Class 07-1


TAIWAN
TECH
National Taiwan University of
Science and Technology


HPCLab
High-Performance Computing Laboratory

NSUserDefaults

NSUserDefaults

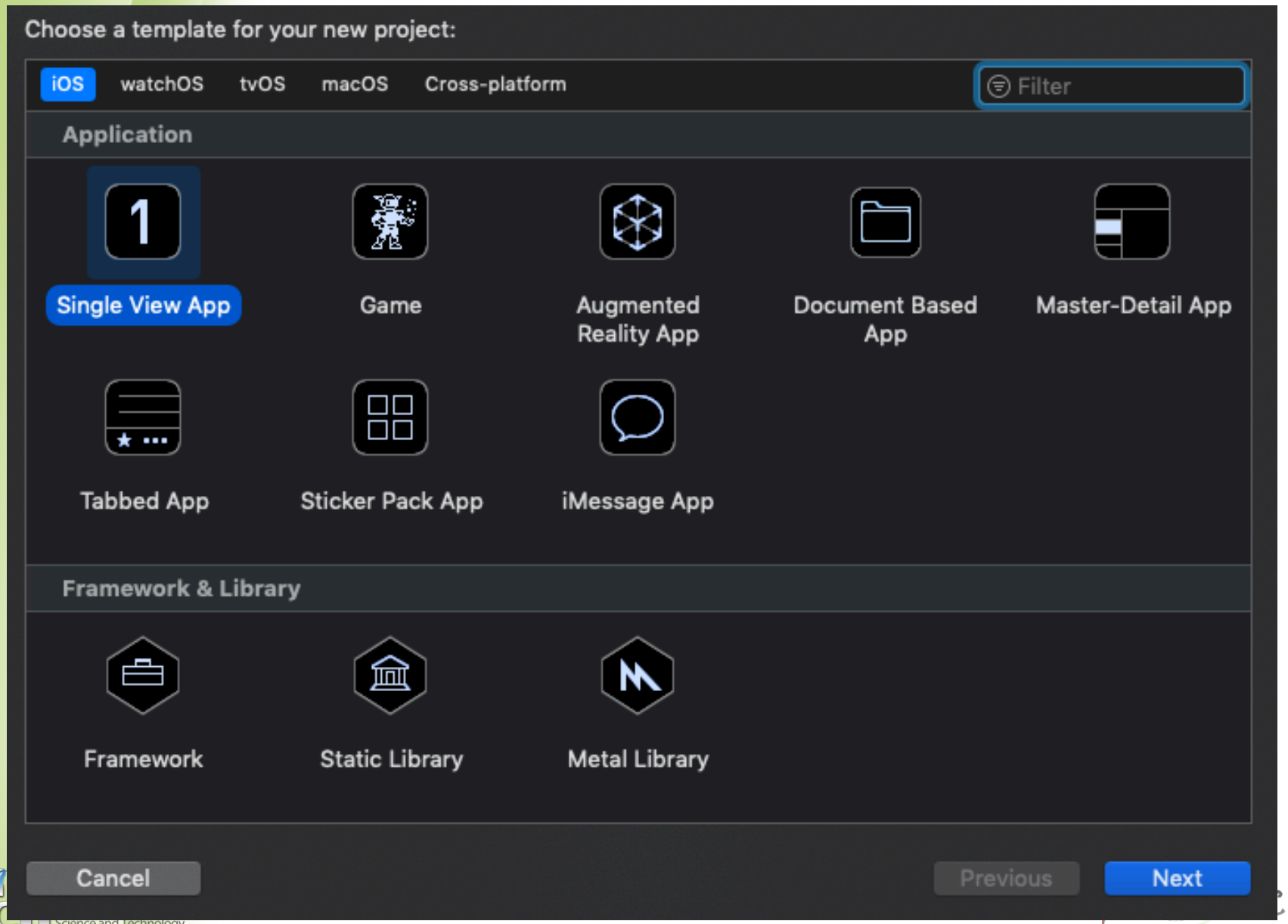
- The NSUserDefaults class provides a programmatic interface for interacting with the defaults system. The defaults system allows an application to customize its behavior to match a user's preferences.

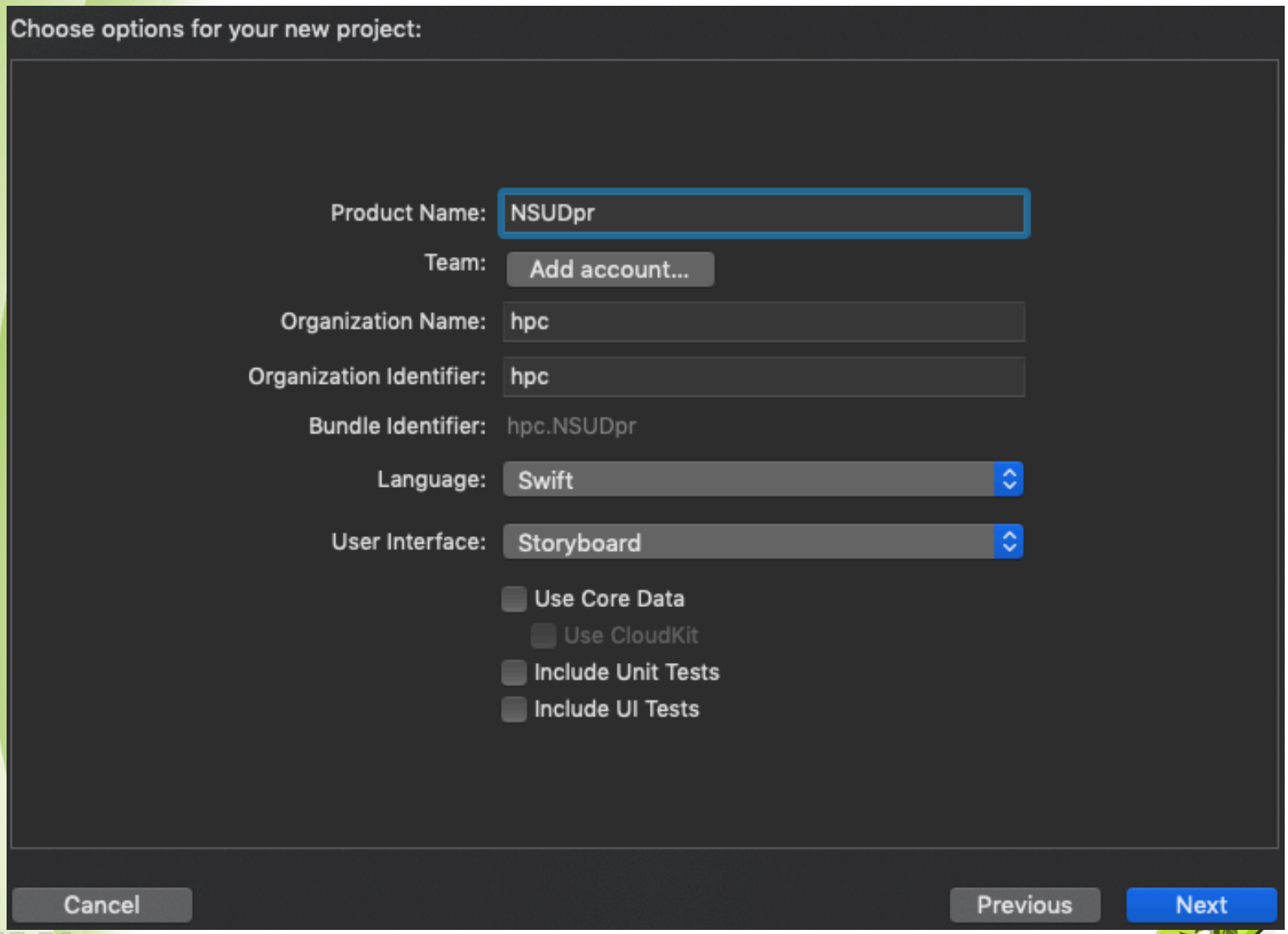
- For example, you can allow users to determine what units of measurement your application displays or how often documents are automatically saved.

- Applications record such preferences by assigning values to a set of parameters in a user's defaults database.

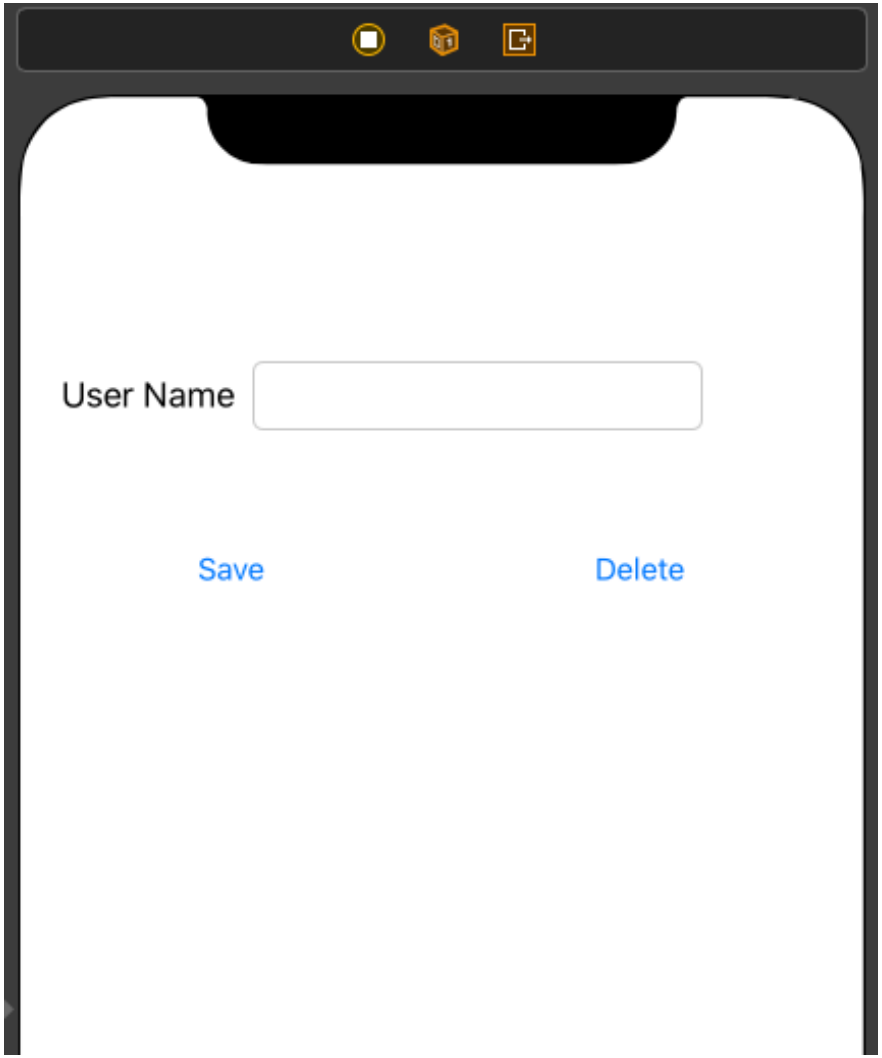
- The parameters are referred to as defaults since they're commonly used to determine an application's default state at startup or the way it acts by default.

Persisting Data with NSUserDefaults




Choose options for your new project:
Product Name:
NSUDpr
Team:
Add account...
Organization Name:
hpc
Organization Identifier:
hpc
Bundle Identifier:
hpc.NSUDpr
Language:
Swift
User Interface:
Storyboard
Use Core Data
Use CloudKit
Include Unit Tests
Include UI Tests
Cancel
Previous
Next

Interface



建立NSUserDeafaults物件

```swift
import UIKit

class ViewController: UIViewController {
    var userDefault : UserDefaults = UserDefaults.standard

    @IBOutlet weak var textName: UITextField!
    @IBAction func saveClick(_ sender: UIButton) {
    }
    @IBAction func deleteClick(_ sender: UIButton) {
    }
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }
}
```

寫入資料

```
@IBAction func saveClick(_ sender: UIButton) {
    userDefault.set(textName.text, forKey: "userName")
    userDefault.synchronize()
}
@IBAction func deleteClick(_ sender: UIButton) {
    userDefault.removeObject(forKey: "userName")
    textName.text = ""
}
```

清除此物件的資料

加入一個alertController

```
15      func alertBtn(_ Title:String, Message:String, BtnTitle:String){
16          let alertController = UIAlertController(title: Title, message: Message,
                preferredStyle: .alert)
17          let action = UIAlertAction(title: BtnTitle, style: .default, handler: nil)
18          alertController.addAction(action)
19          self.present(alertController,animated: true,completion: nil)
20      }
```

viewDidLoad->viewWillAppear->viewDidAppear->viewWillDisappear->viewDidDispappear

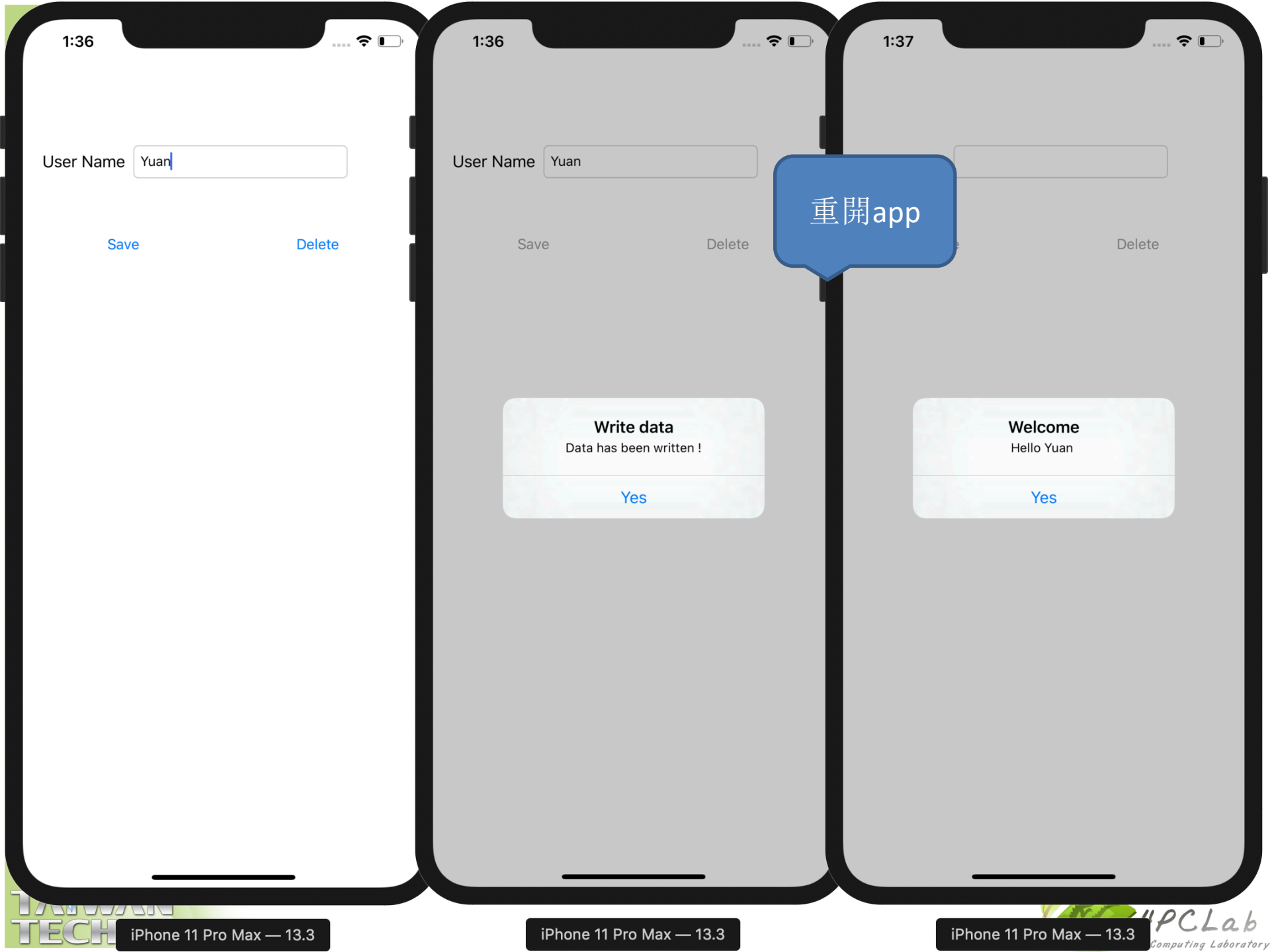
當viewController被通知視圖已加入到view hierarchy時

讀取物件內的資料

```
25      override func viewDidAppear(_ animated: Bool) {
26          let name:String? = userDefault.object(forKey: "userName") as! String?
27          if name == nil  {
28              alertBtn("Enter the name", Message: "Welcome to this app !", BtnTitle:
                    "Yes")
29          }else    {
30              let msg:String = "Hello " + name!
31              alertBtn("Welcome", Message: msg, BtnTitle: "Yes")
32          }
33      }
34  }
```

