Swift 分享

Swift 目前最新版本 4.2 beta

Lawrence Shen

學習資源

- Swift 基本語法最新版本 https://docs.swift.org/swift-book/ .
- Swift 正體中文版 https://tommy60703.gitbooks.io/swift-language-traditional-chinese/chapter1/01_swift.html 參考就好，版本好像是 Swift 2.x 語法 .
- CocoaPods
  Carthage https://gist.github.com/weihanglo/97e949a9dbf92deb111999b6e42e9654 .

01
Mac 基本環境


Finder
MyGitHub
Carthage
Mac
MyGit
CertificateSigning Request....Request
Visual Studio Code
com.edotw.um
testing.html
AppCleaner
Test Media
tiny_uploader.js
MacEm2
HRAppBuilder
DmgBuild
DB Browser for SQLite
Code
com.edotw.uninst
test.command

Mac 基本環境

- Apple Icon
- App Menu
- Dock
- Finder
- Launchpad
- AppStore

02
Xcode 與 Swift

Version	Released		Part of
Swift 4.2 Beta	2018	TBA	Xcode 10 Beta
Swift 4.1	2018	March 29	Xcode 9.3
Swift 4.0.3	2017	Dec 5	Xcode 9.2
Swift 4.0.2		Nov 1	Xcode 9.1
Swift 4.0		Sept 19	Xcode 9.0, Xcode 9.0.1
Swift 3.1.1		Apr 21	Xcode 8.3.2, Xcode 8.3.3
Swift 3.1		Mar 27	Xcode 8.3, Xcode 8.3.1
Swift 3.0.2	2016	Dec 13	Xcode 8.2, Xcode 8.2.1
Swift 3.0.1		Oct 28	Xcode 8.1
Swift 3.0		Sept 13	Xcode 8.0
Swift 2.3			Xcode 8.0, Xcode 8.1, Xcode 8.2, Xcode 8.2.1
Swift 2.2.1		May 3	Xcode 7.3.1
Swift 2.2		Mar 21	Xcode 7.3

03
Swift 常用語法

let 、var 關鍵字

```
//變數宣告
var swift : String?;
var swift: String = "";
var swift = "";

//常數宣告，使用方式同上
let swift = "";
```

03
Swift 常用語法

_、?、!關鍵字

```
var variable: String?;

let newVariable: String = variable!;

if let myClass = variable as? SomeClass { /* Do something */ }
```

```
if let _ = variable { /* Do something */ }

let _ = myFunc("Name");

func myFunc(_ variable: String) -> String { return "Hello \(variable)"; }
```

03 Swift 常用語法

let、guard 關鍵字

```
var variable: String?;

if let newVar = variable {
    print(newVar);
}
else {
    return;
}

guard let newVar = variable else { return; }
print(newVar);
```

```swift
override func viewDidLoad()
{
    super.viewDidLoad()

    let _ = try! myFunc("ok"); //如果你很肯定絕對不會發生錯誤，可以這樣用
    let _: String? = try? myFunc("ok"); //透過 ? 寫法，若發生錯誤會拋回一個 Optional 的回傳值

    do {
        let _ = try myFunc("lawrence");
    }
    catch CustomError.empty {
        print("error type is empty");
    }
    catch CustomError.notSwift {
        print("error type is notSwift");
    }
    catch { //剩下的錯誤自動放到 區域變數 error 中
        print("error type is other : \(error)");
    }
}

func myFunc(_ variable: String) throws -> String
{
    if variable.trimmingCharacters(in: .whitespaces) == "" { throw CustomError.empty; }
    if variable.lowercased() == "swift" { throw CustomError.notSwift; }
    if variable.lowercased() == "lawrence" { throw NSError(domain: "ohoh", code: 0, userInfo: nil) }
    return "Hello \(variable)";
}

enum CustomError: Error
{
    case empty
    case notSwift;
}
```

04 例外處理

Swift 的例外處理機制只負責處理已知可能會拋出錯誤的方法，對於型別或者邏輯錯誤造成的異常無法攔截(大部分的情況下 Swift 編譯器會找出)，因此程式內常見大量的 if let & guard let 等判斷與法

