



THE UNIVERSITY OF
CHICAGO



ADVANCED iOS APPLICATION DEVELOPMENT

MPCS 51032 • SPRING 2020 • SESSION 4

APP GROUP

WHAT IS AN APP GROUP?

- Allow applications and targets to share data via a special directory (outside of the sandbox)
 - Creates a folder that can be read/written to by different targets
 - Coordinated with your developer account



APP GROUP

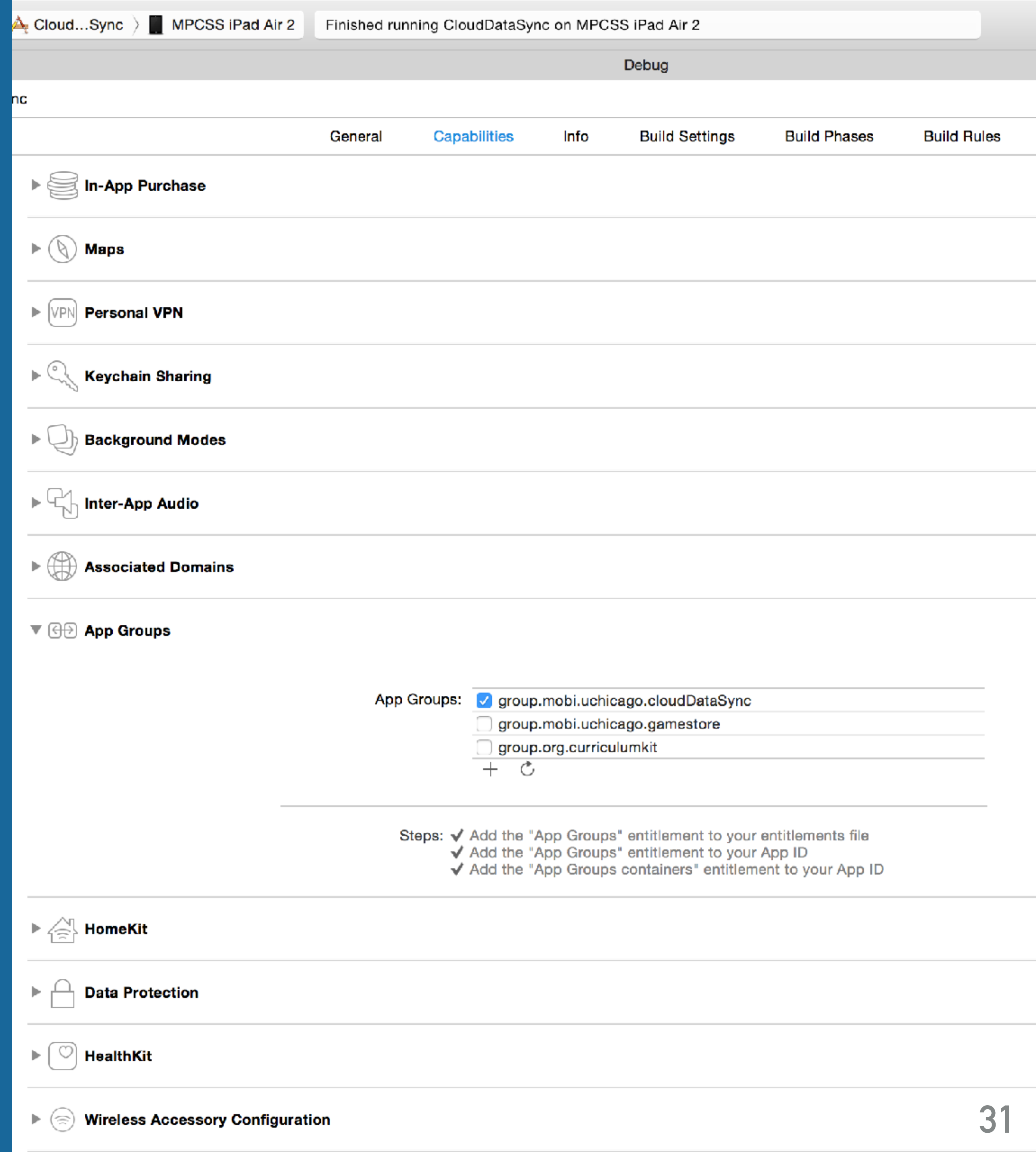
- This should become the default place you think about writing data for all of your applications



APP GROUPS

SUBTITLE

- App Groups are allowed to share access to a folder
- When sharing data you must be aware of who may be accessing the data at different times (not obvious)
 - Deal with concurrent writes through NSFileCoordination
 - CoreData and NSUserDefaults handle this out of the box