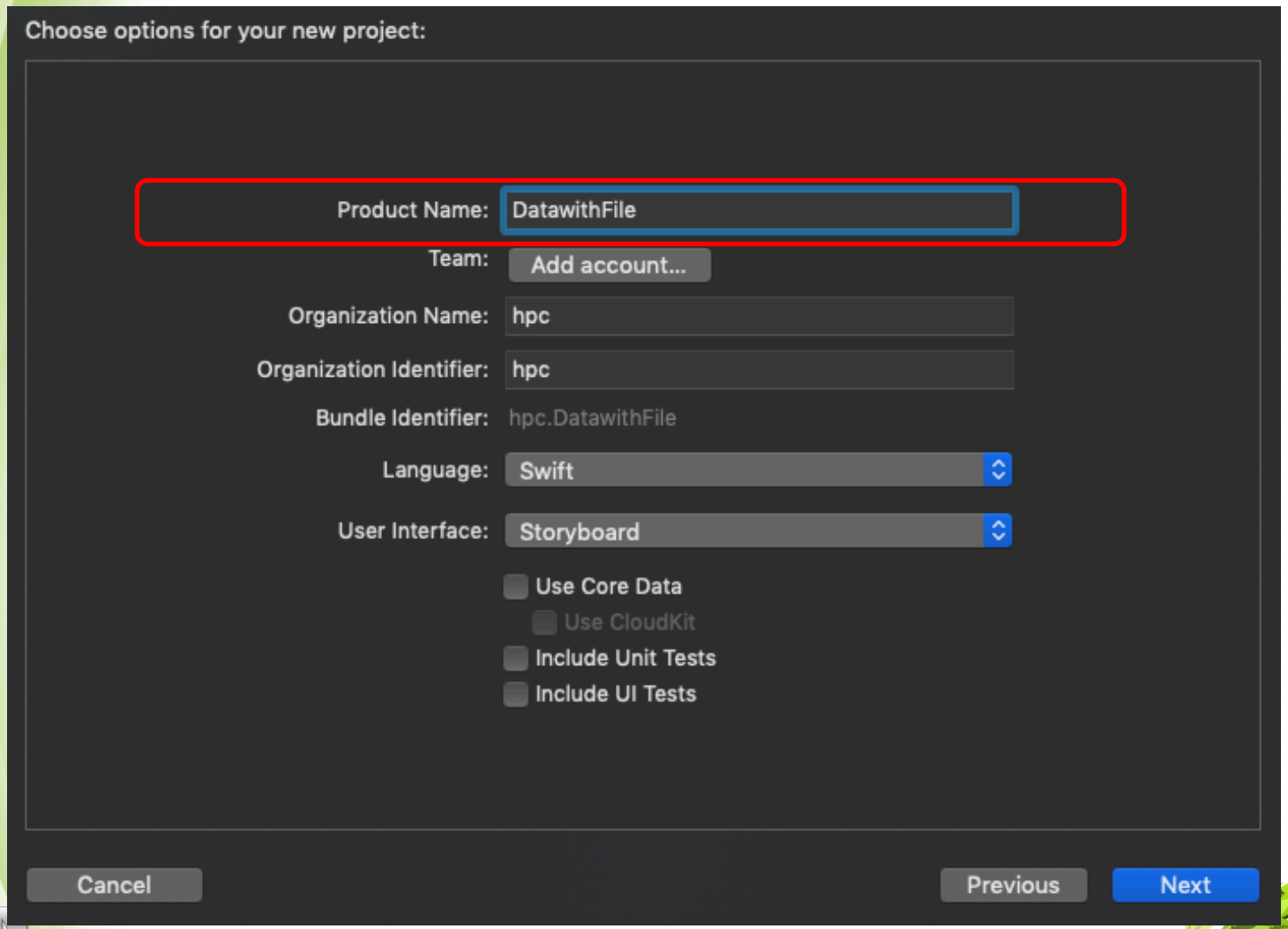
資料已存在

```
    @IBAction func saveClick(_ sender: UIButton) {
        userDefault.set(textName.text, forKey: "userName")
        userDefault.synchronize()
        alertBtn("Write data", Message: "Data has been written !", BtnTitle: "Yes")
    }
    @IBAction func deleteClick(_ sender: UIButton) {
        userDefault.removeObject(forKey: "userName")
        textName.text = ""
        alertBtn("Clear data", Message: "Data has been cleared !", BtnTitle: "Yes")
    }
```

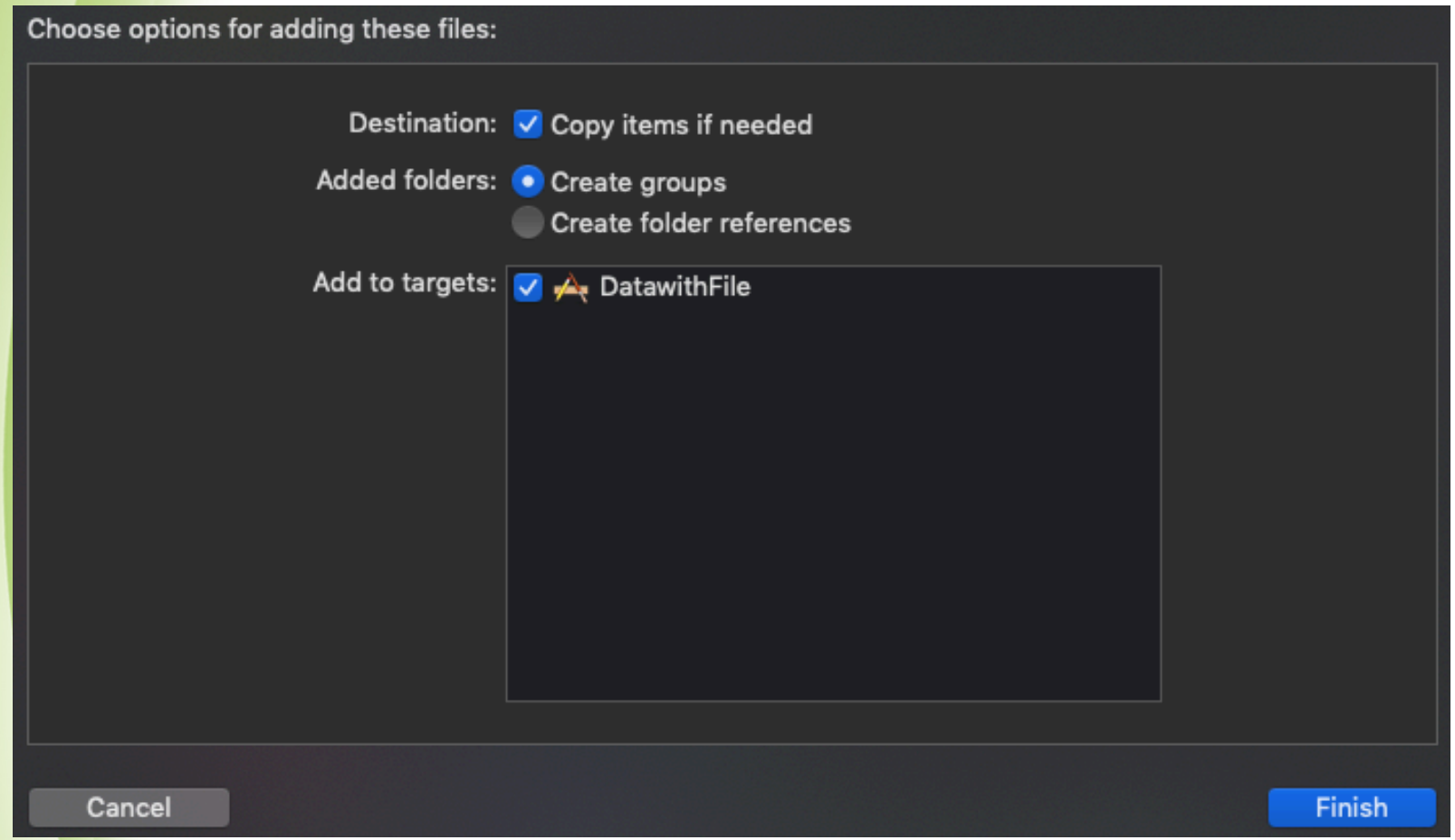

1:36
User Name
Yuan
Save
Delete
iPhone 11 Pro Max — 13.3
1:36
User Name
Yuan
Save
Delete
Write data
Data has been written !
Yes
iPhone 11 Pro Max — 13.3
重開app
1:37
Delete
Welcome
Hello Yuan
Yes
iPhone 11 Pro Max — 13.3

Data with file

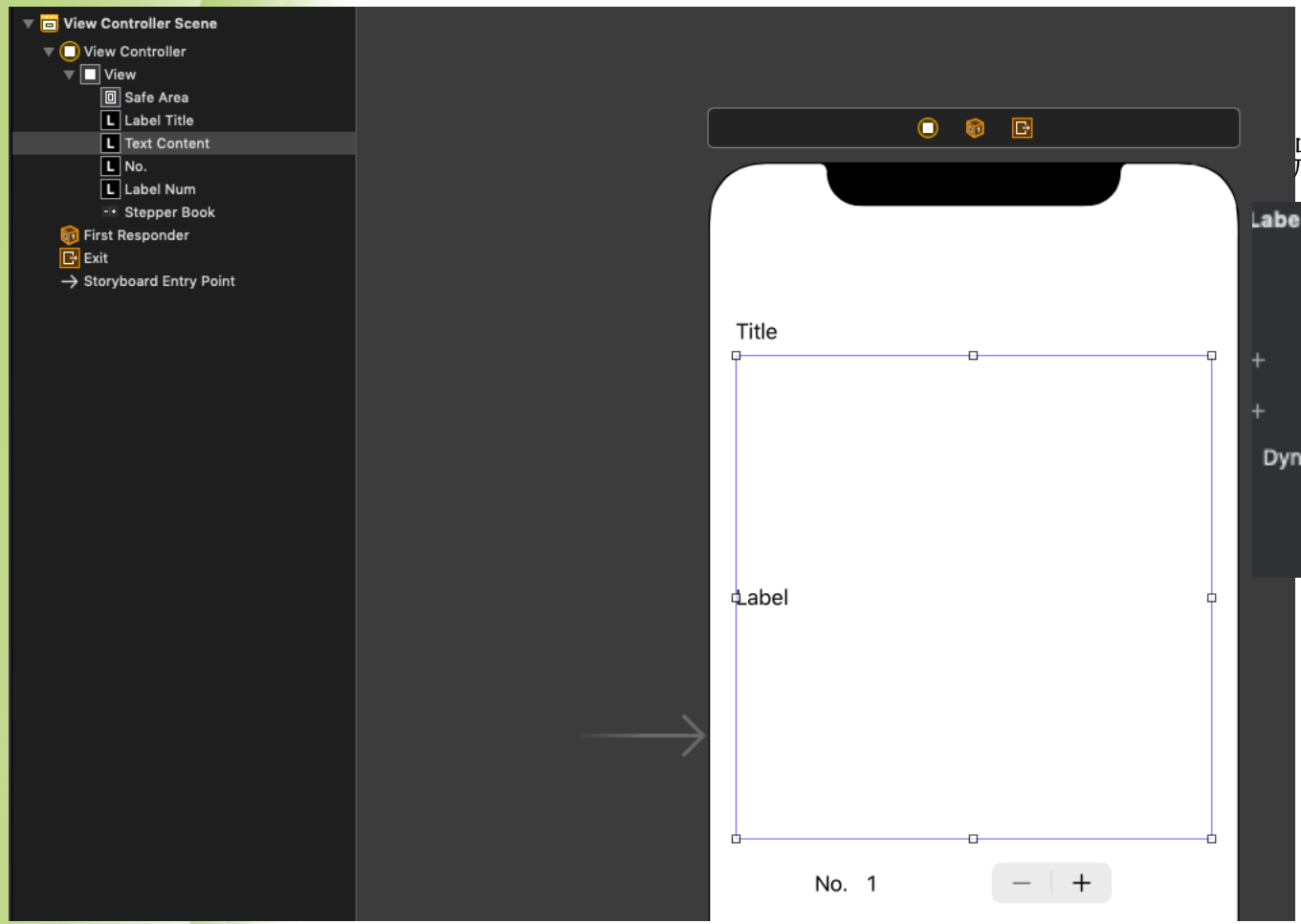
Create new project



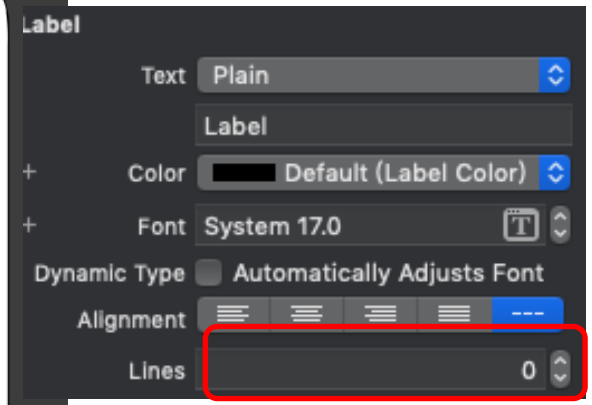
將兩個txt檔放入專案



修改至如圖所示
4個label
1個stepper



將內容label's line 改為 0



```swift
import UIKit

class ViewController: UIViewController {
    var arrayTitle:Array<String> = []
    var arrayContent:Array<String> = []

    @IBOutlet weak var labelTitle: UILabel!
    @IBOutlet weak var textContent: UILabel!
    @IBOutlet weak var labelNum: UILabel!
    @IBOutlet weak var stepperBook: UIStepper!
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }
}
```