Error Handling 說明

05 region替代

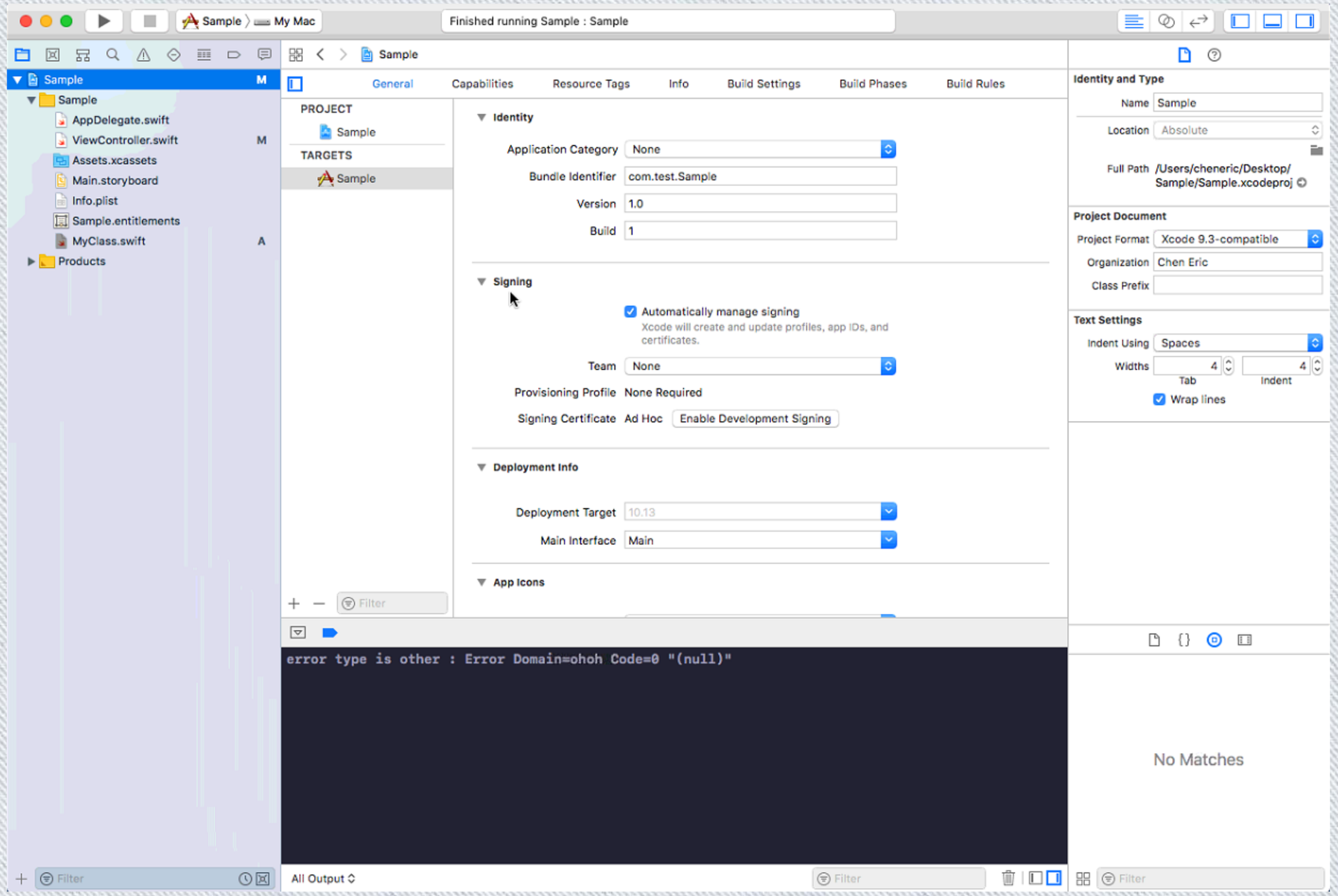
Swift 沒有類似 C# region 的區域，當程式碼很多的時候可以透過 Nav Bar 快速的找到方法外，另外一個就是使用 extension 的方式，將程式碼區隔在不同區塊，甚至是不同檔案。

```
import Foundation

class MyClass
{
    func MyClass() { }
}

//MARK: - region 替代方法
extension MyClass
{

}
```


