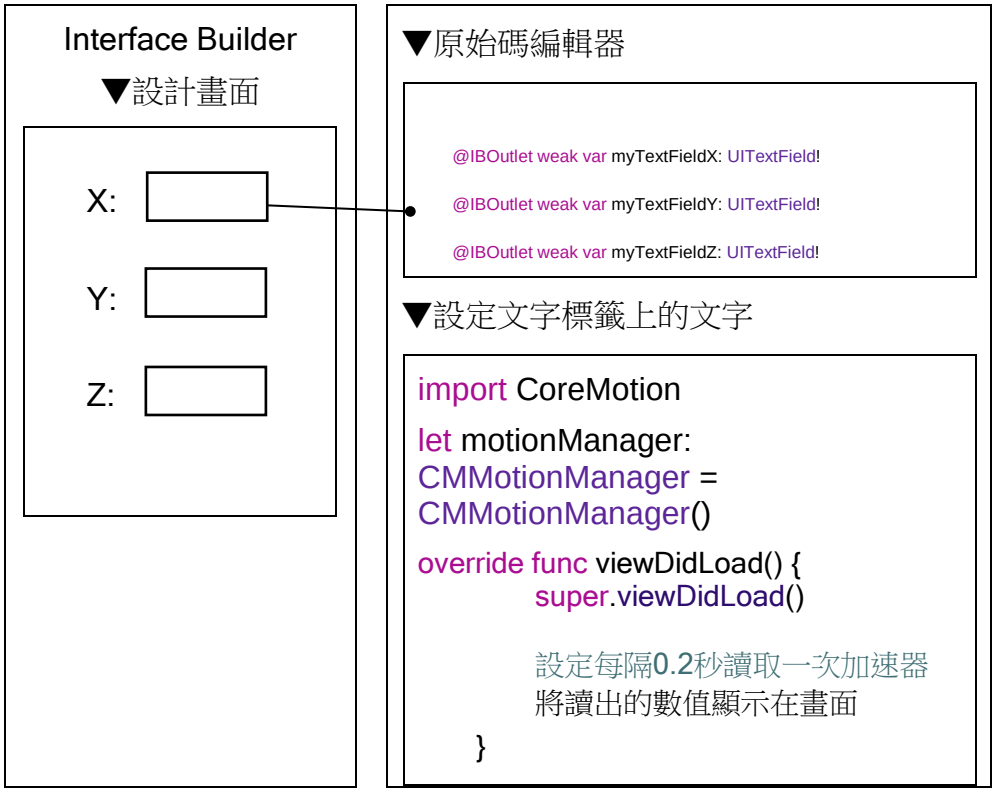
CHAPTER 7-2

CoreMotion: 讀取加速感應器

在手機中顯示加速計的數值

學習概念：

1. 首先用 IB 建立【文字標籤】與【文字欄位】。
2. 接著將【文字欄位】與【程式碼連結】。
3. 最後在實作相關程式，載入 CoreMotion 函式庫
4. 在處理畫面載入後所觸發的事件，也就是【ViewDidLoad】方法。