宣告兩個陣列
存標題與內容

對應的做連接

取得01.txt檔在專案中的路徑

取得檔案內的內容

以分行號分解字串

將內容轉換為字串

```
25      override func viewDidLoad() {
26          super.viewDidLoad()
2           // Do any additional setup after loading the view.
2           var path = Bundle.main.path(forResource: "01", ofType: "txt")! as NSString
2           let imgName = try? Data(contentsOf: URL(fileURLWithPath: path as String))
3           let imgString = NSString(data: imgName!, encoding: String.Encoding.utf8.rawValue)! as String
3           let arrayImage = imgString.components(separatedBy: "\n")
3
33          for i in 0...arrayImage.count-2   {
34              let arrayTemp = arrayImage[i].components(separatedBy: ";")
35              arrayTitle.append(arrayTemp[1])
36          }
```

分解字串後，依序加入到title陣列中

For loop 之後

```
37
38          path = Bundle.main.path(forResource: "02", ofType: "txt")! as NSString
39          let contentData = try? Data(contentsOf: URL(fileURLWithPath: path as String))
40          let contentString :String = NSString(data: contentData! as Data, encoding:
                String.Encoding.utf8.rawValue)! as String
41          arrayContent = contentString.components(separatedBy: "\n")
42
43          labelTitle.text = arrayTitle[0]
44          textContent.text = arrayContent[0]
45
46          stepperBook.minimumValue = 0
47          stepperBook.maximumValue = Double(arrayTitle.count) + 1
48          stepperBook.value = 1
49      }
50  }
```

顯示第一筆title與內容

設定stepper元件的初始值

```
@IBAction func stepperChange(_ sender: UIStepper) {
    var num = Int(stepperBook.value)
    if num == arrayTitle.count + 1  {
        num = 1
        stepperBook.value = 1
    }
    else if num == 0{
        num = arrayTitle.count
        stepperBook.value = Double(arrayTitle.count)
    }
    labelTitle.text = arrayTitle[num-1]
    textContent.text = arrayContent[num-1]
    labelNum.text = String(num)
}
```