APP GROUPS

CREATE A NEW GROUP
(DEVELOPER BASED)

The screenshot displays the Xcode development environment. On the left, the Project Navigator shows a project named 'DaysDemo' with various source files including 'Journal.swift', 'AppDelegate.swift', 'ViewController.swift', 'JournalTabl...ntroller.swift', 'Main.storyboard', 'Assets.xcassets', 'LaunchScreen.storyboard', 'Info.plist', and a 'DataKit' folder. The 'DataKit' folder is expanded, showing 'DataKit.h', 'Info.plist', and 'Greeting.sv'. The main editor area shows the 'DaysDemo' project selected, with a 'DataKit' folder icon visible. On the right, the 'App Groups' settings panel is open, showing a list of app groups: 'group.daysdemo' (checked), 'group.2016-extend-this-app', and 'group.2017-extend-this-app'. Below the list are '+' and refresh icons. The 'App Groups' toggle is turned 'ON'. Below the settings, a 'Steps' section lists three tasks: 'Add the App Groups entitlement to your entitlements file', 'Add the App Groups feature to your App ID.', and 'Add App Groups to your App ID'. Other settings like 'Keychain Sharing', 'Associated Domains', 'Data Protection', and 'HomeKit' are visible but turned 'OFF'.

Project Files:

- Journal.swift
- DaysDemo.entitlements
- AppDelegate.swift
- ViewController.swift
- JournalTabl...ntroller.swift
- Main.storyboard
- Assets.xcassets
- LaunchScreen.storyboard
- Info.plist
- DataKit
 - DataKit.h
 - Info.plist
 - Greeting.sv
- Products
- Frameworks

App Groups Settings:

- App Groups: ☒ group.daysdemo, ☐ group.2016-extend-this-app, ☐ group.2017-extend-this-app
- + (Add new group)
- ↻ (Refresh)

Steps:

- ✓ Add the App Groups entitlement to your entitlements file
- ✓ Add the App Groups feature to your App ID.
- ✓ Add App Groups to your App ID

Other Settings:

- Keychain Sharing: OFF
- Associated Domains: OFF
- Data Protection: OFF
- HomeKit: OFF

App Groups

The screenshot shows the Xcode interface with the DaysDemo.entitlements file selected in the left sidebar. The main area displays the contents of this file as a table. A yellow callout points to the file in the sidebar, and another points to the 'group.daysdemo' value in the table.

Key	Type	Value
▼ Entitlements File	Dictionary	(4 items)
APS Environment	String	development
▼ iCloud Container Identifiers	Array	(1 item)
Item 0	String	iCloud.\$(CFBundleIdentifier)
▼ iCloud Services	Array	(1 item)
Item 0	String	CloudKit
▼ App Groups	Array	(1 item)
Item 0	String	group.daysdemo

File created when using groups

Same info as in GUI

APP GROUPS

```
// Print out the app group (for fun)
let sharedAppGroup: String = "group.2017-extend-this-app"
let directory: NSURL = FileManager.default.containerURL(forSecurityApplicationGroupIdentifier: sharedAppGroup)! as NSURL
print("App Group URL: \(directory)")
```

Access the app group container using file manager

- Can read and write to this directory from a container application and application extension points
- No other difference (its just a directory)

