

Apple Map

Apple Map or Google Map?

Google Map 是要收費的，即使免費的部份，也必需經過電信認證才能使用(2018 年起)，這也是 Apple 當年被罵也要自己做地圖的原因之一。

Apple Map 在 iOS 或 macOS 上是完全免費的，其功能也相當完整。

而且使用起來非常簡單，特別適合新手上路

我的第一個 Apple Map 程式

1. 在 Storyboard 上, 拉一個 MKMapView
2. 拉成 IBOutlet
3. 在 ViewController 中加入 `import Mapkit`
4. 可以執行了, 正確的話就會出現台灣地圖

地圖會預設使用手機的語言, 若要用其他語言, 請先設好手機的語言

顯示一個區域

經度 + 緯度 = 位置

X比例 + Y比例 = 比例

位置 + 比例 = 範圍

地圖 > 範圍 > 完成

```
DispatchQueue.main.asyncAfter(deadline: .now() + 3) {  
    let latitude:CLLocationDegrees = 25.0444032  
    let longitude:CLLocationDegrees = 121.5141468  
    let location:CLLocationCoordinate2D = CLLocationCoordinate2DMake(latitude, longitude)  
    let xScale:CLLocationDegrees = 0.01  
    let yScale:CLLocationDegrees = 0.01  
    let span:MKCoordinateSpan = MKCoordinateSpan(latitudeDelta: yScale, longitudeDelta:  
xScale)  
    let region:MKCoordinateRegion = MKCoordinateRegion.init(center: location, span: span)  
    self.mapView.setRegion(region, animated: true)  
}
```

地圖的種類

`mapView.mapType`

`.standard` 一張街道地圖，顯示所有道路和一些道路名稱的位置。

`.satellite` 該地區的衛星圖像。

`.hybrid` 區域的衛星圖像，道路和道路名稱信息分層在上面。

`.satelliteFlyover` 具有 POI 的區域的衛星圖像(如果可用)。

`.hybridFlyover` 具有 POI 的混合衛星圖像(如果可用)。

`.mutedStandard` A street map where your data is emphasized over the underlying map details

地圖的屬性

Apple Map 有相當多的屬性，如放大縮小轉向等

<https://developer.apple.com/documentation/mapkit/mkmapview>

例如：

```
mapView.isScrollEnabled = false
```

加上大頭針

```
let annotation = MKPointAnnotation()  
annotation.coordinate = location  
annotation.title = "譯智"  
annotation.subtitle = "教育訓練中心 "  
self.mapView.addAnnotation(annotation)
```

自訂長鉞大頭針

1. 在Storyboard 加上 long press gesture 並加上設定
2. 把它拉一個 Action

```
@IBAction func mapLongPress(_ sender: UILongPressGestureRecognizer) {  
    let touchPoint = sender.location(in: mapView)  
    let location = mapView.convert(touchPoint, toCoordinateFrom: mapView)  
    let annotation = MKPointAnnotation()  
    annotation.coordinate = location  
    annotation.title = "自選點"  
    self.mapView.addAnnotation(annotation)  
}
```

CoreLocation

CoreLocation 是定位，也就是 GPS 相關的程式庫，要取得手機的定位，就要使用它，首先，要取得授權。要取得授權，要先設定好說明。請在 Info.plist 中，加入以下的設定，並在使用前 import CoreLocation

Key		Type	Value
▼ Information Property List		Dictionary	(16 items)
Privacy - Location When In Use Usage Description	⌵	String	我想要用你的位置
Application Language	⌵	String	
Localization native development region	⌵	String	\$(DEVELOPMENT_LANGUAGE)
Executable file	⌵	String	\$(EXECUTABLE_NAME)
Bundle identifier	⌵	String	\$(PRODUCT_BUNDLE_IDENTIFIER)

取得使用者位置

```
locationManager = CLLocationManager()  
locationManager?.requestWhenInUseAuthorization()
```

```
if let coordinate = locationManager?.location?.coordinate{  
    let xScale:CLLocationDegrees = 0.01  
    let yScale:CLLocationDegrees = 0.01  
    let span:MKCoordinateSpan = MKCoordinateSpanMake(yScale, xScale)  
    let region = MKCoordinateRegionMake(coordinate, span)  
    map.setRegion(region, animated: true)  
}
```

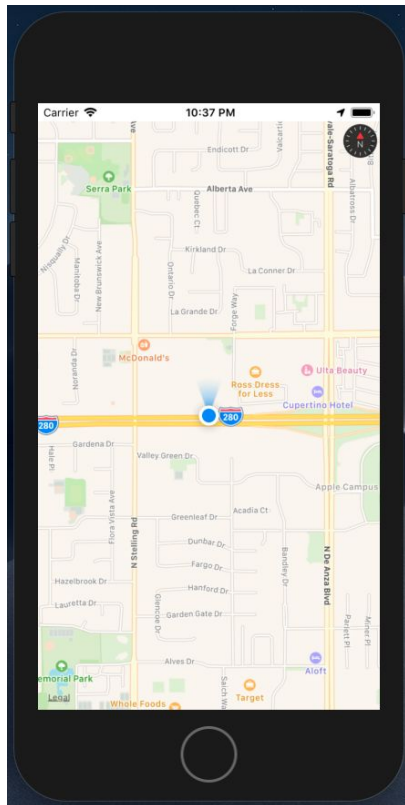
持續取得位置

```
locationManager = CLLocationManager()  
locationManager?.requestWhenInUseAuthorization()  
locationManager?.delegate = self
```

```
locationManager?.desiredAccuracy = kCLLocationAccuracyBest  
locationManager?.activityType = .automotiveNavigation  
locationManager?.startUpdatingLocation()
```

```
mapView.userTrackingMode = .followWithHeading
```

```
func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {  
    let coordinate = locations[0].coordinate  
}
```