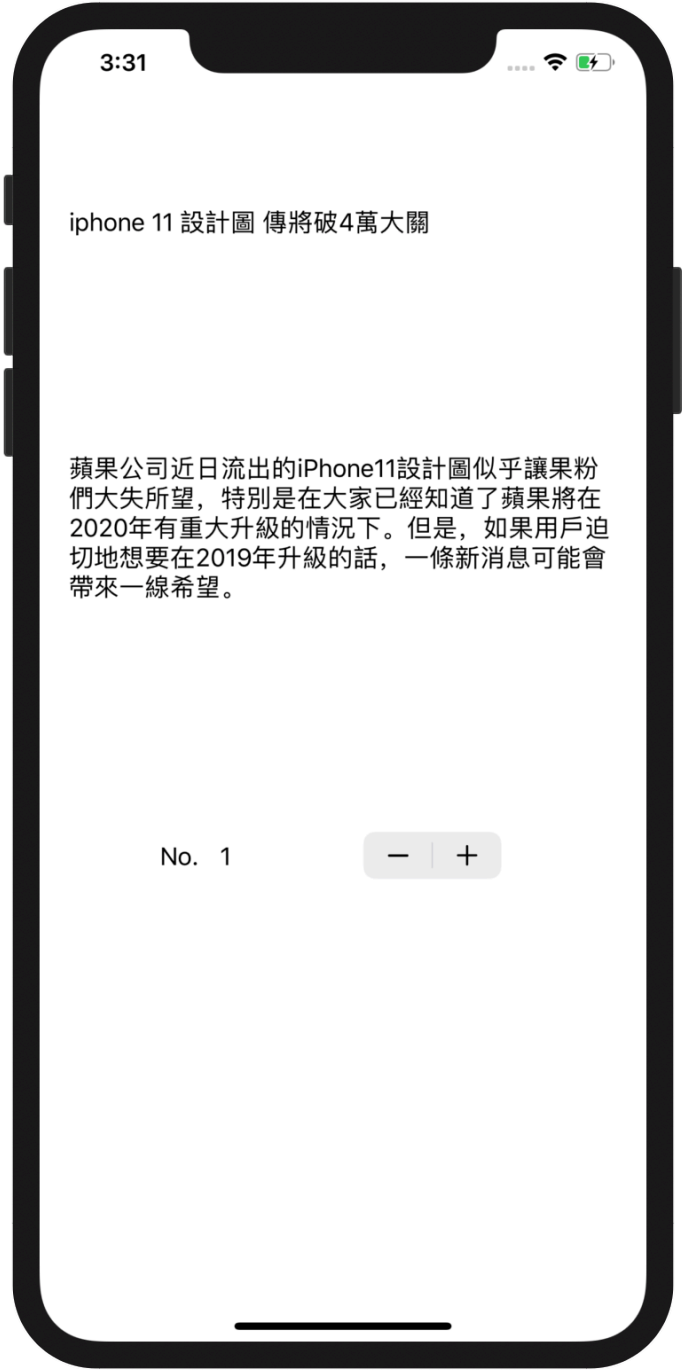
若超過最後一筆則返回第一筆

若到第0筆則移到最後一筆

取得stepper value後用來換頁

顯示對應的title與內容

Run it



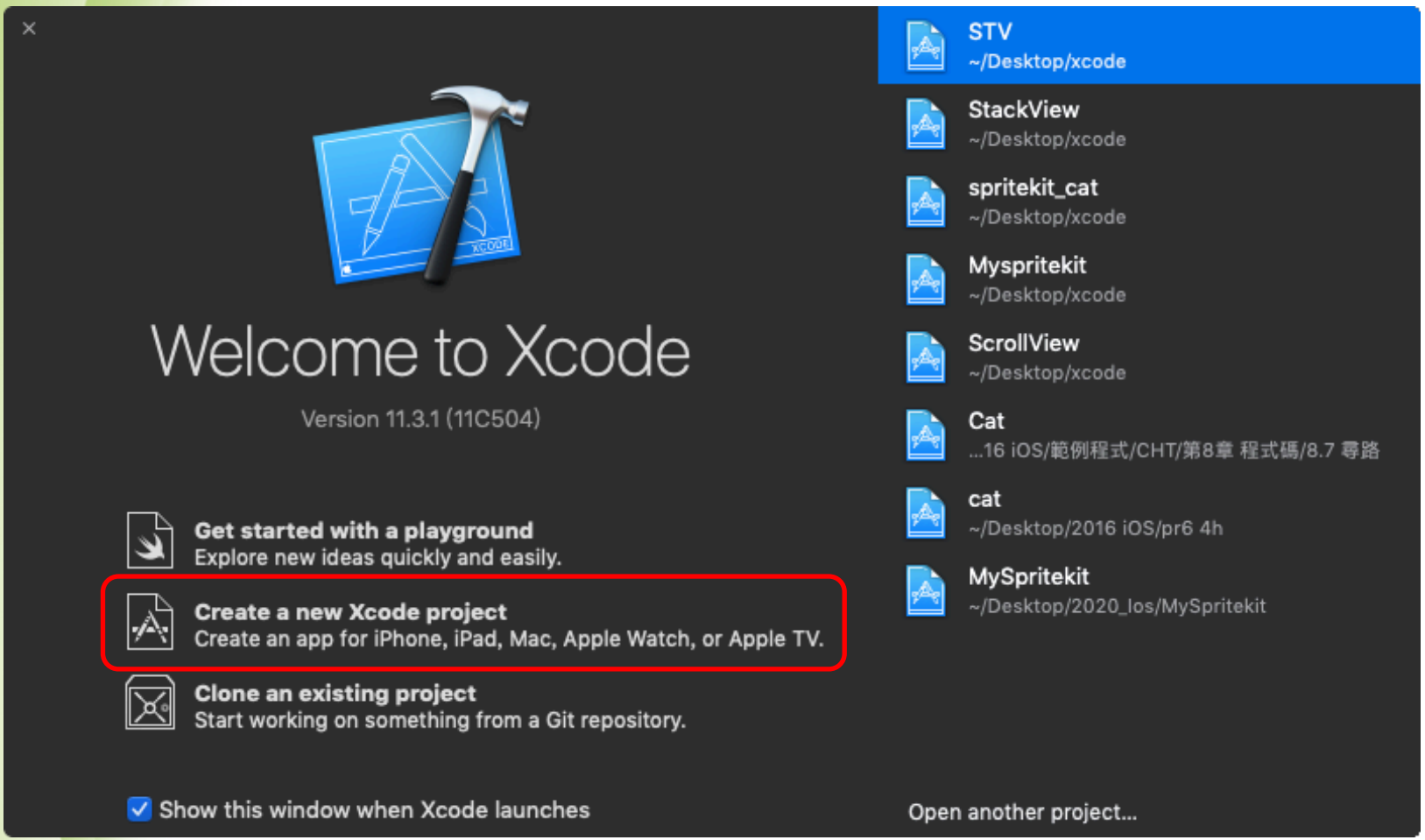
iPhone 11 Pro Max — 13.3

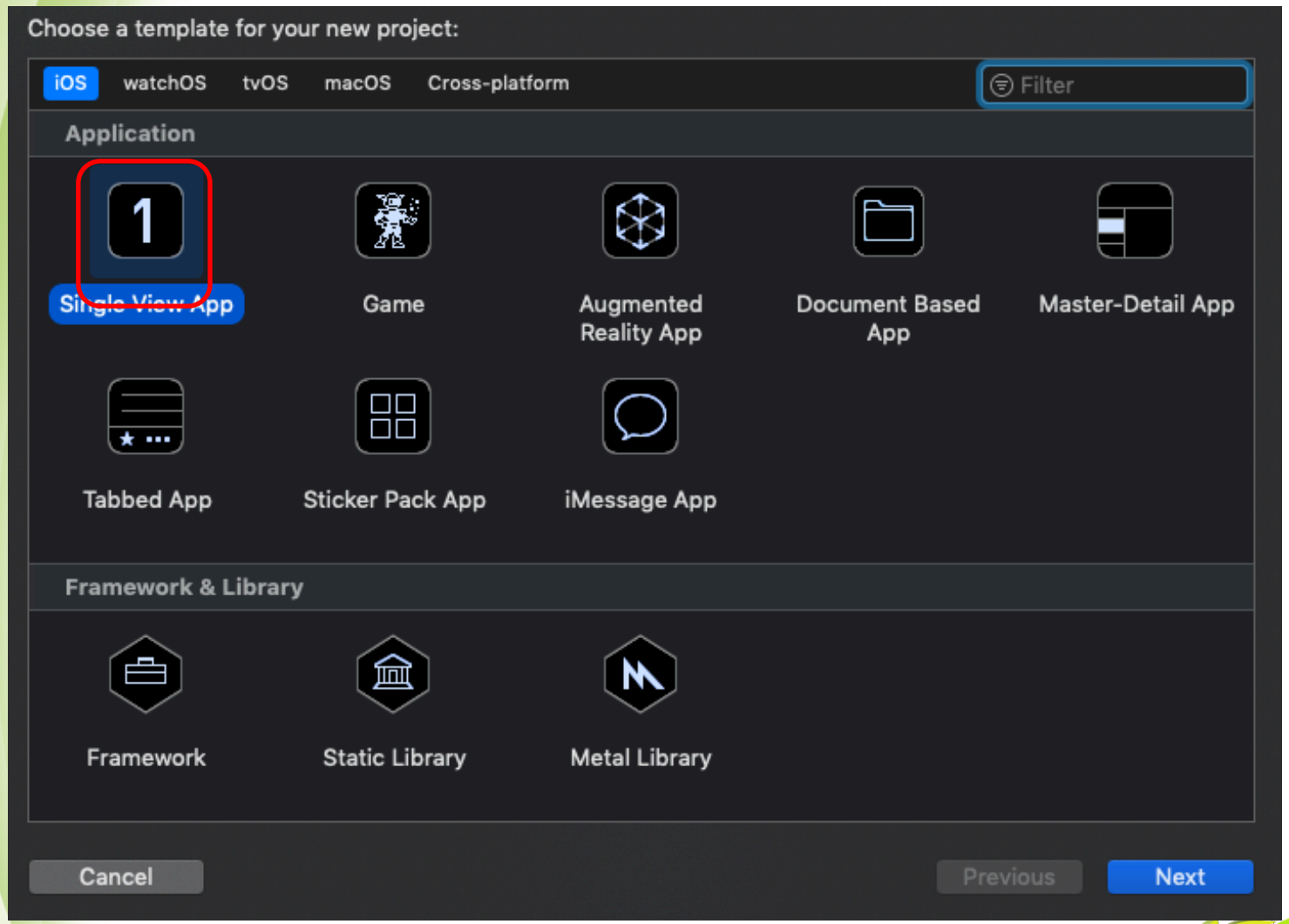
結構化資料存取

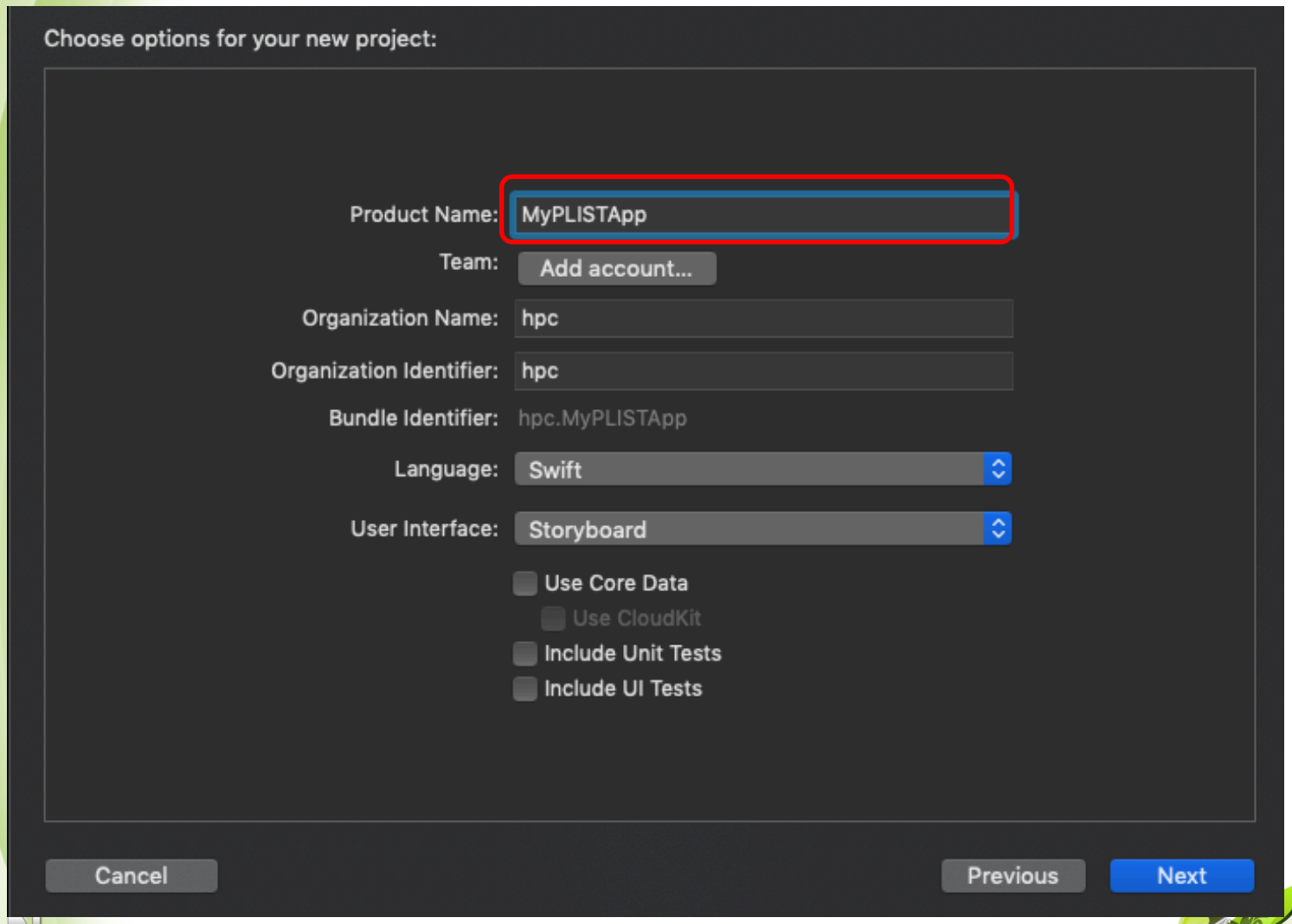
- PLIST檔
  - XML格式 <key>屬性名稱 <string>屬性值
  - 適合有"配對"特性而且資料量不大的資料
- SQLite
  - 標準資料庫系統
  - SQL Command
  - 適合熟悉資料庫操作的開發者
- Core Data
  - 將資料庫特性包裝
  - 不需學SQL Command

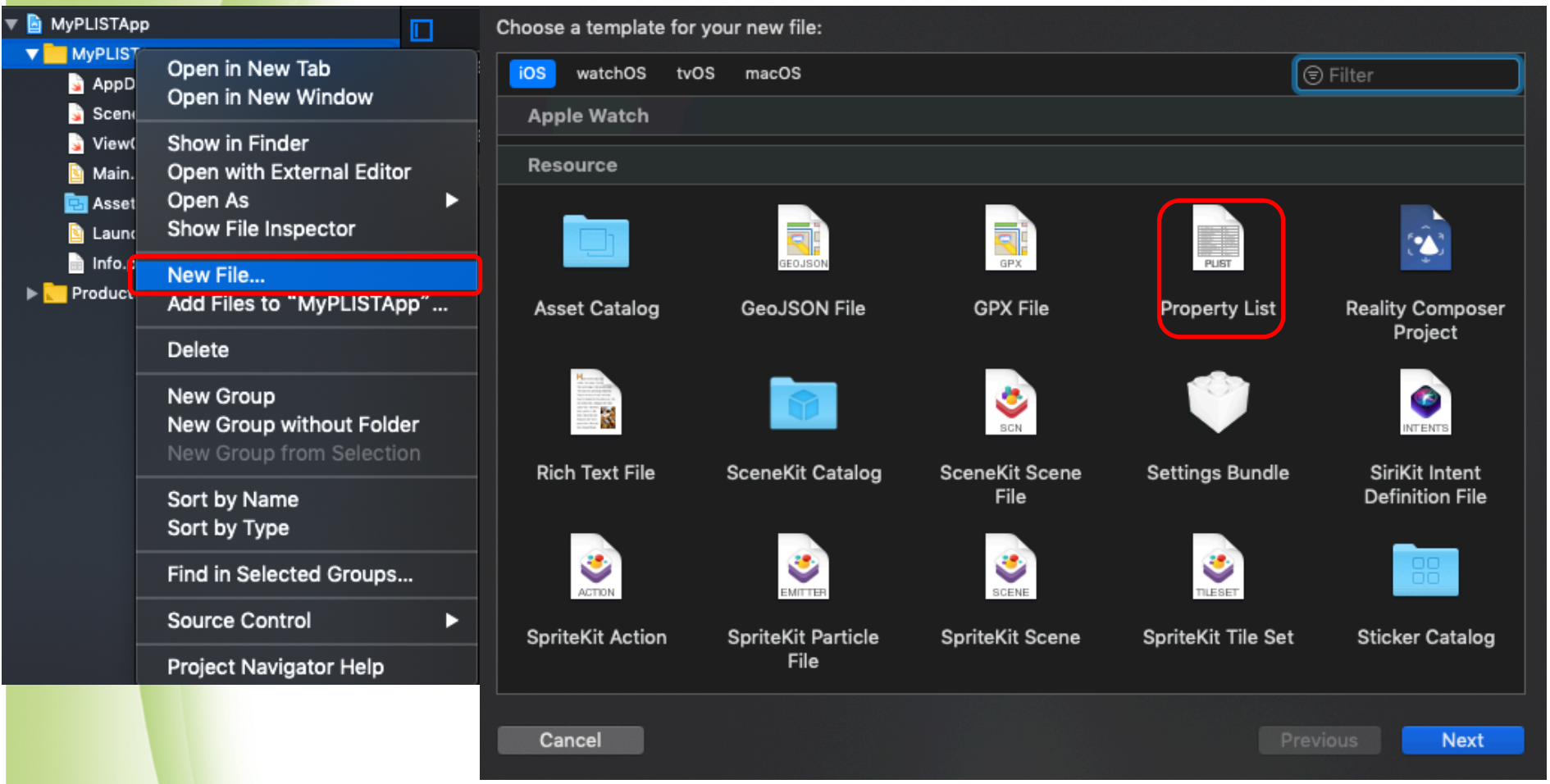
PLIST

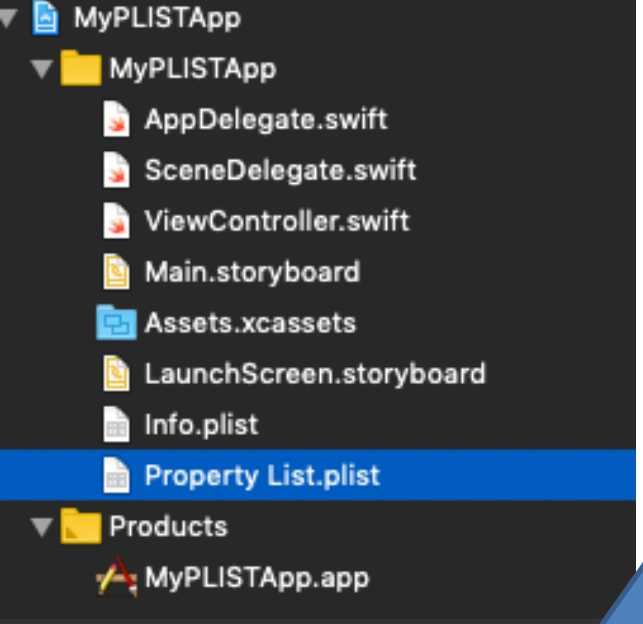
Property List


Welcome to Xcode
Version 11.3.1 (11C504)
Get started with a playground
Explore new ideas quickly and easily.
Create a new Xcode project
Create an app for iPhone, iPad, Mac, Apple Watch, or Apple TV.
Clone an existing project
Start working on something from a Git repository.
Show this window when Xcode launches
STV
~/Desktop/xcode
StackView
~/Desktop/xcode
spritekit_cat
~/Desktop/xcode
Myspritekit
~/Desktop/xcode
ScrollView
~/Desktop/xcode
Cat
...16 iOS/範例程式/CHT/第8章 程式碼/8.7 尋路
cat
~/Desktop/2016 iOS/pr6 4h
MySpritekit
~/Desktop/2020_Ios/MySpritekit
Open another project...



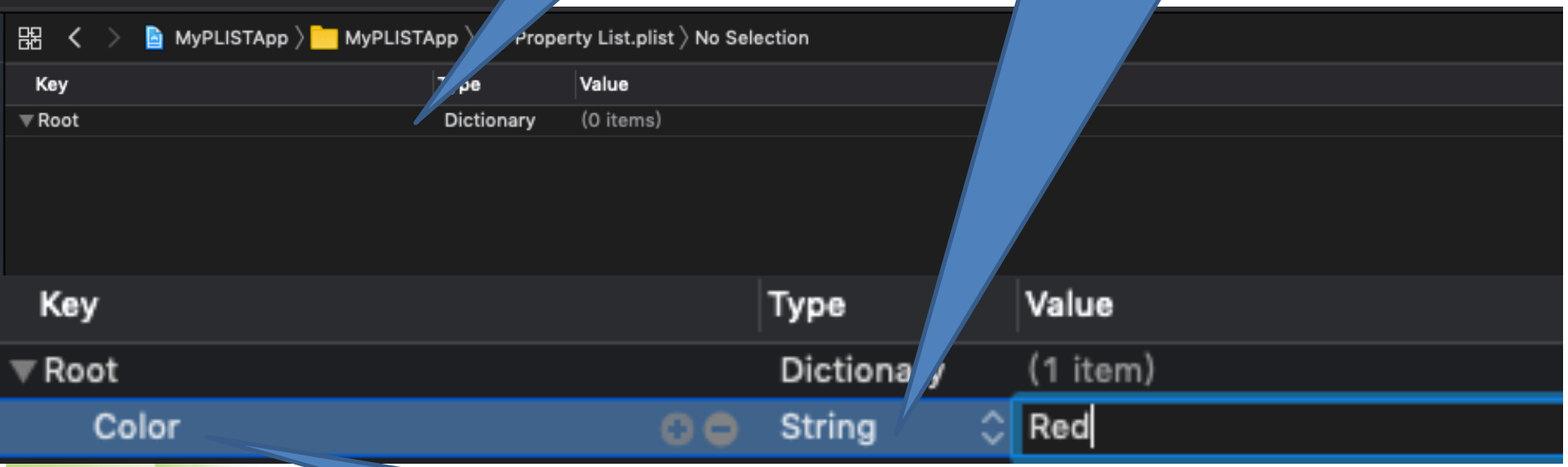

Choose options for your new project:
Product Name:
MyPLISTApp
Team:
Add account...
Organization Name:
hpc
Organization Identifier:
hpc
Bundle Identifier:
hpc.MyPLISTApp
Language:
Swift
User Interface:
Storyboard
Use Core Data
Use CloudKit
Include Unit Tests
Include UI Tests
Cancel
Previous
Next


MyPLISTApp
Open in New Tab
Open in New Window
Show in Finder
Open with External Editor
Open As
Show File Inspector
New File...
Add Files to "MyPLISTApp"...
Delete
New Group
New Group without Folder
New Group from Selection
Sort by Name
Sort by Type
Find in Selected Groups...
Source Control
Project Navigator Help
Choose a template for your new file:
iOS
watchOS
tvOS
macOS
Filter
Apple Watch
Resource
Asset Catalog
GeoJSON File
GPX File
Property List
Reality Composer Project
Rich Text File
SceneKit Catalog
SceneKit Scene File
Settings Bundle
SiriKit Intent Definition File
SpriteKit Action
SpriteKit Particle File
SpriteKit Scene
SpriteKit Tile Set
Sticker Catalog
Cancel
Previous
Next



Click “+” 來加入資料

Type可自行選定
有布林值、字串、陣列等形式



我們將搜尋Key來取得他對應的value

- Apple 不允許在app執行時寫入plist檔
- 只有三個資料夾具備寫入權限
  - Documents
  - Library
  - Tmp
- 若有資料要寫入.plist, 必須在App執行後將.plist移到有寫入權限的資料夾, 大部分放入Documents

AppDelegate.swift

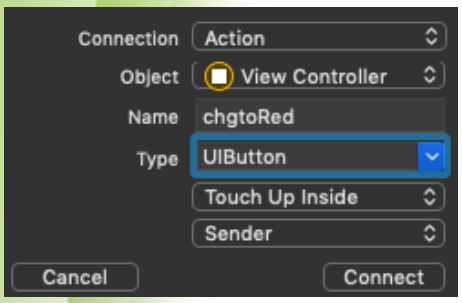
```swift
class AppDelegate: UIResponder, UIApplicationDelegate {



    func application(_ application: UIApplication, didFinishLaunchingWithOptions
        launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        // Override point for customization after application launch.
        let fm = FileManager.default
        let src = Bundle.main.path(forResource: "Property List", ofType: "plist")
        let dst = NSHomeDirectory() + "/Documents/Property List.plist"

        if !fm.fileExists(atPath: dst)  {
            try! fm.copyItem(atPath: src!, toPath: dst)
        }
        return true
    }
```

取得我們的plist檔在專案中的路徑

檢查目的路徑內是否有plist檔存在
如果不存在，則copy到目的路徑中，若已存在確保不會被覆蓋掉

dst為要複製到的目的路徑

修改至如圖所示
兩個button
一個label



Button Connection
為Action

ViewController.swift

```swift
import UIKit

class ViewController: UIViewController {

    var path:String = ""
    @IBOutlet weak var chgLabel: UILabel!
```

```swift
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
        path = NSHomeDirectory() + "/Documents/Property List.plist"
        if let plist = NSMutableDictionary(contentsOfFile: path)    {
            if let color = plist["Color"]   {
                chgLabel.text = "The color is \(color)"
            }
        }
    }
}
```