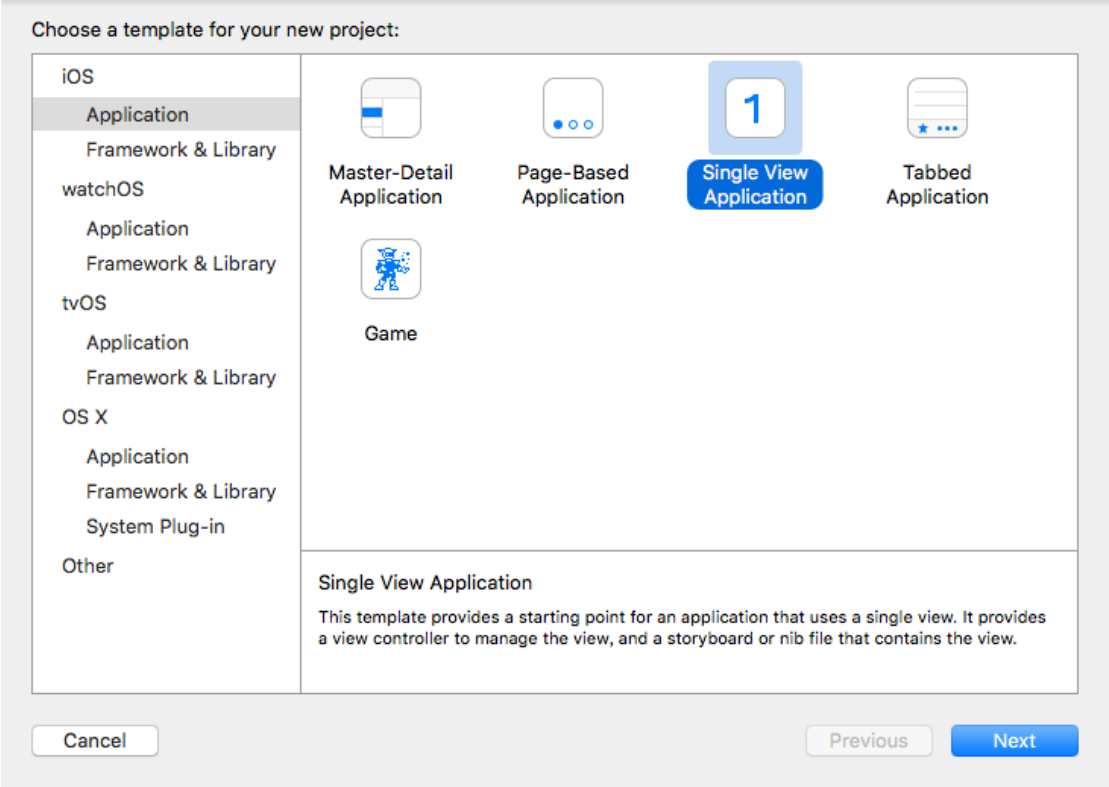


【執行結果】

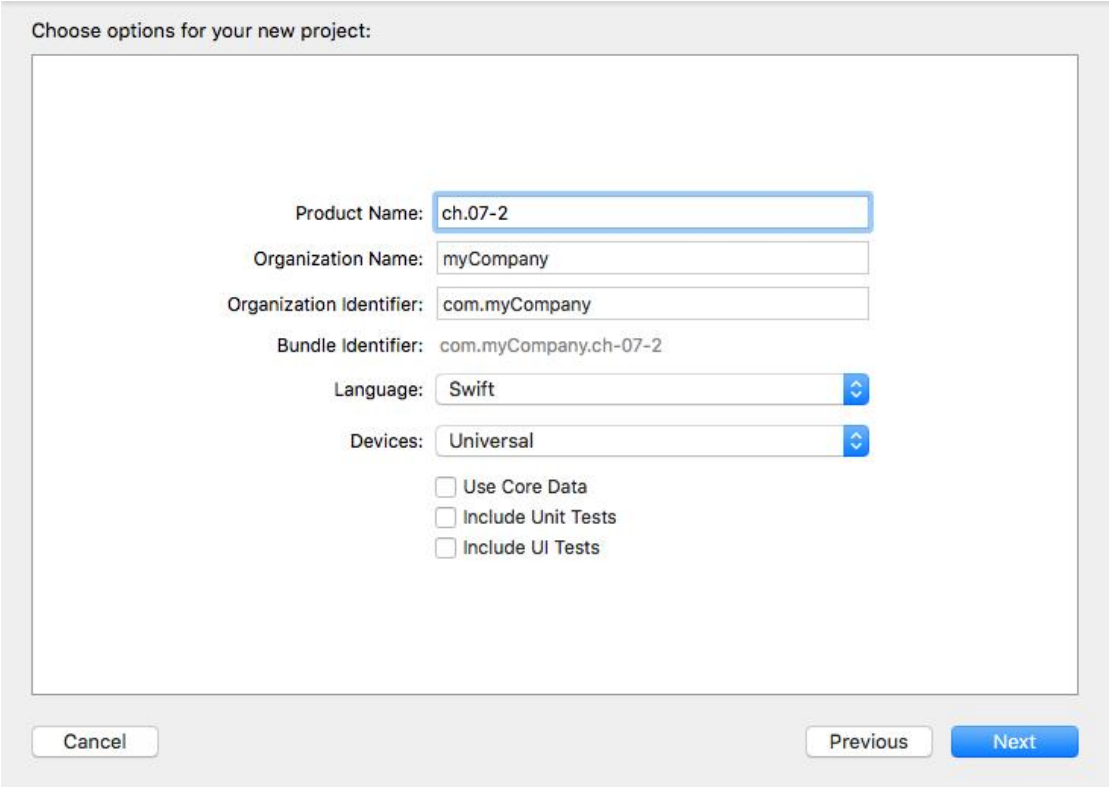
當 App 執行後，會隨著手機的移動【文字欄位】上顯示數值。

Step.1

開啟 xcode 時會出現的畫面，點選 iOS 下的【Application】，接著右視窗選擇【Single View Application】，點選【Next】選項後進入設定的基本視窗。

