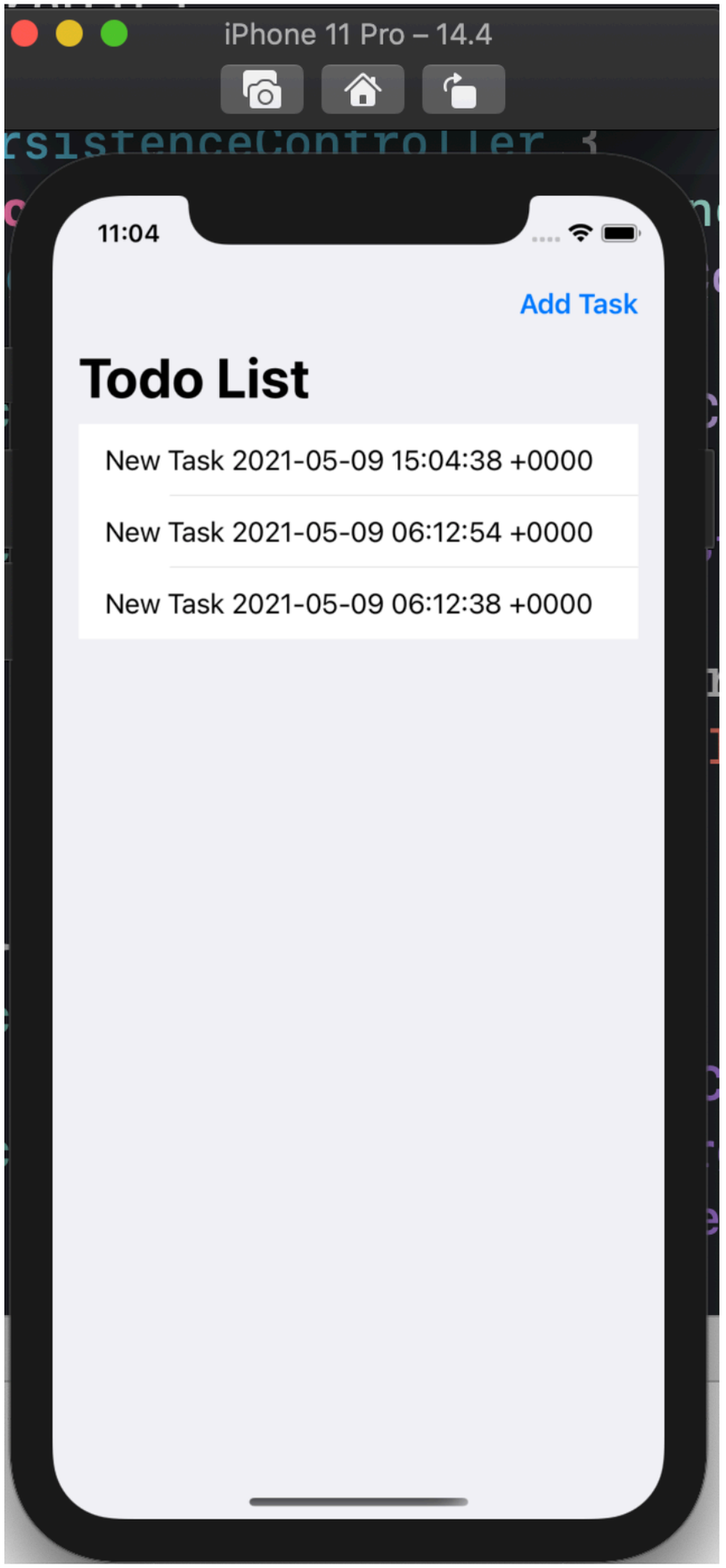
測試步驟B：手機與模擬器都使用Todoslist


11:02
Add Task
Todo List
New Task 2021-05-09 06:12:38 +0000
New Task 2021-05-09 06:12:54 +0000


iPhone 11 Pro – 14.4
11:03
Add Task
Todo List
New Task 2021-05-09 06:12:54 +0000
New Task 2021-05-09 06:12:38 +0000

測試步驟C：模擬器增加資料，手機也會看到新增資料


11:05
Add Task
Todo List
New Task 2021-05-09 15:04:38 +0000
New Task 2021-05-09 06:12:38 +0000
New Task 2021-05-09 06:12:54 +0000


iPhone 11 Pro – 14.4
11:04
Add Task
Todo List
New Task 2021-05-09 15:04:38 +0000
New Task 2021-05-09 06:12:54 +0000
New Task 2021-05-09 06:12:38 +0000