將plist中key為“Color”的value讀出

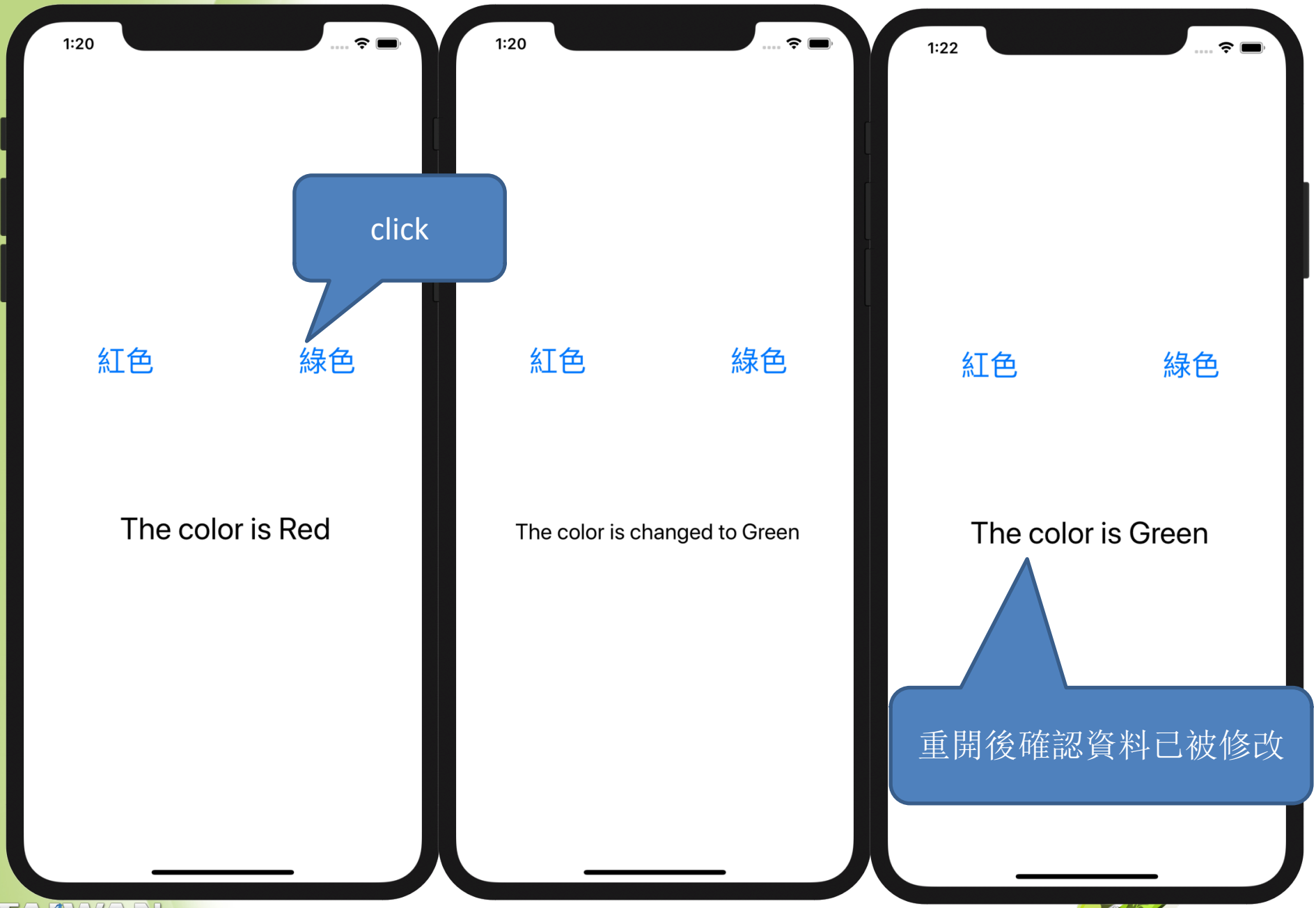
atomically: true 表示ios會先將資料寫入一個暫存檔,再將暫存檔改為真正的檔案
可防止寫入資料時系統crash導致資料遺失

根據path找到鍵值對

ViewController.swift

修改value為
Red

```
    @IBAction func chgtoRed(_ sender: UIButton) {
        if let plist = NSMutableDictionary(contentsOfFile: path)    {
            plist["Color"] = "Red"
            if let color = plist["Color"]   {
                if plist.write(toFile: path, atomically: true)  {
                    chgLabel.text = "The color is changed to \(color)"
                }
            }
        }
    }
    @IBAction func chgtoGreen(_ sender: UIButton) {
        if let plist = NSMutableDictionary(contentsOfFile: path)    {
            plist["Color"] = "Green"
            if let color = plist["Color"]    {
                if plist.write(toFile: path, atomically: true)  {
                    chgLabel.text = "The color is changed to \(color)"
                }
            }
        }
    }
```


1:20
click
紅色
綠色
The color is Red
iPhone 11 Pro Max — 13.3
1:20
紅色
綠色
The color is changed to Green
iPhone 11 Pro Max — 13.3
1:22
紅色
綠色
The color is Green
重開後確認資料已被修改
iPhone 11 Pro Max — 13.3

Core Data
設計與規劃

Core Data design

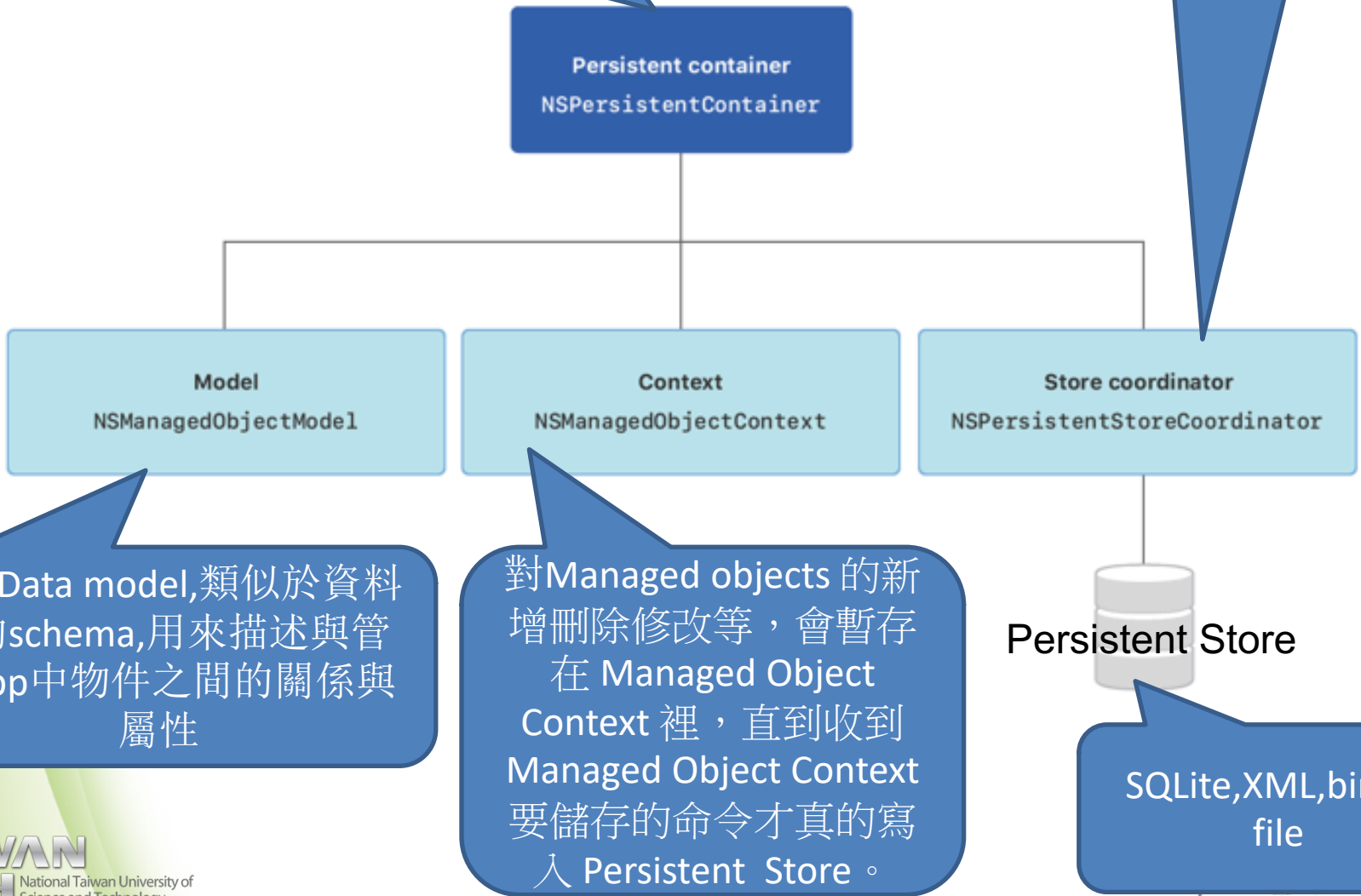
Core Data 使用iOS內建的SQLite,但其簡化資料庫處理,使用者不用懂SQL指令仍可存取資料。

當使用Core Data 存取資料時，必須要先設計資料存放的方式

- 所有資料是存放在實體(entity)之中，實體概念大約等於資料庫中的資料表(table),我們須先將entity之間的關係設計出來

透過對下列三者的創建，來簡化CoreData Stack的建立與管理

用來協調與處理多個PS的存取,當我們創建後,它將會自己做該做的事,無須理會

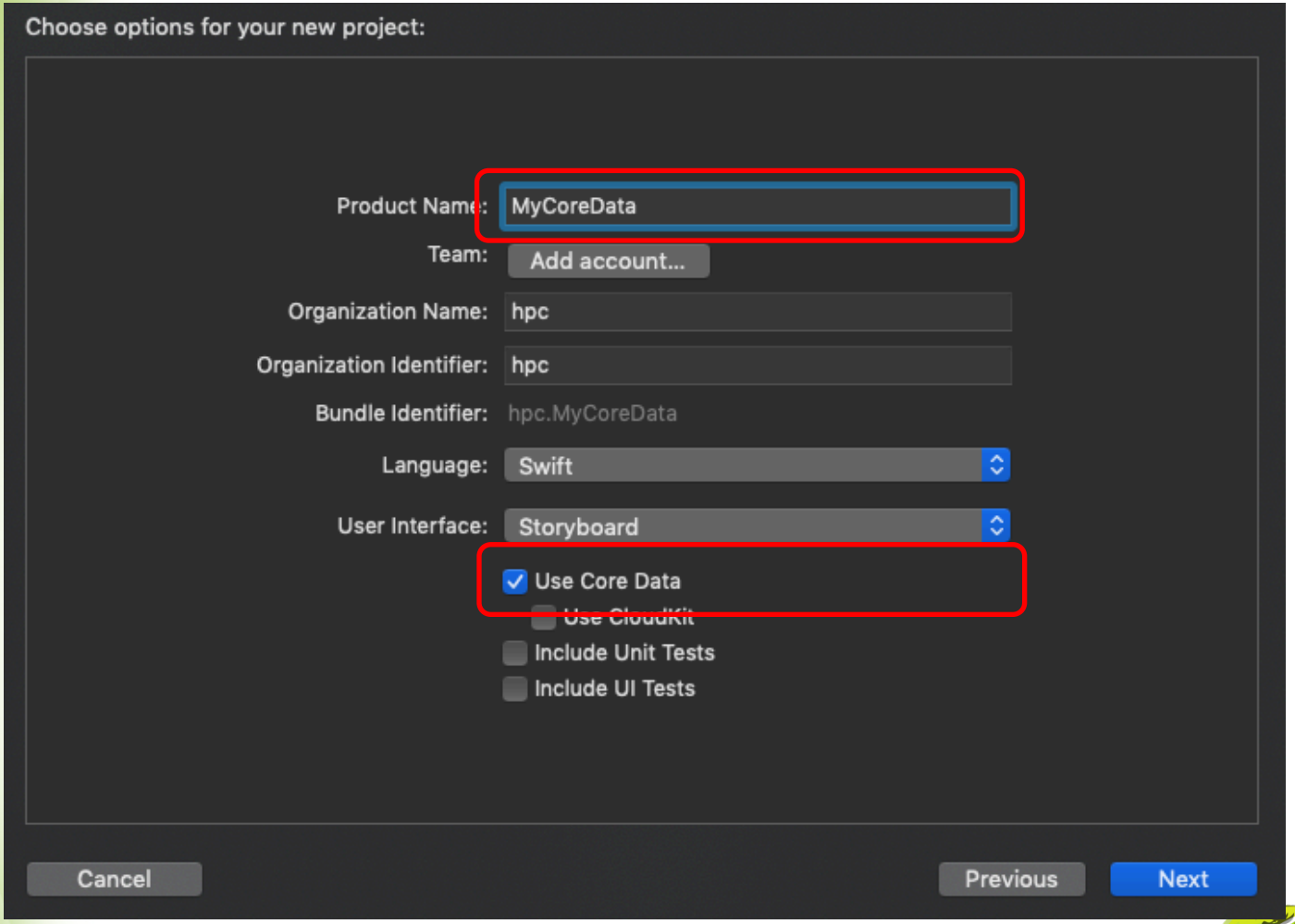


又稱Data model,類似於資料庫的schema,用來描述與管理App中物件之間的關係與屬性

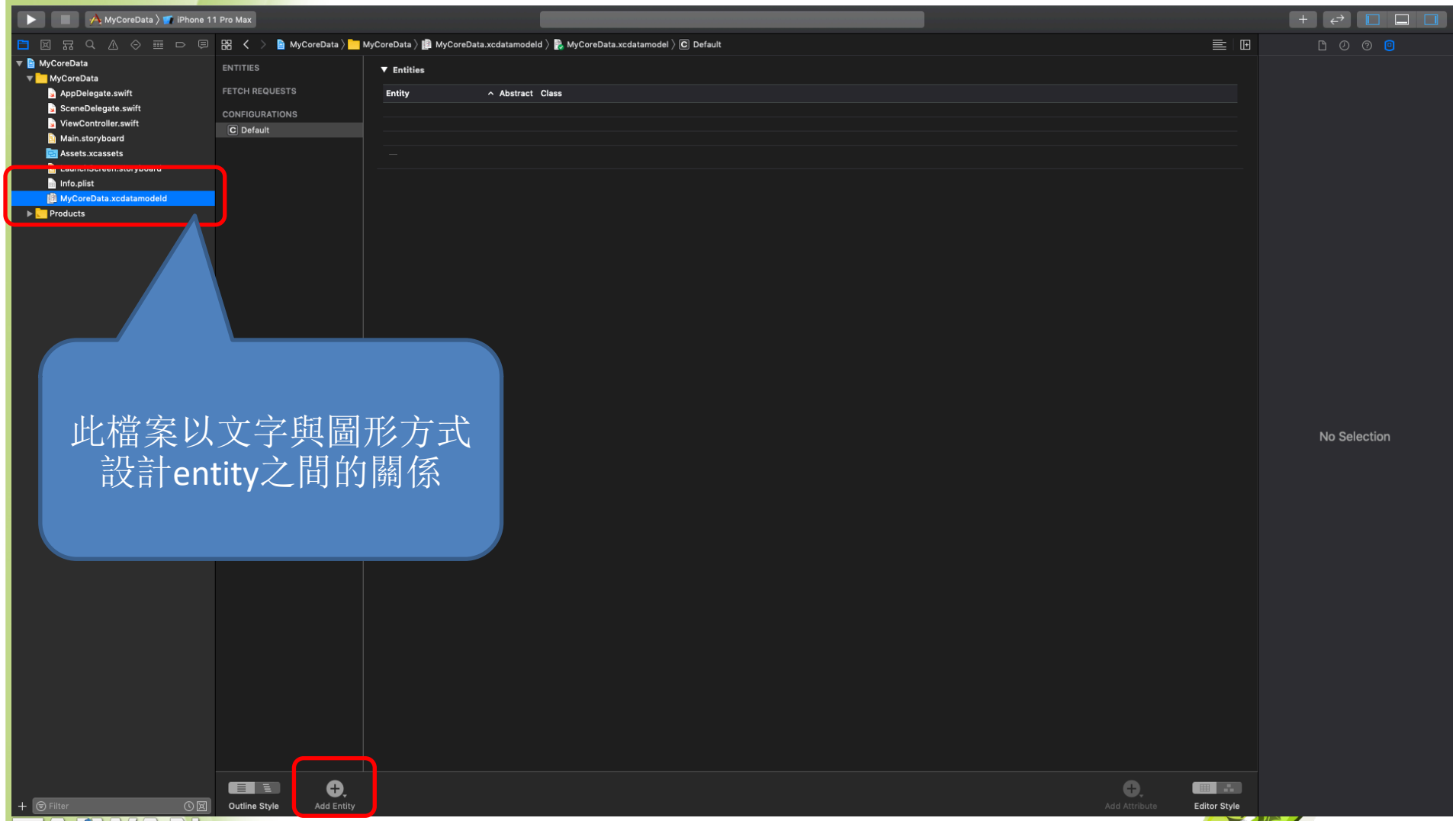
對Managed objects 的新增刪除修改等，會暫存在 Managed Object Context 裡，直到收到 Managed Object Context 要儲存的命令才真的寫入 Persistent Store。