Swift UI

Danny Shen

本節的主題

Swift UI 是什麼？

必要條件與限制

一些簡單的案例

與 Story Board 交互應用

為什麼要用 Swift UI

Swift UI 是什麼

WWDC 2019 發佈, 它是一種有效率的頁面建構方式

Swift 是一個新的程式庫 (對, 它不是一個UI設計方法, 它是程式庫), 它的位階類似 UIKit, 是最上層的東西, 所以 View 元件和 UIKit 不完全重覆

它的設計理念也和 UIKit 完全不同, 基本上不是以座標系統為主, 而是以容器 (Container) 系統為主, 它比較像是 Flutter 或 React Native 的設計理念

沒有分為 UI 檔和程式檔, 但仍可以預覽結果檔案

大量的使用 Closure 來設計你的 code, 在大多數的狀況下, 可以減少許多 UI 設計與拉頁面的程式碼

基本條件

限用於 iOS 13~ macOS 10.15~ (對, 舊版不能用),

和同一期的 tvOS 13, watchOS 6

開發工具 Xcode 11 以上

開發環境最好是 macOS 10.15 (10.14 仍可以編譯, 但只能在實體 iOS 13~執行),
而且沒有預覽

只能使用 Swift 不支援 Objective-C

來, 開始我們的 Swift UI 吧

Container

Container 是一種放置 UI 元件的容器，有各種不同的 Container，基本上把內容 (content) 放到 container 中，元件就會自動排列了

每個 View 只能有一個主要的 Container，但 Container 中，可以有其他的 Container 作巢狀排列

```
VStack {  
    Text("Hello, World!")  
    Text("This is my first SwiftUI")  
}
```

試試 自訂格式

```
VStack {  
    Text("AAA").font(.largeTitle)  
    Text("BBB")  
    HStack {  
        Text("AAA")  
        Text("BBB")  
    }  
    ZStack {  
        Text("AAA")  
        Text("BBB")  
    }  
}
```


搜尋官方文件

google 語法

【Text SwiftUI inurl:apple.com】

The screenshot shows the Apple Developer website's navigation bar with the 'SwiftUI' link highlighted by a red box. Below the navigation bar, the breadcrumb trail is 'Documentation > SwiftUI > Views and Controls > Text'. The main content area is titled 'Text' and describes it as 'A view that displays one or more lines of read-only text.' Below this, the 'Declaration' section shows the code snippet: `@frozen struct Text`. On the right side, a list of SDKs is shown, with 'SwiftUI' highlighted by a red box. At the bottom right, there is a 'On This Page' section with links to 'Declaration', 'Topics', 'Relationships', and 'See Also'. The 'Topics' section is currently empty.

tui/text

Developer Discover Design Develop Distribute Support Account

Documentation > SwiftUI > Views and Controls > Text Language: Swift API Changes: Sh

Structure

Text

A view that displays one or more lines of read-only text.

Declaration

```
@frozen struct Text
```

SDKs

- iOS 13.0+
- macOS 10.15+
- Mac Catalyst 13.0+
- tvOS 13.0+
- watchOS 6.0+
- Xcode 11.0+

Framework

SwiftUI

On This Page

- Declaration
- Topics
- Relationships
- See Also

Topics

UIKit 的 UIView vs. SwiftUI 的 View

是一種 class

通常是 UINavigationController 的 屬性

由事件驅動更新

由上層繼承下來層層的物件

頁面切換由多個 ViewController 組成

由Storyboard Xib 預覽

是一種 struct

通常由 SceneDelegate 來啟動

由資料驅動更新

由回傳值套用some View不是固定物件

頁面切換由不同 View 與資料切換

編譯 _Preview 實現 Canvas 預覽

TabView & 資料驅動(binding)

實作 TabBar View

@State 為狀態設定的變數，該變數直接加上 \$ 然後放在取值的地方，要顯示地方就直接放變數，當變數有變化時，其值就會影響到畫面重新繪製

加上 \$ 的操作，稱之為【綁定】(binding)

```
@State var activeTab: Int = 0
var body: some View {
    TabView(selection: $activeTab) {
        Text("in page \(activeTab)")
            .tabItem {
                Image(systemName: "list.bullet")
            }.tag(1)
        Text("in page \(activeTab)")
            .tabItem {
                Image(systemName: "list.bullet")
            }.tag(2)
    }
}
```

有參數的View

如果一個 @State 的 屬性, 沒有變預設值, 建構器就會只留下必需輸入屬性的建構器, 我們就可以在建立時, 輸入該項目的建構器

```
@State var displayString:String
var body: some View {
    VStack{
        Text(displayString)
        HStack{
            Text("請輸入: ")
            TextField("", text: $displayString)
        }
    }
}
```

製作一個調色盤.....

使用 Slider

使用 Color

學習使用各種排s版方式, 對齊等

Zstack, Spacer

自訂子 View

使用 binding 來改變顏色

