

# 推薦序

---

+ foreword

行動應用的程式設計就像它的手機載具一樣，發展有如光速般的日新月異。

對於廣大有需求的開發者而言，重點不在找到一本書教我們如何寫手機程式。而是找到一本書，讓我們可以跟上最新的手機平台發展；讓我們可以把手機最新的功能發揮到淋漓盡致，讓我們可以和其他的競爭者拉開距離。

這本《iOS 8 程式設計實戰》滿足了我們尋找最新技術的強烈需求，所有 iOS8 (iPhone 6) 上最新的功能 CloudKit、HealthKit、Extension 還有最新最紅的 iBeacon 應用，全都在這本書的涵蓋範圍之內。

這本書同時採用了讓我們最快速上手的範例式引導，在被一堆理論淹沒之前，馬上創造出屬於你的應用。從做中學，從做中啟發，快速的解決眼前的問題，正是這本書的精華。

在追逐技術的洪流中，我們往往得從一堆原文的資料裡學習跟上潮流的技能。雖然吃力，但也習慣了。在這種傳統學習的趨勢之下，竟然可以有一本這樣的中文參考資料，而其分量不亞於原文資訊，更是讓我在受用之餘，還多了一份小小的感動。

如果你已經泡好一杯咖啡準備向長夜挑戰，記得你的右手要帶上這本《iOS 8 程式設計實戰》。

游鴻志

宏碁智慧型手機事業群處長

# 序

+ preface

每年六月 Apple Worldwide Developers Conference (WWDC) 過後，就是跟時間賽跑的開始，除了要快速瞭解 Apple 到底在新版的 iOS 中加入了哪些新東西，也要瞭解他們移除了哪些舊東西，並且還要忍受手機裝了 beta 版後三不五時當掉的狀況，這是很忙碌的一段時間。

iOS8 是一個非常大的改版，增加了好多的亮點功能，大部分的亮點功能都已經寫在本書中（例如 Size Classes Tool、CloudKit、HealthKit、Extension... 等），少部分因為出版時間的壓力，只能暫時捨棄（例如 Handoff），我認為寫在書中的亮點功能都是非常重要的，非常有機會引領一波新的科技風潮與應用，所以如果在這本書中不放進去，恐怕有違讀者的期待，也讓這本書黯然失色。

感謝上一本 iOS7 讀者的捧場，讓該書在天瓏書局得到暢銷排行版第二名的佳績，當然這樣的成果也讓這本書在改版進入到 iOS8 時頗有壓力。這裡特別謝謝幾位朋友的幫忙，協助確認書中的程式碼在 iOS8 是有效的，他們的名字是江承翰、柯明倫與梁國基。

這本書在 Facebook 上有個粉絲團－「研蘋果」，建議讀者可以加入，有關這本書的一些訊息，包含了錯誤修訂、技術 meeting 或是研習訊息，都會直接公布在這裡，當然您也可以在這裡問問題。

最後，一樣祝福各位讀者，所有事情都能順心愉快。

朱克剛  
2014 初秋

chapter

# 09

## 座標與地圖

iPhone 或是 iPad 最重要的功能之一就是「定位」，也就是讓 iPhone 或 iPad 知道自己目前所在的位置。只有定位是不夠的，配合定位的功能，這些行動裝置還會再搭載一套地圖來顯示周邊的資訊。於是，一個圖資豐富的地圖加上知道目前所在的位置，就可衍生出很多很生活化的 App，例如當知道自己在哪裡後，透過地圖就可以得知周邊有什麼好吃好玩的景點以及想要去的地方該怎麼走。Google 是一間將地圖電子化並且發揚光大的公司，上面各式各樣的圖資以及街景，讓使用者在一個陌生的地方不會迷路。在 iOS 5 以前的版本搭載的就是 Google 地圖。但是在 iOS 6 開始，Apple 將地圖改用自己的地圖了。新版的地圖除了畫面明顯不一樣外，地圖服務也可以跟其他的 App 程式整合的更密切。在過去，若 App 想要使用地圖服務，必須在 App 中內嵌一個地圖元件，但 iOS 6 之後，我們的 App 可以直接呼叫系統內建的地圖服務，並且要求他作一些事情（例如導航），而我們的 App 完全不用事先插入一個地圖元件才能使用地圖的功能。