SQLite,XML,binary file

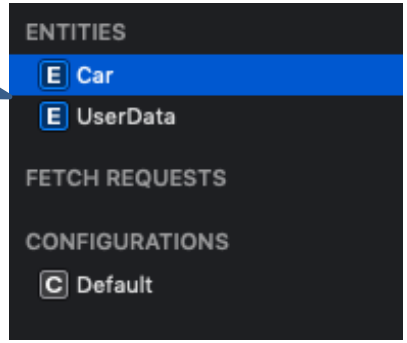
New file


Choose options for your new project:
Product Name:
MyCoreData
Team:
Add account...
Organization Name:
hpc
Organization Identifier:
hpc
Bundle Identifier:
hpc.MyCoreData
Language:
Swift
User Interface:
Storyboard
Use Core Data
Include Unit Tests
Include UI Tests
Cancel
Previous
Next

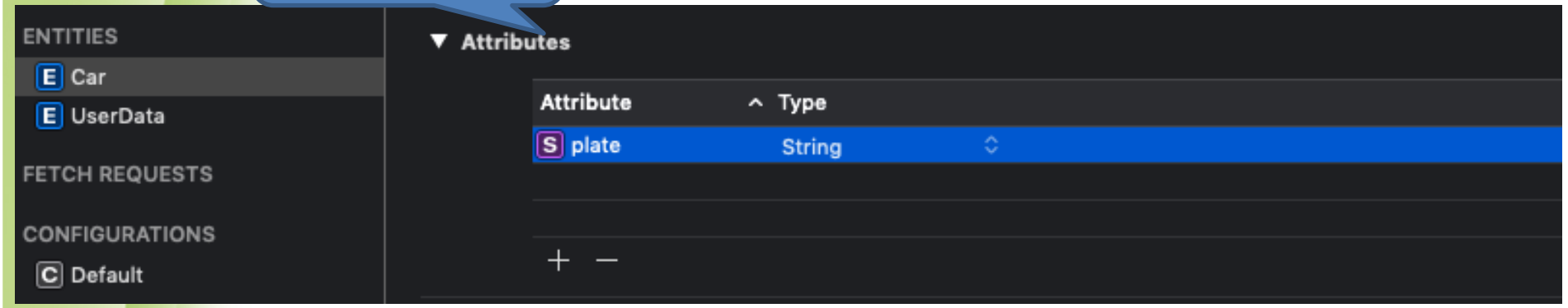
建立Data model

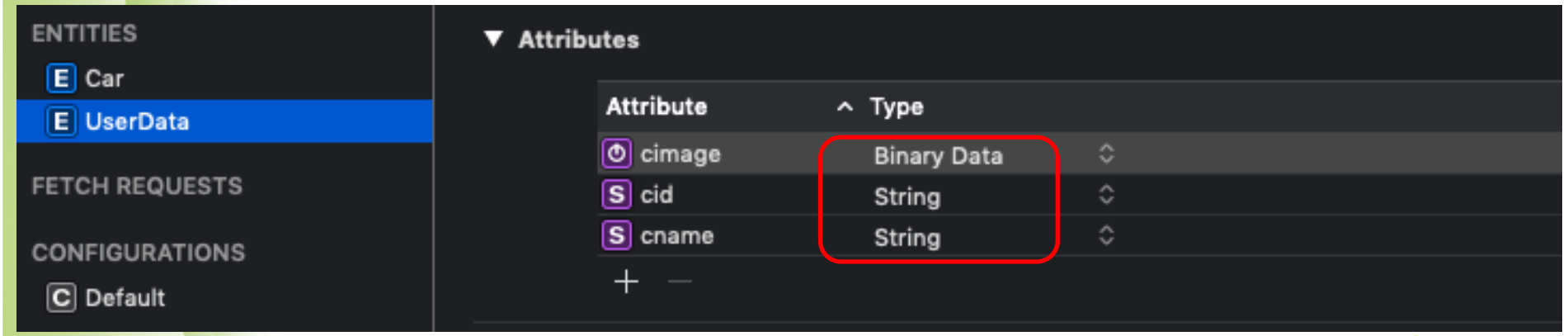


加入2個entity
Entity ≈ Table

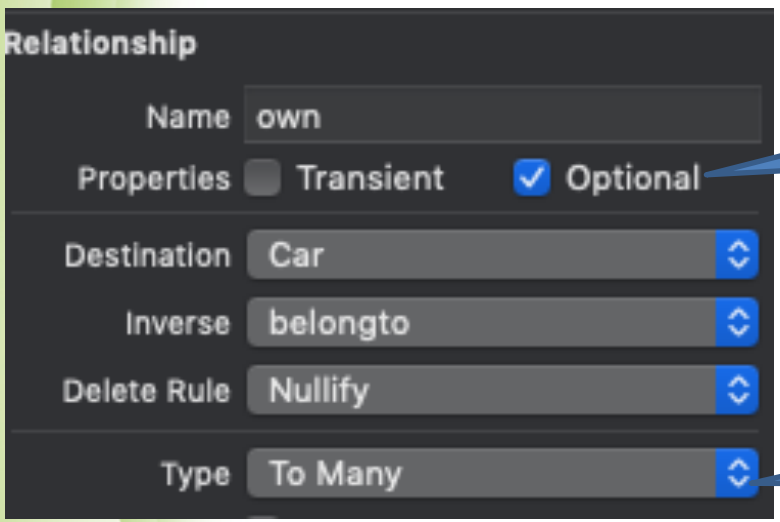


Attribute相當於
SQL中的Field





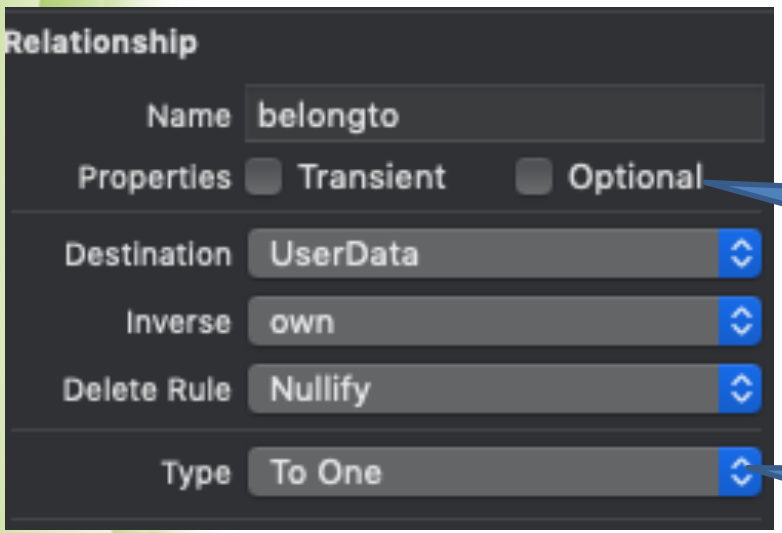
UserData relationships



客戶可能沒有車子

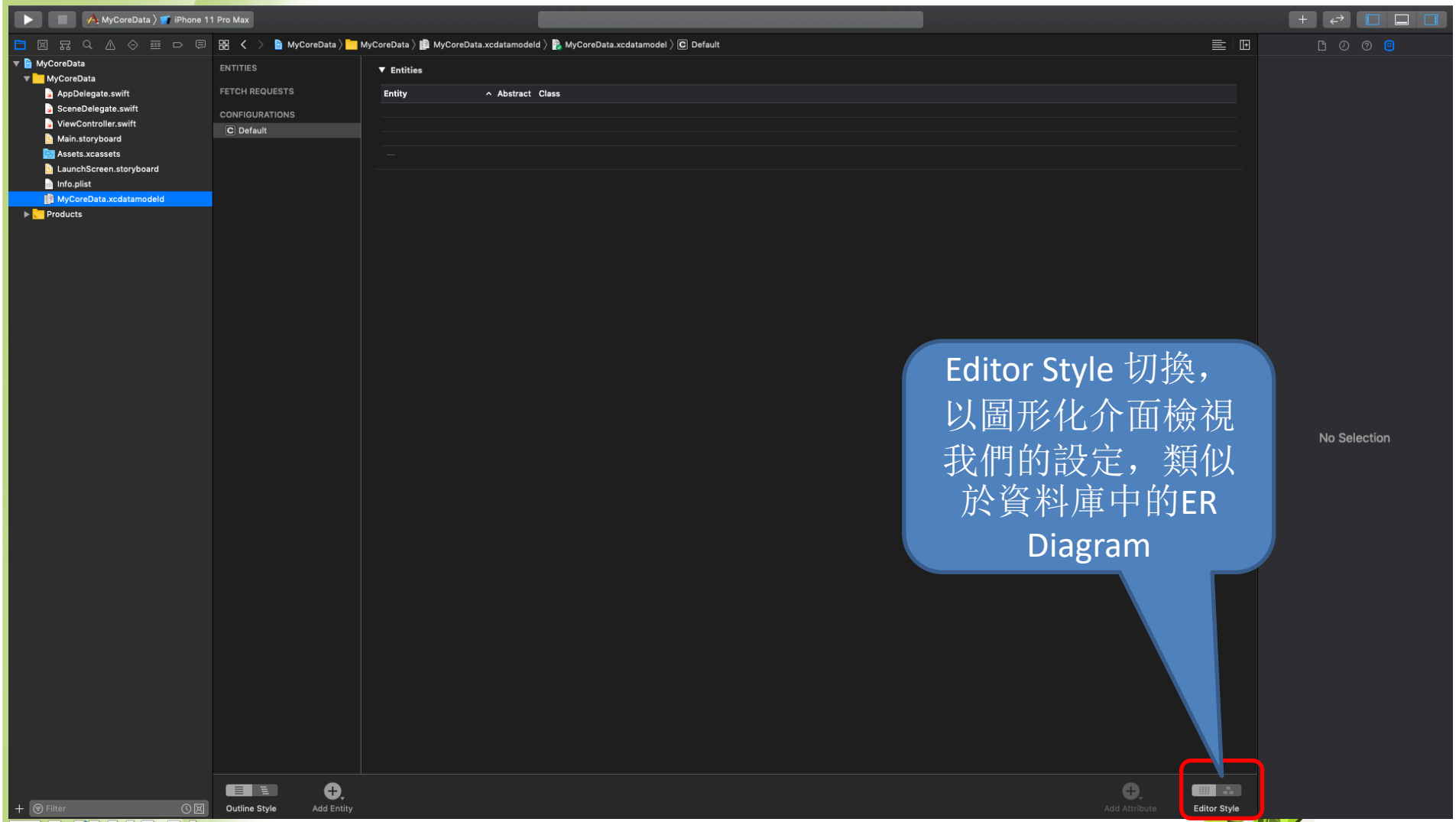
客戶可能有一部以上的車子

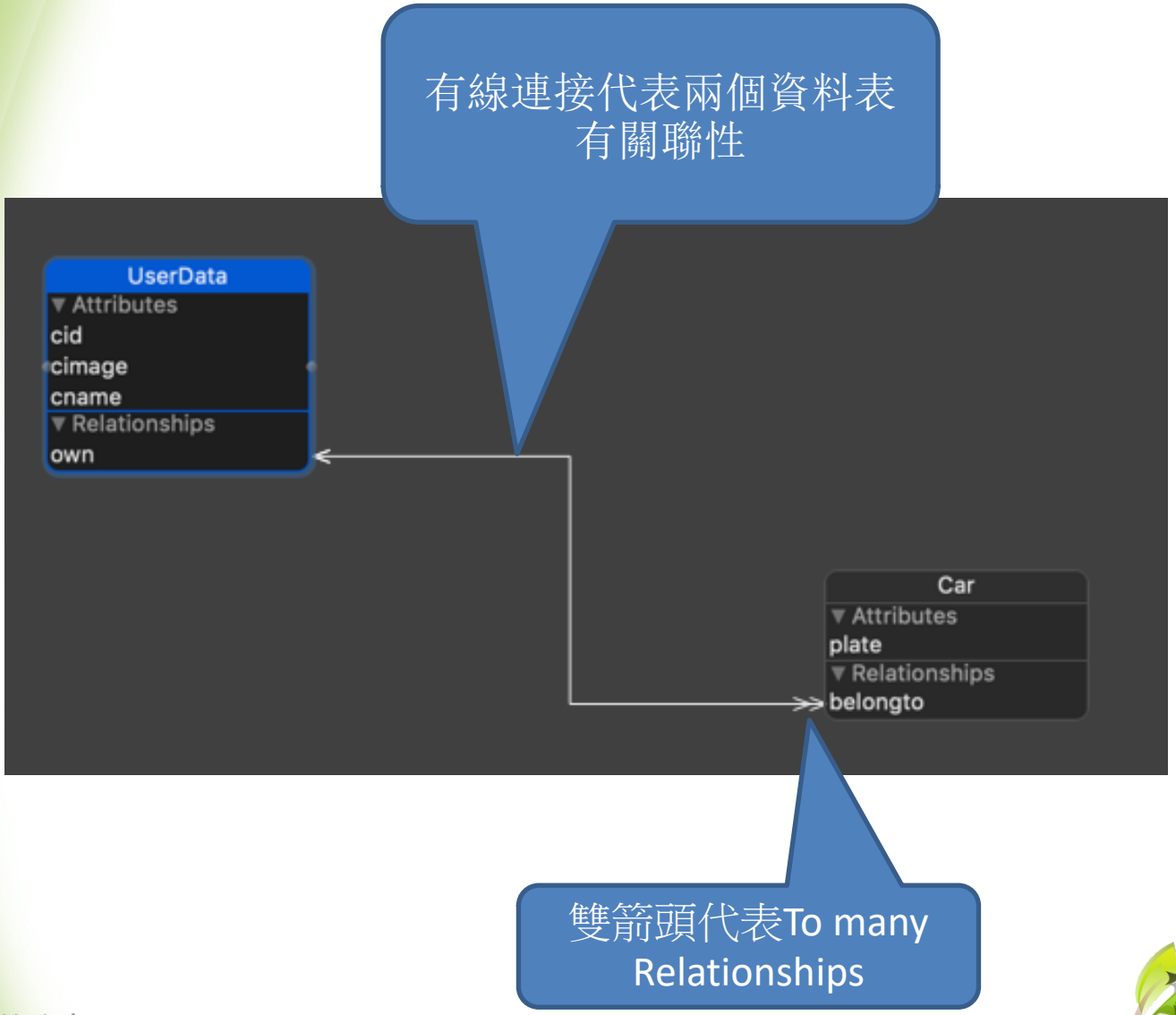
Car Relationships

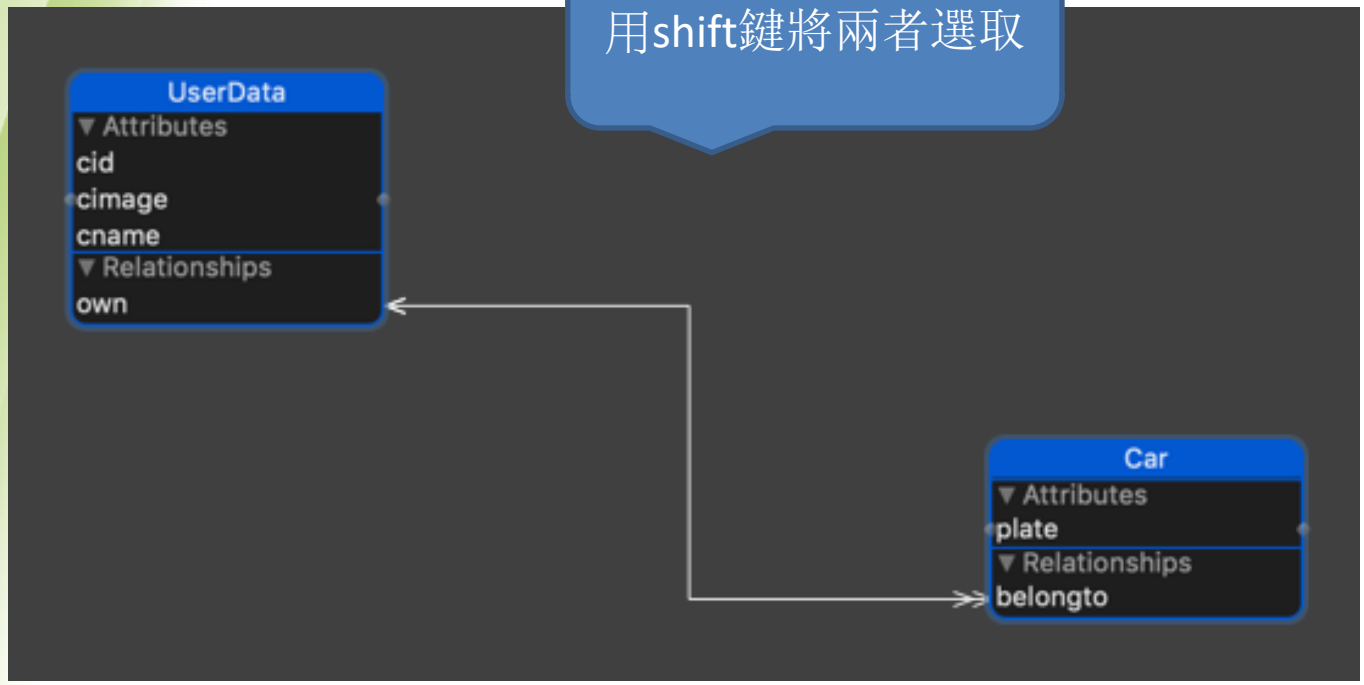


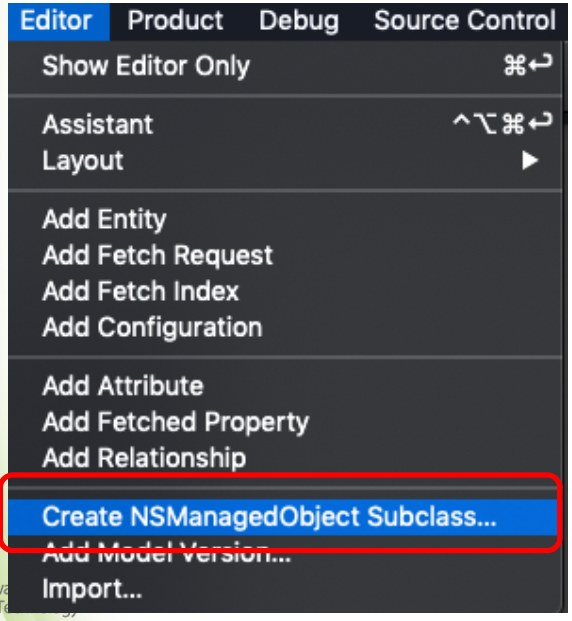
不允許Optional,因為每部車一定要有擁有者

每部車只會有一個擁有者

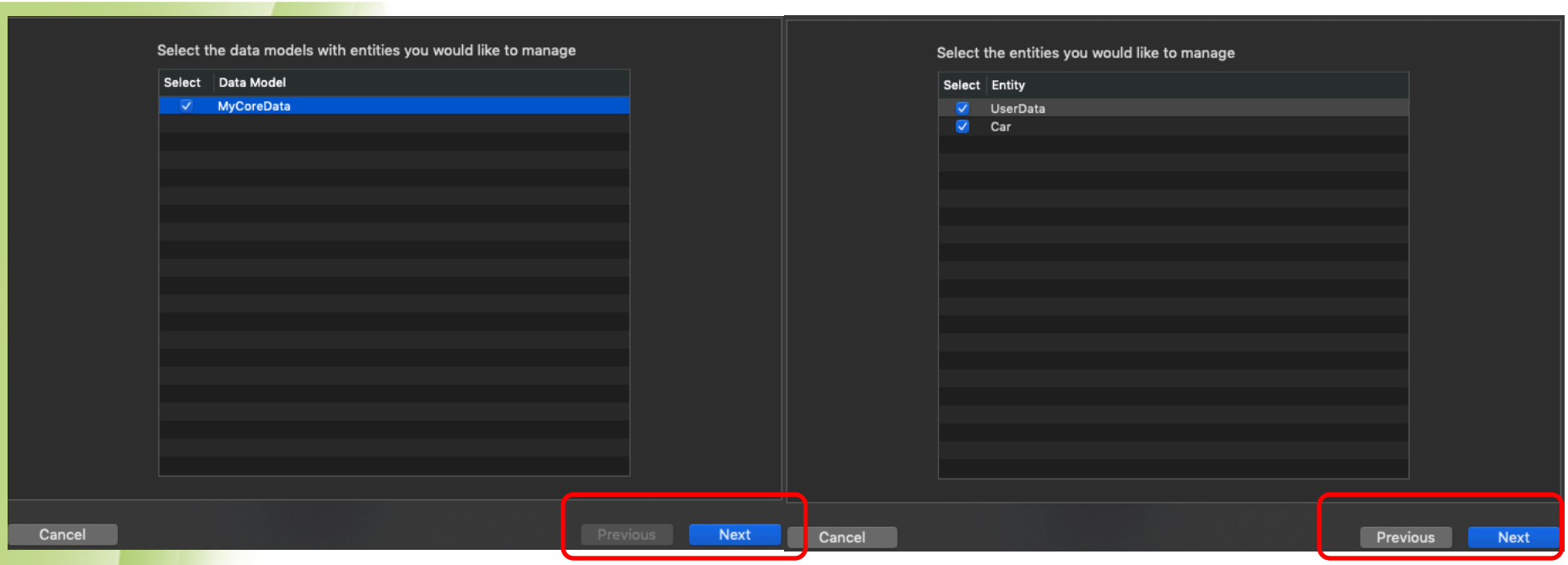

MyCoreData
iPhone 11 Pro Max
MyCoreData
MyCoreData
MyCoreData.xcdatamodeld
MyCoreData.xcdatamodel
Default
MyCoreData
MyCoreData
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MyCoreData.xcdatamodeld
Products
ENTITIES
FETCH REQUESTS
CONFIGURATIONS
Default
Entities
Entity
Abstract Class
No Selection
Editor Style 切換，以圖形化介面檢視我們的設定，類似於資料庫中的ER Diagram
Outline Style
Add Entity
Add Attribute
Editor Style

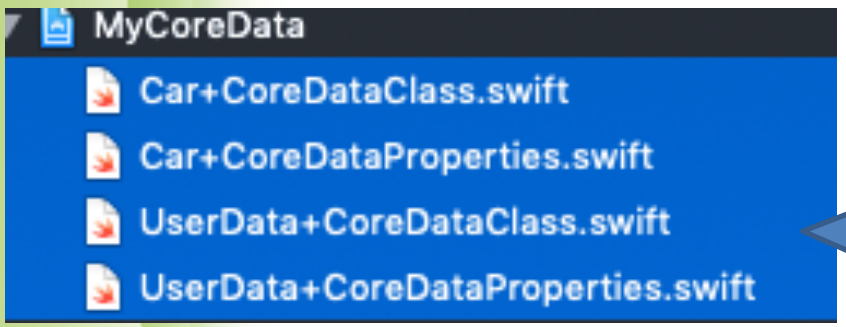

有線連接代表兩個資料表
有關聯性
UserData
Attributes
cid
cimage
cname
Relationships
own
Car
Attributes
plate
Relationships
belongto
雙箭頭代表To many
Relationships





專案會針對所選之entity建立所需類別檔,用來存取Core Data資料





通常我們不會去修改這幾個檔案的內容
若有改變entity的資料或關係，則重新
Create NSManagedObject Subclass即可

Core Data
存取

當建立一個有CoreData的app時 ,AppDelegate中會自動建好所需code

取得AppDelegate物件後即可取得MOC

ViewController

```
1  import UIKit
2  import CoreData
3
4  class ViewController: UIViewController {
5      let app = UIApplication.shared.delegate as! AppDelegate
6      var viewContext: NSManagedObjectContext!
7
8      override func viewDidLoad() {
9          super.viewDidLoad()
10         // Do any additional setup after loading the view.
11         viewContext = app.persistentContainer.viewContext
12         print(NSPersistentContainer.defaultDirectoryURL())
13         //deleteAllUserData()
14         insertUserData()
15         queryAllUserData()
16     }
```

顯示資料實際存放位置

呼叫NSEntityDescription中的insertNewObject函數來新增一筆MO(資料)，將暫時存放MOC中，所對應的entity為forEntityName定義

```
17      func insertUserData()   {
18          var user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
                viewContext) as! UserData
19          user.cid = "M10815055"
20          user.cname = "Yuan"
21
22          user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
                viewContext) as! UserData
23          user.cid = "M10815066"
24          user.cname = "Evan"
25
26          app.saveContext()
27      }
```

新增完之後呼叫MOC來做存入CoreData的動作

在do之中的try函式若發生錯誤，則跳到catch區塊處理

呼叫MOC做一個對UserData的存取請求

```
28      func queryAllUserData()     {
29          do {