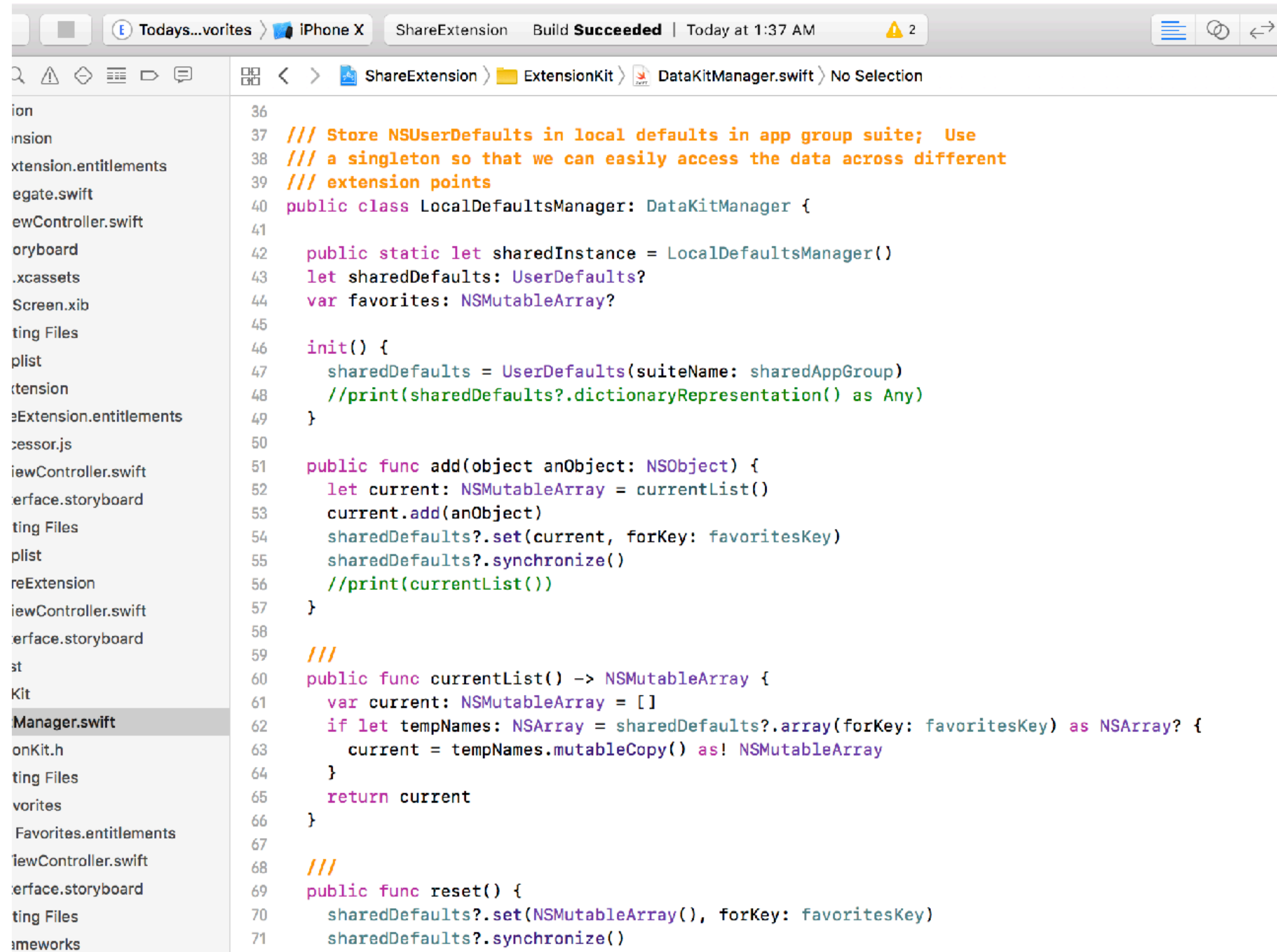
APP GROUPS

```
sharedDefaults = UserDefaults(suiteName: sharedAppGroup)
print(sharedDefaults?.dictionaryRepresentation() as Any)
```

- User defaults has a special API for shared defaults in app groups

APP GROUPS

- Playground
 - DataKitManager accessible by different targets



The screenshot shows the Xcode IDE with the 'ShareExtension' target selected. The file 'DataKitManager.swift' is open, showing the implementation of the 'LocalDefaultsManager' class. The code is written in Swift and uses the 'DataKitManager' protocol. The class is a singleton and stores user defaults in a local app group suite. It includes methods for adding, resetting, and retrieving a list of favorites.

```
36
37 /// Store UserDefaults in local defaults in app group suite; Use
38 /// a singleton so that we can easily access the data across different
39 /// extension points
40 public class LocalDefaultsManager: DataKitManager {
41
42     public static let sharedInstance = LocalDefaultsManager()
43     let sharedDefaults: UserDefaults?
44     var favorites: NSMutableArray?
45
46     init() {
47         sharedDefaults = UserDefaults(suiteName: sharedAppGroup)
48         //print(sharedDefaults?.dictionaryRepresentation() as Any)
49     }
50
51     public func add(object anObject: NSObject) {
52         let current: NSMutableArray = currentList()
53         current.add(anObject)
54         sharedDefaults?.set(current, forKey: favoritesKey)
55         sharedDefaults?.synchronize()
56         //print(currentList())
57     }
58
59     ///
60     public func currentList() -> NSMutableArray {
61         var current: NSMutableArray = []
62         if let tempNames: NSArray = sharedDefaults?.array(forKey: favoritesKey) as NSArray? {
63             current = tempNames.mutableCopy() as! NSMutableArray
64         }
65         return current
66     }
67
68     ///
69     public func reset() {
70         sharedDefaults?.set(NSMutableArray(), forKey: favoritesKey)
71         sharedDefaults?.synchronize()
```




THE UNIVERSITY OF
CHICAGO



ADVANCED iOS APPLICATION DEVELOPMENT

MPCS 51032 • SPRING 2020 • SESSION 4