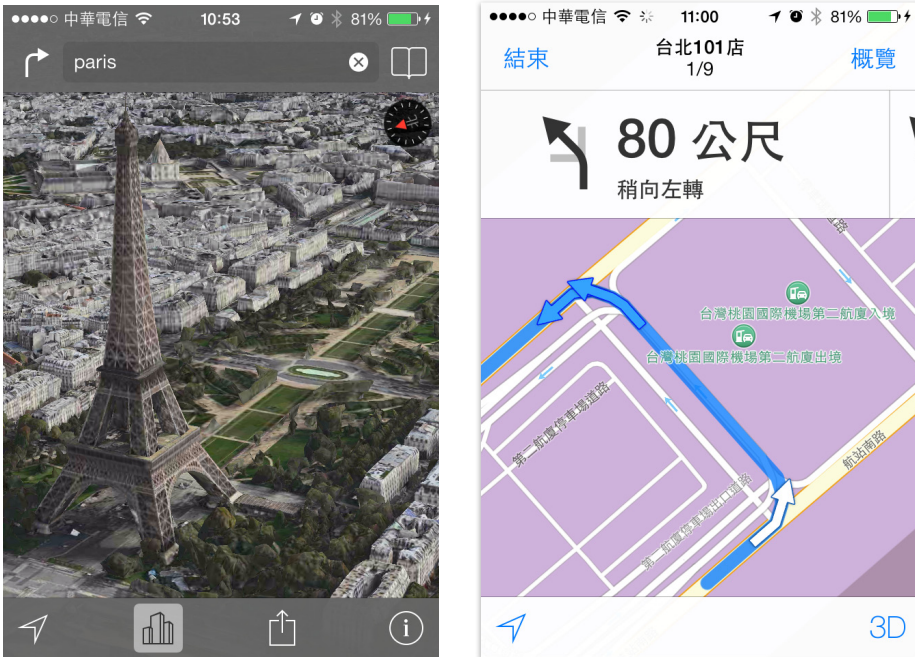


想要使用地圖服務有兩種選擇，一種是在 App 中插入一個地圖元件（Map View 元件），然後透過程式碼來操作這個元件，另一種就是直接呼叫系統的地圖服務。地圖元件所使用的類別為 MKMapView，該類別實作了一個地圖頁面供使用者操作，地圖類型有標準，衛星與混合三種模式。標準是以街道、建築物為主的地圖，不具自然景觀，一切以符號、線條來呈現；衛星則是衛星空照圖；混合則是將標準地圖與衛星空照圖疊在一起顯示。地圖元件本身就具備了捲軸功能，讓使用者可以透過捲動的方式看到原本在螢幕範圍之外的地方。

電子地圖必須提供一個非常重要的功能，就是標示特定位置。例如當我們查詢某個餐廳的時候，如果地圖只是顯示一個範圍而沒有標示出該餐廳在哪裡，這樣只是增加使用者看地圖時的困擾，因此，好的設計就是要在地圖上準確的標示出餐廳所在的位置，這樣才能讓使用者一看就知道「餐廳就在這裡」。在地圖上標示出特定的符號或是區域，使用的是 annotation 與 overlay 功能。Annotation 與 overlay 的差異在於 annotation 是針對單一座標點進行標記，例如使用者現在的位置、特定的地址或某個經緯度座標。而 overlay 是針對一個連續或不連續的範圍進行標示，例如標示國家或是湖泊的範圍、道路交通狀況或是導航路線等。舉例來說，如果我們想要在地圖上框出一個矩形區域，就必須使用 overlay。iOS 預設的 annotation 標記長相就如同大頭針，有三種顏色可以選擇，分別是紅色、綠色與藍色。除了預設的標記外，我們也可以自訂標記，自訂標記使用的是一張圖片。例如我們可以在每個國家的首都上插上該國的國旗，或是美食地圖 App 上的餐廳的位置就以該餐廳的特色食物圖片來替代大頭針，這樣可以讓使用者觀看時一目了然。