30              let allUsers = try viewContext.fetch(UserData.fetchRequest())
31              for user in allUsers as! [UserData] {
32                  print("\((user.cid)!),\((user .cname)!)")
33              }
34          }   catch{
35              print(error)
36          }
37      }
```

利用迴圈將所有資料都打印出來

```
38      func deleteAllUserData()   {
39          do {
40              let allUsers = try viewContext.fetch(UserData.fetchRequest())
41              for user in allUsers as! [UserData] {
42                  viewContext.delete(user)
43              }
44              app.saveContext()
45              print("Successful delete")
46          }   catch{
47              print(error)
48          }
49      }
```

Delete函式可以
刪除此筆資料

```
file:///Users/hpc/Library/Developer/CoreSimulator/Devic
    es/8DF24519-BE45-4B9E-A968-9AB583FAFAAC/data/Contai
    ners/Data/Application/946F34E9-7D75-432B-8A87-6DE41
    0FB4F50/Library/Application%20Support/
M10815066,Evan
M10815055,Yuan
```

Core Data

查詢附帶條件並排序

```swift
override func viewDidLoad() {
    super.viewDidLoad()
    // Do any additional setup after loading the view.
    viewContext = app.persistentContainer.viewContext
    print(NSPersistentContainer.defaultDirectoryURL())
    //deleteAllUserData()
    insertUserData()
    //queryAllUserData()
    queryWithPredicate()
}
```

```swift
func insertUserData()   {
    var user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
        viewContext) as! UserData
    user.cid = "M10815071"
    user.cname = "Alex"

    user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
        viewContext) as! UserData
    user.cid = "M10815060"
    user.cname = "Andy"

    user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
        viewContext) as! UserData
    user.cid = "M10815069"
    user.cname = "Abby"

    user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
        viewContext) as! UserData
    user.cid = "M10815055"
    user.cname = "Yuan"

    app.saveContext()
}
```

返回結果type

For entity name:

```
59      func queryWithPredicate(){
60          let fetchRequest: NSFetchRequest<UserData> = UserData.fetchRequest()
61          let predicate = NSPredicate(format: "cname like 'A*'")
62          fetchRequest.predicate = predicate
            let sort = NSSortDescriptor(key: "cid", ascending: true)
            fetchRequest.sortDescriptors = [sort]

            do{
67              let allUsers = try viewContext.fetch(fetchRequest)
68              for user in allUsers    {
69                  print("\((user.cid)!), \((user.cname)!)")
70              }
71          }catch{
72              print(error)
73          }
74      }
```

做物件之間的排序

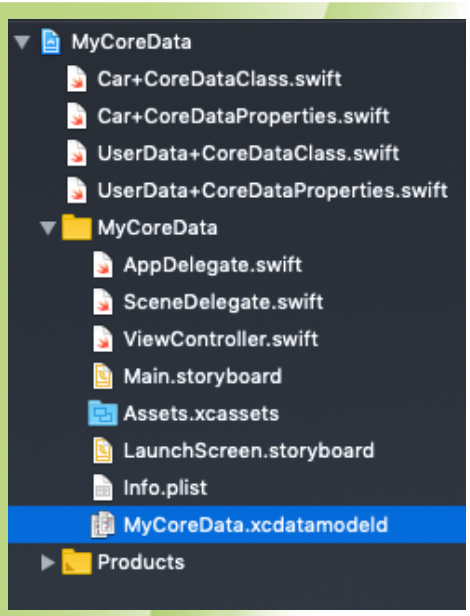
是否遞增排序

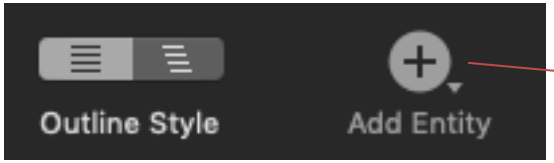
敘述的格式定義為A開頭類似於Where

```
file:///Users/hpc/Library/Developer/CoreSimulator/Devic
    es/8DF24519-BE45-4B9E-A968-9AB583FAFAAC/data/Contai
    ners/Data/Application/92ED7AFD-8DD9-4624-BD7F-8A1B3
    CDDE255/Library/Application%20Support/
M10815060, Andy
M10815069, Abby
M10815071, Alex
```