Sample
My Mac
Finished running Sample : Sample
Sample
Sample
AppDelegate.swift
ViewController.swift
M
Assets.xcassets
Main.storyboard
Info.plist
Sample.entitlements
MyClass.swift
A
Products
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Sample
TARGETS
Sample
Identity
Application Category
None
Bundle Identifier
com.test.Sample
Version
1.0
Build
1
Signing
Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.
Team
None
Provisioning Profile
None Required
Signing Certificate
Ad Hoc
Enable Development Signing
Deployment Info
Deployment Target
10.13
Main Interface
Main
App Icons
error type is other : Error Domain=ohoh Code=0 "(null)"
All Output
Identity and Type
Name
Sample
Location
Absolute
Full Path
/Users/cheneric/Desktop/Sample/Sample.xcodeproj
Project Document
Project Format
Xcode 9.3-compatible
Organization
Chen Eric
Class Prefix
Text Settings
Indent Using
Spaces
Widths
4
Tab
4
Indent
Wrap lines
No Matches

Xcode 介紹

Xcode 介紹

- Xcode 操作區域
- Project 相關設定
- Storyboard & AutoLayout
- 引用第三方類別，使用Carthage
- Application、Window、View、Controller、Delegate
- 常用的轉場方式
- 常用的資料傳遞方式
- 建置

Carthage使用方式

1.開啟終端機，切換至預計儲存檔案的資料夾路徑，ex. cd Desktop/Carthage .
2.建立一個空的carthage文件，touch Cartfile .
3.使用Xcode打開該文件，open -a Xcode Cartfile .
4.編輯Cartfile，github "adamhartford/SwiftR" .
5.執行Carthage，carthage update --platform macOS .
6.Swift版本升級時，可能會出現錯誤，ex. Module compiled with swift 4.0 cannot be imported in swift 3.1，解決方式可參考.
7.若要清除快取可到 ~/Library/Caches/org.carthage.CarthageKit/DerivedData 資料夾清除.
8.Framework 簽入時請注意一併遷入資料夾內各項東西，XCode 的 git 可能會漏掉，如果可以請使用 VSCode 來確認，Carthage 目錄說明可參考.
9.使用到的套件 Charthage 備份到專案跟目錄底下.
10.Carthage安裝使用教學，到此處參考.
11.版本说明如下，更多說明請到此處參考：