新版的 iOS 地圖在某些區域提供了 3D 立體地圖，切換到 3D 地圖後，畫面極其驚艷，非常地漂亮，只是目前還不是全世界的都市都有 3D 地圖，台灣也還沒有，有興趣的讀者可以搜尋巴黎並切換到衛星畫面看看。新版地圖也提供了導航功能，我們很容易的可以寫一個車用導航系統，唯一的缺點就是地圖是線上地圖，手機必須一直連線在網路上才行。



使用地圖服務時 Xcode 要載入 Core Location 框架，這個框架除了提供地圖服務外也提供了定位服務，也就是說，如果想要知道目前所在位置的經緯度座標或是高度，Core Location 提供了這樣的資訊。除此之外，Core Location 還提供了目前的方位，想要知道目前是面向北方還是東方，Core Location 就可以告訴我們，想要寫指南針 App 就要靠 Core Location 了。

iOS 行動裝置透過三種方式取得裝置所在的地理位置資訊（即經緯度座標與高度）：（一）GPS；（二）WiFi 基地台定位；（三）手機基地台定位。其中最精準的當屬 GPS 定位，誤差最大的則屬於手機基地台定位。GPS 雖然最精確，但是在室內或是地下室等收不到衛星訊號的地方，GPS 就無用武之地。但是我們會發現，在室內或地下室，甚至有些沒有 GPS 功能的行動裝置（例如

WiFi 版的 iPad），依然可以在地圖中看到自己所在的位置，這種神奇的定位方式就是使用 WiFi 基地台定位，或是手機基地台定位。

WiFi 基地台定位的原理其實很有趣，在蘋果公司或是 Google 公司的資料庫中有一大堆 WiFi 基地台所在的位置資料庫，所以當你連上某個 WiFi 基地台，iOS 只要跟他的資料庫比對就知道你現在的位置了。問題是 Apple 或是 Google 怎麼知道全世界每個 WiFi 基地台的位置呢？當 Google 街景車在大街小巷巡迴的時候，其實也就順便收集沿路 WiFi 基地台的資訊。或是某個使用者開啟了 GPS 裝置並且周圍有 WiFi 基地台，那麼這個 WiFi 基地台所在的位置也同時被收錄到 WiFi 基地台資料庫中，下一次有人連上這個基地台時，iOS 自然知道這個人所在的位置了。雖然這種定位方式沒有 GPS 來的精準，但是在室內或是沒有 GPS 功能的情況下，這種定位方式卻非常的巧妙與有效，2011 年底 Google Map 提供了一些機場的室內定位功能，讓旅行者在這些機場裡面都可以掌握自己所在的位置，如果讀者下次去機場的時候（例如成田機場、舊金山國際機場），可以開啟 Google Map 看看，是否可以在該機場裡面找到自己的位置。至於手機基地台定位也是同樣的原理，只不過手機基地台的訊號範圍比 WiFi 範圍來的大，因此當行動裝置使用手機基地台來定位時，誤差可能會高達 1~2 公里。

雖然 iOS 行動裝置有三種定位方式，但是程式設計師不用去思考現在的環境到底該用哪一種比較好，這部分 iOS 自動幫我們處理掉了。程式設計師只要管何時開啟定位，然後接收座標，不用時再關掉定位，這樣就好了。

# 9-1

難易度



## 取得目前所在位置的經緯度座標與高度

先備知識：無

Framework：CoreLocation.framework

iOS 透過幾種不同的方式來得到經緯度座標與高度，有些方式誤差很小（例如 GPS），有些方式誤差則較大（例如基地台），iOS 會根據當時的狀況自動選擇最精準的方式提供經緯度座標與高度等資料，程式設計師不需要煩惱是否要根據不同的狀況撰寫不同的程式碼，因為程式碼都是同樣的寫法。