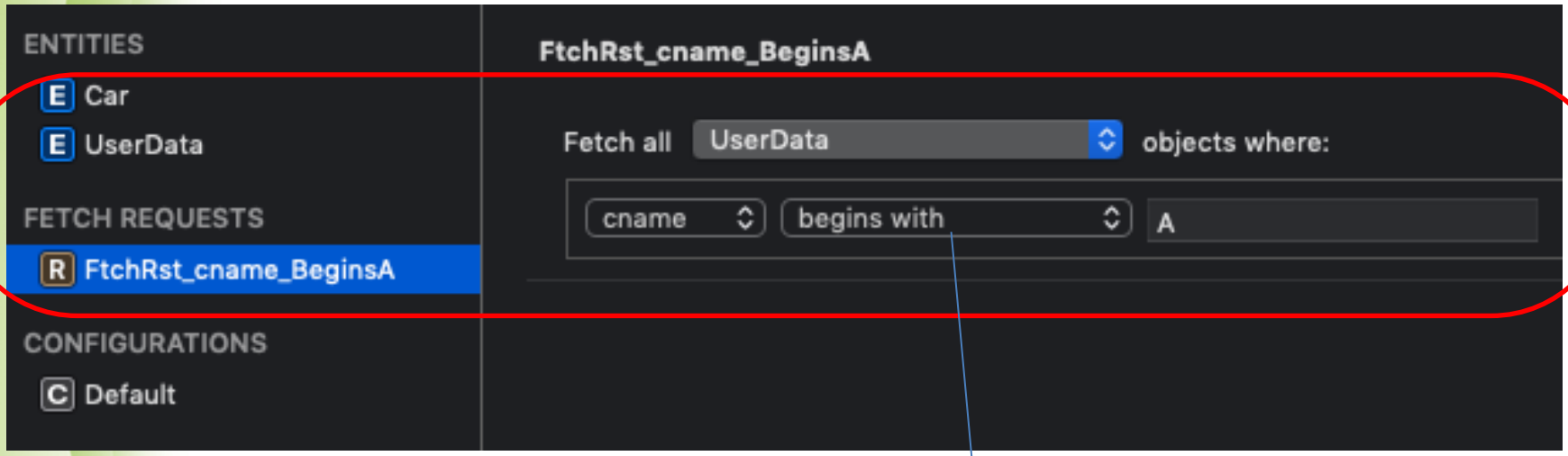
Core Data

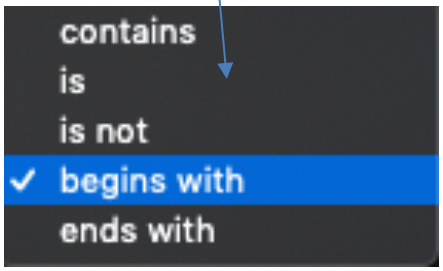
Fetch Requests


MyCoreData
Car+CoreDataClass.swift
Car+CoreDataProperties.swift
UserData+CoreDataClass.swift
UserData+CoreDataProperties.swift
MyCoreData
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MyCoreData.xcdatamodeld
Products


Outline Style
Add Entity

Long press


ENTITIES
Car
UserData
FETCH REQUESTS
FtchRst_cname_BeginsA
CONFIGURATIONS
Default
FtchRst_cname_BeginsA
Fetch all
UserData
objects where:
cname
begins with
A


contains
is
is not
begins with
ends with

```
8       override func viewDidLoad() {
9           super.viewDidLoad()
10          // Do any additional setup after loading the view.
11          viewContext = app.persistentContainer.viewContext
12          print(NSPersistentContainer.defaultDirectoryURL())
13          //deleteAllUserData()
14          //insertUserData()
15          //queryAllUserData()
16          //queryWithPredicate()
17          storedFetch()
18      }
```

```
76      func storedFetch(){
77          let model = app.persistentContainer.managedObjectModel
78          if let fetechRequest = model.fetchRequestTemplate(forName: "FtchRst_cname_BeginsA"){
79              do{
80                  let allUsers = try viewContext.fetch(fetechRequest)
81                  for user in allUsers as! [UserData]{
82                       print("\((user.cid)!), \((user.cname)!)")
83                  }
84              }catch {
85                  print(error)
86              }
87          }
88      }
89  }
```

```
file:///Users/hpc/Library/Developer/CoreSimulator/Devices/8DF24519
    -BE45-4B9E-A968-9AB583FAFAAC/data/Containers/Data/Application/
    87F7759A-342F-4970-84F7-09E1498375C9/Library/Application%20Sup
    port/
M10815060, Andy
M10815069, Abby
M10815071, Alex
```

比較方便而已
通常用在當App開啟時即抓取所需資料
例如對於餐廳介紹App會預先抓在你附近的資料出來顯示

Core Data

存取一對多關係

```
99      func insert_onetooMany(){
100         let user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into: viewContext) as!
                UserData
101         user.cid = "M10955933"
102         user.cname = "Jack"
103         var car = NSEntityDescription.insertNewObject(forEntityName: "Car", into: viewContext) as! Car
104         car.plate = "811-MYG"
105         user.addToOwn(car)
106
107         car = NSEntityDescription.insertNewObject(forEntityName: "Car", into: viewContext) as! Car
108         car.plate = "BBT-9088"
109         user.addToOwn(car)
110         app.saveContext()
111     }
```

```
112     func query_onetooMany(){
113         let fetchRequest: NSFetchRequest<UserData> = UserData.fetchRequest()
114         let predicate = NSPredicate(format: "cid like 'M10955933'")
115         fetchRequest.predicate = predicate
116         do{
117             let allUsers = try viewContext.fetch(fetchRequest)
118             for user in allUsers  {
119                 if  user.own == nil  {
120                     print("\((user.cname)!), 沒有車")
121                 }
122                 else   {
123                     print("\((user.cname)!) 有 \((user.own?.count)!)部車")
124                     for car in user.own as! Set<Car>    {
125                         print("車牌是 \((car.plate)!)")
126                     }
127                 }
128             }
129         }catch {
130             print(error)
131         }
132     }
```

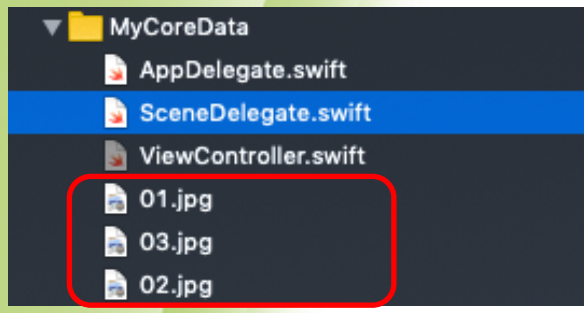
客戶所擁有的car

```
8       override func viewDidLoad() {
9           super.viewDidLoad()
10          // Do any additional setup after loading the view.
11          viewContext = app.persistentContainer.viewContext
12          print(NSPersistentContainer.defaultDirectoryURL())
13          //deleteAllUserData()
14          //insertUserData()
15          //queryAllUserData()
16          //queryWithPredicate()
17          //storedFetch()
18          insert_onetooMany()
19          query_onetooMany()
20      }
```

```
file:///Users/hpc/Library/Developer/CoreSimulator/Devices/8DF24519
    -BE45-4B9E-A968-9AB583FAFAAC/data/Containers/Data/Application/
    8F31329B-5F72-4C43-85DC-DA83E8BC6930/Library/Application%20Sup
    port/
Jack 有 2部車
車牌是 BBT-9088
車牌是 811-MYG
```

Core Data
圖片

將素材放入專案中




```
150     func saveImage(){
151         let user = NSEntityDescription.insertNewObject(forEntityName: "UserData", into:
                viewContext) as! UserData
152         user.cid = "M10715044"
153         user.cname = "Jessica"
154         let image = UIImage(named: "01.jpg")
155         let imageData  = image?.pngData()
156         user.cimage = imageData
157         app.saveContext()
158     }
```

Image要存在CoreData中
需要先將圖片轉型為
NSData
NSData為binary type
通常用來儲存圖片與音訊

```
    @IBOutlet weak var MyImage: UIImageView!
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
        viewContext = app.persistentContainer.viewContext
        print(NSPersistentContainer.defaultDirectoryURL())
        //deleteAllUserData()
        //insertUserData()
        //queryAllUserData()
        //queryWithPredicate()
        //storedFetch()
        //insert_onetooMany()
        //query_onetooMany()
        saveImage()
        loadImage()
    }
```

```
func loadImage(){
    let fetchRequest: NSFetchRequest<UserData> = UserData.fetchRequest()
    let predicate = NSPredicate(format: "cid like 'M10715044'")
    fetchRequest.predicate = predicate
    do{
        let allUsers = try viewContext.fetch(fetchRequest)
        for user in allUsers  {
            MyImage.image = UIImage(data: user.cimage! as Data)
        }
    }catch {
        print(error)
    }
}
```

條件為剛剛新增的客戶

將NSData轉回image形式

Run it

iPhone 11 Pro Max — 13.3

Core Data practice

iPhone 11 Pro Max — 13.3

iPhone 11 Pro Max — 13.3

修改至如圖所示

StackView 中含有imageView, Constraints需要設定，否則選取圖片可能跑版

用此按鈕選取相簿裡的圖片

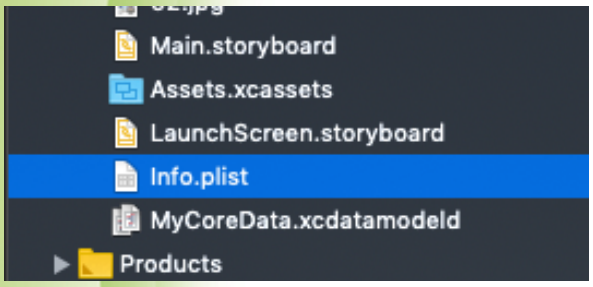
此按鈕用來清空所有資料(螢幕上的)

此按鈕來讀取資料

此按鈕將資料寫入CoreData

In func viewDidLoad()

```
for i in 1...3{
    if let image = UIImage(named: "0\(i).jpg"){
        UIImageWriteToSavedPhotosAlbum(image, nil, nil, nil)
    }
}
```



Privacy — Photo Library Additions Usage Description 給予寫入相簿權限

Privacy — Photo Library Usage Description 給予讀取相簿權限

請對應的連入viewController

```
    @IBOutlet weak var carplate: UITextField!
    @IBOutlet weak var MyImage: UIImageView!
    @IBOutlet weak var clientid: UITextField!
    @IBOutlet weak var clientName: UITextField!

    @IBAction func clearinfo(_ sender: UIButton) {
        clientName.text = ""
        clientid.text = ""
        carplate.text = ""
        MyImage.image = nil
    }
    @IBAction func clickreturn(_ sender: UITextField) {
        sender.resignFirstResponder()
    }
    @IBAction func selected(_ sender: UIButton) {
        let imagePicker = UIImagePickerController()
        imagePicker.sourceType = .photoLibrary
        imagePicker.delegate = self
        imagePicker.modalPresentationStyle = .popover
        show(imagePicker, sender: MyImage)
    }
```

清空所有資料(非CoreData)

按下return鍵收起鍵盤

選取圖片按鈕

```
class ViewController:
    UIViewController,UIImagePickerControllerDelegate
    ,UINavigationControllerDelegate {
```

```
51      func imagePickerController(_ picker: UIImagePickerController,
            didFinishPickingMediaWithInfo info: [UIImagePickerController.InfoKey : Any]) {
52          let image = info[UIImagePickerController.InfoKey.originalImage] as! UIImage
53          MyImage.image = image
54          dismiss(animated: true, completion: nil)
55      }
```

如果有欄位是空的，
利用AlertController來提醒
使用者

```
    @IBAction func saveData(_ sender: UIButton) {
        if (clientid.text == "") || (clientName.text == "") || (carplate.text == "") ||
            (MyImage.image == nil){
            MyAlertController("Error")
        }else{
            let user = NSEntityDescription.insertNewObject(forEntityName: "UserData",
                into: viewContext) as! UserData
            user.setValue(clientName.text, forKey: "cname")
            user.setValue(clientid.text, forKey: "cid")
            user.setValue(MyImage.image?.pngData(), forKey: "cimage")
            let car = NSEntityDescription.insertNewObject(forEntityName: "Car", into:
                viewContext) as! Car
            car.setValue(carplate.text, forKey: "plate")
            user.addToOwn(car)
            app.saveContext()
            MyAlertController("Successful insert")
        }
    }
```

```
    @IBAction func loadData(_ sender: UIButton) {
        let fetchId = NSPredicate(format: "cid BEGINSWITH[cd] %@", clientid.text!)
        let fetchName = NSPredicate(format: "cname BEGINSWITH[cd] %@", clientName.text!)
        let fetchRequest : NSFetchRequest<UserData> = UserData.fetchRequest()
        var predicate = NSCompoundPredicate()
        if (clientid.text != "") && (clientName.text == "") {
            predicate = NSCompoundPredicate(andPredicateWithSubpredicates: [fetchId])
            }
        else if (clientid.text == "" && clientName.text != ""){
            predicate = NSCompoundPredicate(andPredicateWithSubpredicates: [fetchName])
        }
        else{
            predicate = NSCompoundPredicate(andPredicateWithSubpredicates:
                [fetchId,fetchName])
        }
        fetchRequest.predicate = predicate
        do {
            let Users = try viewContext.fetch(fetchRequest)
            if Users == [] {
                MyAlertController("Unsuccessful load")
            }
            for user in Users  {
                clientid.text = user.cid
                clientName.text = user.cname
                MyImage.image = UIImage(data: user.cimage! as Data)
                for car in user.own as! Set<Car> {
                    carplate.text = car.plate
                }
            }
        }catch{
            print(error)
        }
    }
```

利用id或name來抓取資料

NSCompoundPredicate 可以執行一連串的敘述

根據敘述搜尋後，沒有相應結果

將CoreData中的image轉回來

```
103     func MyAlertController(_ result: String){
104         var alert = UIAlertController()
105         if result == "Error" {
106             alert = UIAlertController(title: "Error", message:"Please enter complete
                    information", preferredStyle: .alert)
107         }
108         else if (result == "Successful insert"){
109             alert = UIAlertController(title: "Successful insert", message:
                    String(clientName.text!)+" added finish", preferredStyle: .alert)
110         }
111         else {
112             alert = UIAlertController(title: "Unsuccessful load", message:"There is no
                    such data", preferredStyle: .alert)
113         }
114         let action = UIAlertAction(title: "I got it!", style: .default, handler: nil)
115         alert.addAction(action)
116         self.present(alert,animated: true,completion: nil)
117     }
```

Run it

請練習做出
If 有輸入車牌
則以車牌搜尋車主相關資料
else
則以id或name搜尋相關資料