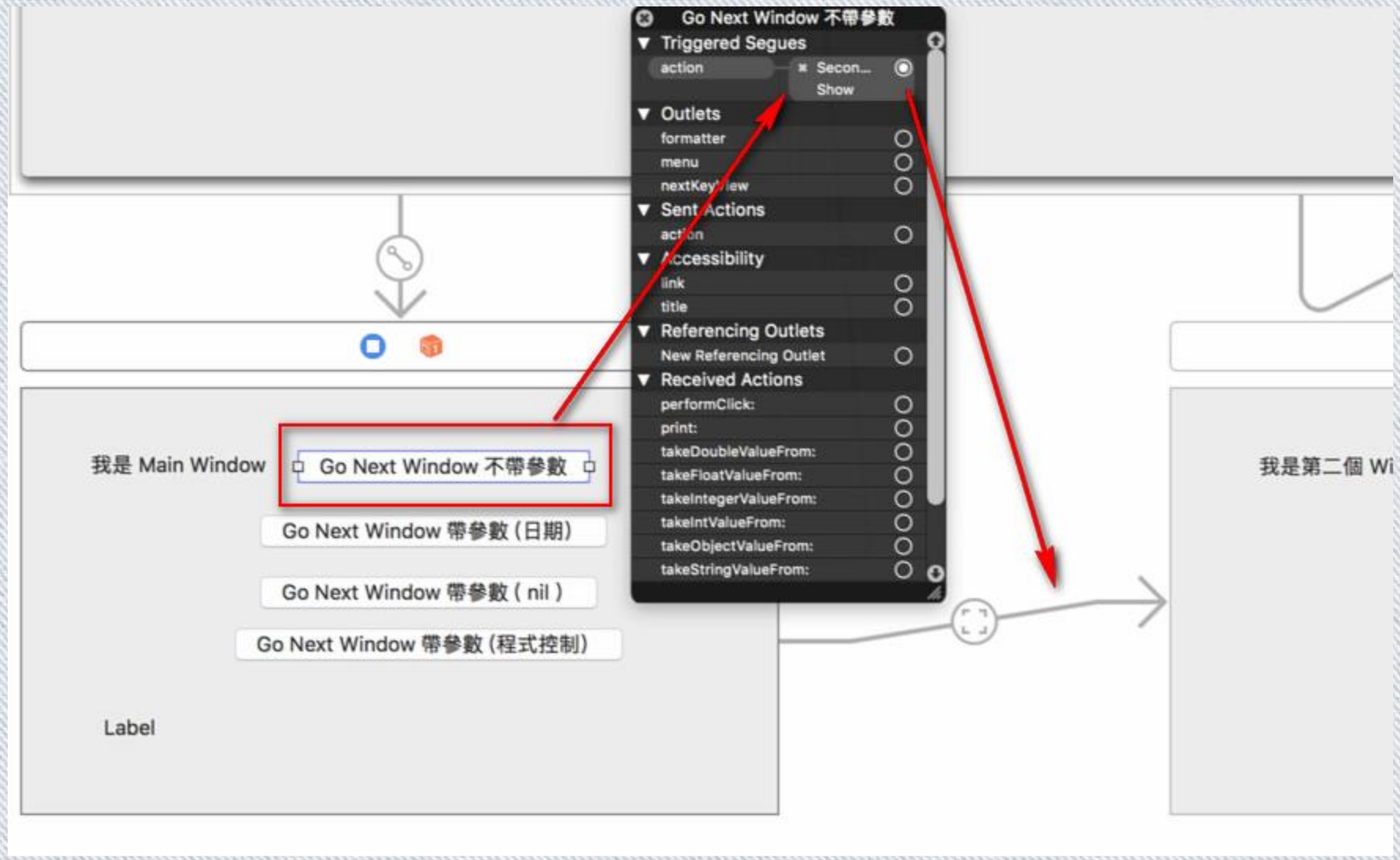
- ~> 3.0 表示使用版本3.0以上但是低於4.0的最新版本，如3.5, 3.9
- == 3.0 表示使用3.0版本
- //>= 3.0表示使用3.0或更高的版本
- 如果沒有特別指明版本號，則會自動使用最新的版本

常用的轉場方式

- Storyboard Element Action Segue 直接轉場
- Storyboard Controller Action Segue 轉場
- 程式控制轉場

Storyboard Element Action Segue 直接轉場

單純的按鈕切換頁面，不需要指定參數，可在 Storyboard 中的 Button 直接使用 Action Segue 來切換參數



Storyboard Controller Action Segue 轉場

多個按鈕共用同一個 Action Segue，並且需判斷不同按鈕需傳遞不同的參數，可在 Storyboard 中的 ViewController 拉取一條 Segue 並且定義 Identifier

1. 在 Storyboard Main ViewController 拖拉一個 Action Segue 到 Sub Controller （ WindowController 或 ViewController 視需求決定）。
2. 定義這條 Segue 的 Identifier 名稱。
3. 在 Button Action 中，使用 performSegue 傳遞至指定的 Segue。
4. （非必要）若轉場需要傳遞資料，override prepare 將資料傳遞過下一個頁面。

程式控制轉場

不依靠 Storyboard Segue

1. 在 Storyboard Sub ViewController 定義 Restoration ID。
2. 在 Button Action 中，從 Storyboard 中找出該 ID 的 ViewController。
3. 開啟視窗。
4. （非必要）若轉場需要傳遞資料，初始化資料。

常用的資料傳遞方式

- 使用 Protocol 回傳
- 通知中心傳值
- NSApplication.shared

使用 Protocol 回傳

最標準的回傳方式

1. 定義 protocol。
2. 在子視窗（被呼叫端）內需告該 protocol 的 delegate 並實作相關方法。
3. 在主視窗（呼叫端）繼承該 protocol。
4. 主視窗，在開啟視窗前的事件將 ViewController 的 delegate 指定為自己。
5. 主視窗實作相關方法用來接收子視窗的呼叫。

通知中心傳值

適合一對多通知

1. 建立一個 Listen。
2. 發送訊息。
3. 移除 Listen。

NSApplication.shared

大絕招

1. 在呼叫端直接搜尋要執行的 Controller 直接呼叫其屬性＆方法。

THANK YOU FOR WATCHING!

ANY QUESTIONS?