特別要注意的是，現在如果要取得使用者目前所在的經緯度座標等資料，必須要在 Info.plist 檔案中加入：`NSLocationAlwaysUsageDescription` 或是 `NSLocationWhenInUseUsageDescription` 鍵值，前者表示不論 App 在前景或是背景都會使用 GPS 取得座標資料，後者則是需要的時候（通常是前景）才會開啟 GPS。這兩個鍵值所對應的方法分別為 `requestAlwaysAuthorization` 與 `requestWhenInUseAuthorization`，他們是 `CLLocationManager` 類別的成員函數。



### 步驟與說明

- 1 建立 Single View Application 專案。
- 2 將 CoreLocation.framework 加到專案中。
- 3 在 ViewController.h 中匯入 CoreLocation.h 標頭檔，並且讓此類別符合 CLLocationManagerDelegate 協定的規範。除此之外，再宣告一個型態為 CLLocationManager 的變數，姑且稱為 location。location 變數專門負責用來開關行動裝置的座標接收。

```
#import <UIKit/UIKit.h>
#import <CoreLocation/CoreLocation.h>

@interface ViewController : UIViewController <CLLocationManagerDelegate>
{
    CLLocationManager *location;
}

@end
```

**4** 開啟 `ViewController.m`，在 `viewDidLoad` 方法中先初始化 `location` 變數，並且告訴系統當行動裝置的地理座標更新時，要通知哪一個物件去做處理，一般來說通知自己就好，也就是符合 `CLLocationManagerDelegate` 協定的那一個類別。最後記得要開啟地理座標更新，否則程式會收不到新的座標。

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    location = [[CLLocationManager alloc] init];
    location.delegate = self;
    // 詢問是否要給 APP 有定位功能權限
    [location requestWhenInUseAuthorization];
    // 開啟計算目前行動裝置所在位置的功能
    [location startUpdatingLocation];
}
```

**5** 由於這個 App 要加入定位功能，因此需在 `info.plist` 加入一個 `NSLocationWhenInUseUsageDescription` 鍵值，用來跟使用者說明 APP 為何需要使用到定位功能的描述（可為空值）。

| Key                                    | Type       | Value                                      |
|--|------------|--|
| ▼ Information Property List            | Dictionary | (15 items)                                 |
| Localization native development region | String     | en   |
| NSLocationWhenInUseUsageDescription    | String     | Hello                                      |
| Executable file                        | String     | \$(EXECUTABLE_NAME)                        |
| Bundle identifier                      | String     | com.cck.\$(PRODUCT_NAME:rfc1034identifier) |
| InfoDictionary version                 | String     | 6.0  |
| Bundle name                            | String     | \$(PRODUCT_NAME)                           |

**6** 根據 `CLLocationManagerDelegate` 協定，我們要在 `ViewController.m` 中實做 `locationManager:didUpdateToLocation:` 方法。這個方法主要的功能就是接收新的地理座標資訊，包含了經度、緯度與高度。

```
-(void)locationManager:(CLLocationManager *)manager
    didUpdateLocations:(NSArray *)locations
{
    CLLocation *c = [locations objectAtIndex:0];
    NSLog(@"緯度:%f, 經度:%f, 高度:%f", c.coordinate.latitude,
        c.coordinate.longitude, c.altitude);
}
```



**7** 由於開啟行動裝置的座標更新功能會消耗系統電力資源，因此不用的時候要記得關掉它，在這裡我們就選擇在 `viewDidDisappear:` 中關閉即可。當 App 進入背景時，iOS 也會自動幫我們關掉。

```
- (void) viewDidDisappear: (BOOL) animated
{
    // 關閉計算目前行動裝置所在位置的功能
    [location stopUpdatingLocation];
}
```

**8** 執行看看。

```
CLLocation[6283:907] 緯度：23.876177, 經度：120.493705, 高度：41.000000
CLLocation[6283:907] 緯度：23.876177, 經度：120.493705, 高度：41.000000
CLLocation[6283:907] 緯度：23.879383, 經度：120.492780, 高度：41.000000
```

## 9-2

難易度



## 得知目前所面向的地理方向

先備知識：無 Framework：CoreLocation.framework

到一個陌生的環境，我們常常會發生東南西北搞不清楚方向的窘境，明明要朝東邊走，但事實上卻是往西邊前進，反而越走越遠。除了看星星看太陽之外，手上的 iPhone / iPad 就可以幫我們偵測目前所面對的方向，到底是朝向東邊還是朝向西邊。開啟 iOS 行動裝置偵測方向的功能後，程式會不斷的收到一個浮點數，數字範圍從 0 到 359 之間，透過這個數字，就可以知道目前所面對的方向，因為正北方為 0，東方為 90，南方為 180，西方為 270。

### 步驟與說明

- 1 建立 Single View Application 專案。
- 2 將 CoreLocation.framework 加到專案中。
- 3 在 ViewController.h 中匯入 CoreLocation.h 標頭檔，並且讓此類別符合 CLLocationManagerDelegate 協定的規範。除此之外，再宣告一個型態為 CLLocationManager 的變數，這個變數專門負責用來開關行動裝置的地理方位訊息接收。

```
#import <UIKit/UIKit.h>
#import <CoreLocation/CoreLocation.h>

@interface ViewController : UIViewController <CLLocationManagerDelegate>
{
    CLLocationManager *location;
}

@end
```

- 4 開啟 ViewController.m，在 viewDidLoad 方法中先初始化 location 變數，並告訴系統當行動裝置所面對的方向更新時，要通知哪一個物件去做處理，一般來說通知自己就好，也就是符合 CLLocationManagerDelegate 協定的那一個類別。最後記得要開啟地理方向數值更新，否則程式會收不到新的方向資料。

01  
02  
03  
04  
05  
06  
07  
08  
**09**

10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
A  
B  
C

座標與地圖

```

- (void) viewDidLoad
{
    [super viewDidLoad];

    location = [[CLLocationManager alloc] init];
    location.delegate = self;
    // 開啟偵測目前行動裝置面對方向的功能
    [location startUpdatingHeading];
}

```

**5** 根據 `CLLocationManagerDelegate` 協定，我們要在 `ViewController.m` 中實做 `locationManager:didUpdateHeading:` 方法。在這個方法中，由傳進來的參數 `newHeading` 中的 `magneticHeading` 屬性可以得到目前行動裝置所面對的方向（正北方是 0）。除此之外，屬性 `headingAccuracy` 表示了方向的精確度。如果數值越小，就代表越精確，但如果是負的，表示目前行動裝置需要做磁性校準或是目前周圍有嚴重的磁性物質干擾。校準方式依照 iPhone 內建的指南針 App 所建議的方式是繞 8 字形來進行校準。

```

- (void) locationManager: (CLLocationManager *) manager
    didUpdateHeading: (CLHeading *) newHeading
{
    if (newHeading.headingAccuracy < 0) {
        NSLog(@"請進行校準或遠離磁性干擾源");
        return;
    }

    NSLog(@"目前面對的方位為：%f", newHeading.magneticHeading);
}

```

**6** 不用時記得關掉此項功能，以免不斷消耗系統電力資源。這裡我們就在 `viewDidDisappear:` 中關閉即可。當 App 進入背景時，iOS 也會自動幫我們關掉。

```

- (void) viewDidDisappear: (BOOL) animated
{
    // 不用時要關閉更新數值功能以節省電力
    [location stopUpdatingHeading];
}

```

**7** 執行看看。

```

GeoHeading[6337:907] 目前面對的方位為：52.307968
GeoHeading[6337:907] 目前面對的方位為：51.307968
GeoHeading[6337:907] 目前面對的方位為：50.307968

```

## 9-3

難易度



# 在地圖上釘一根大頭針

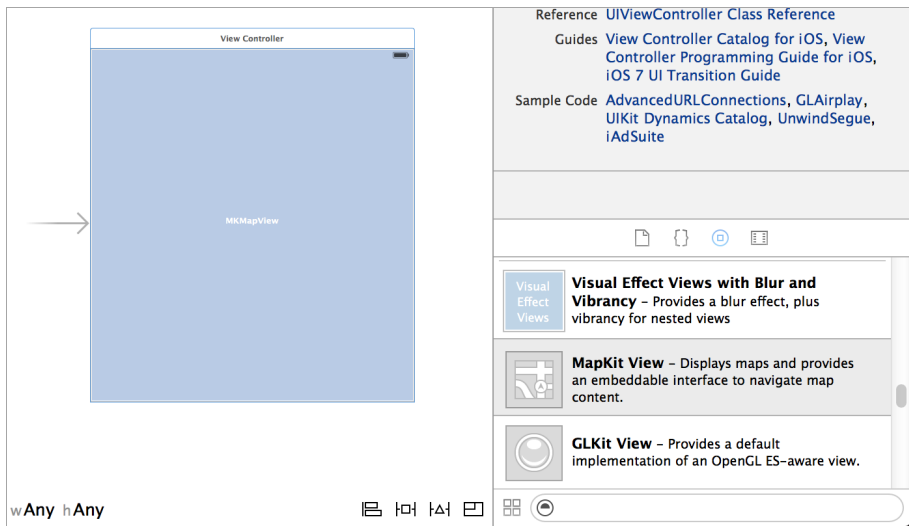
先備知識：無

Framework：MapKit.framework、CoreLocation.framework

我們在很多電影看過這樣的場景，一幅掛在牆壁上的世界地圖，上面釘滿了大頭針，這些大頭針可能代表了這幅地圖的主人去過哪些國家。同樣的場景也可以出現在 iOS 的地圖中。如果要在 iOS 的地圖上標示一個特別的地方，我們就產生一個「大頭針物件」，然後釘在某個特定的座標上。

### 步驟與說明

- 1 建立 Single View Application 專案。
- 2 將 MapKit.framework 與 CoreLocation.framework 加到專案中。
- 3 從元件庫中拖放一個 Map View 元件到 Storyboard 上的 View Controller。



01

02

03

04

05

06

07

08

09

10

11

12

13

14

15

16

17

18

19

20

21

22

23

A

B

C

座標與地圖

**4** 開啟 Assistant editor 視窗（選單 View → Assistant Editor → Show Assistant Editor），在 ViewController.h 中匯入 MapKit.h 標頭檔，並且設定 Map View 的 IBOutlet 變數名稱為 myMap。

```
#import <UIKit/UIKit.h>
#import <MapKit/MapKit.h>

@interface ViewController : UIViewController
@property (weak, nonatomic) IBOutlet MKMapView *myMap;
@end
```

**5** 開啟 ViewController.m，我們打算在台北與東京分別釘上兩根大頭針。如果沒有要特別的變化，例如改變大頭針顏色或是變換大頭針圖片，就使用 MKPointAnnotation 類別即可，這是釘一根大頭針最簡單的方式。每個 MKPointAnnotation 類別的實體可以設定座標、標題與副標題，其中座標一定要設定，標題與副標題忽略它們也沒關係，只是當使用者去點大頭針的時候，不會出現一些文字說明而已。設定完成後呼叫 MKMapView 的 addAnnotation: 方法將每一根設定好的大頭針加進地圖就完成了。如果要加入很多的大頭針，也可以呼叫 addAnnotations: 方法一次加進去比較有效率。

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    MKPointAnnotation *point;

    // 設定在台北的大頭針
    point = [[MKPointAnnotation alloc] init];
    point.coordinate = CLLocationCoordinate2DMake(25.0335, 121.5651);
    point.title = @"台北市";
    point.subtitle = @"中華民國首都";
    [self.myMap addAnnotation:point];

    // 設定在東京的大頭針
    point = [[MKPointAnnotation alloc] init];
    point.coordinate = CLLocationCoordinate2DMake(35.6997, 139.6989);
    point.title = @"東京市";
    point.subtitle = @"日本的首都";
    [self.myMap addAnnotation:point];
}
```

**6** 為了讓地圖一開始的時候就移到特定的位置去，我們可以透過 `centerCoordinate` 屬性讓某個座標顯示在地圖的中間。這段程式碼可以寫在 `ViewController.m` 的 `viewDidAppear:` 方法內，讓地圖出現在螢幕前就先移到正確的位置去。

```
-(void) viewDidAppear: (BOOL) animated
{
    // 台北市座標
    self.myMap.centerCoordinate = CLLocationCoordinate2DMake(25.0335,
                                                            121.5651);
}
```

**7** 執行看看。