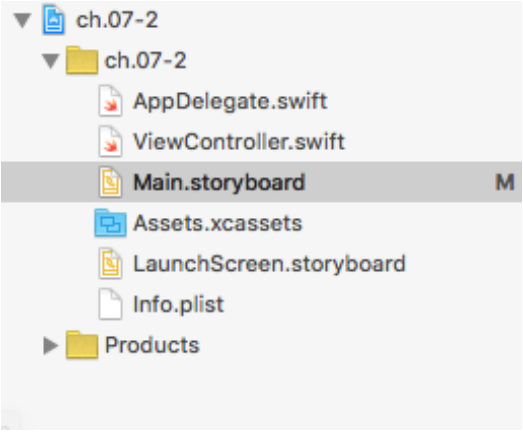


檔名及名稱設定，請將【Product Name】設定為 ch.07-2

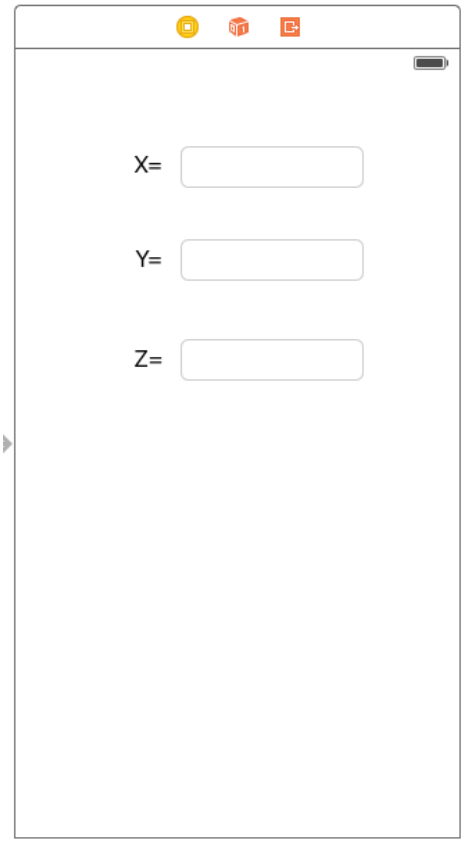


Step.2

選取【Main.storboard】

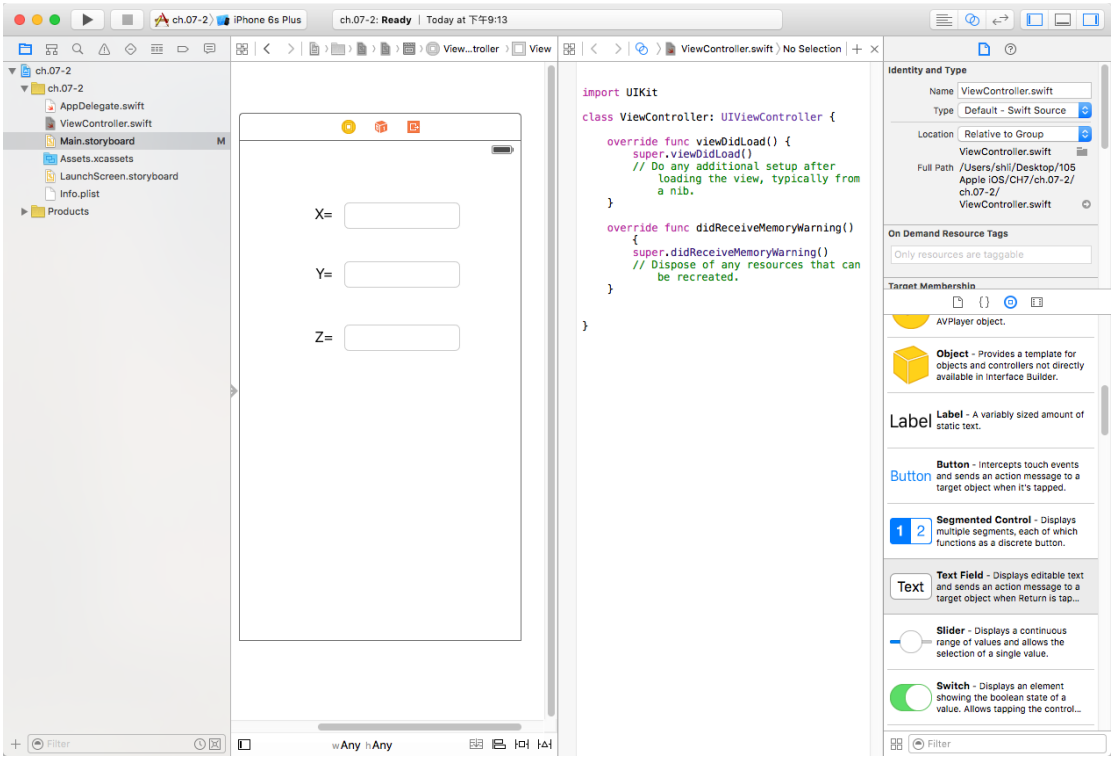


從【物件庫】中拖曳【Label】、【Text Field】到畫面中。



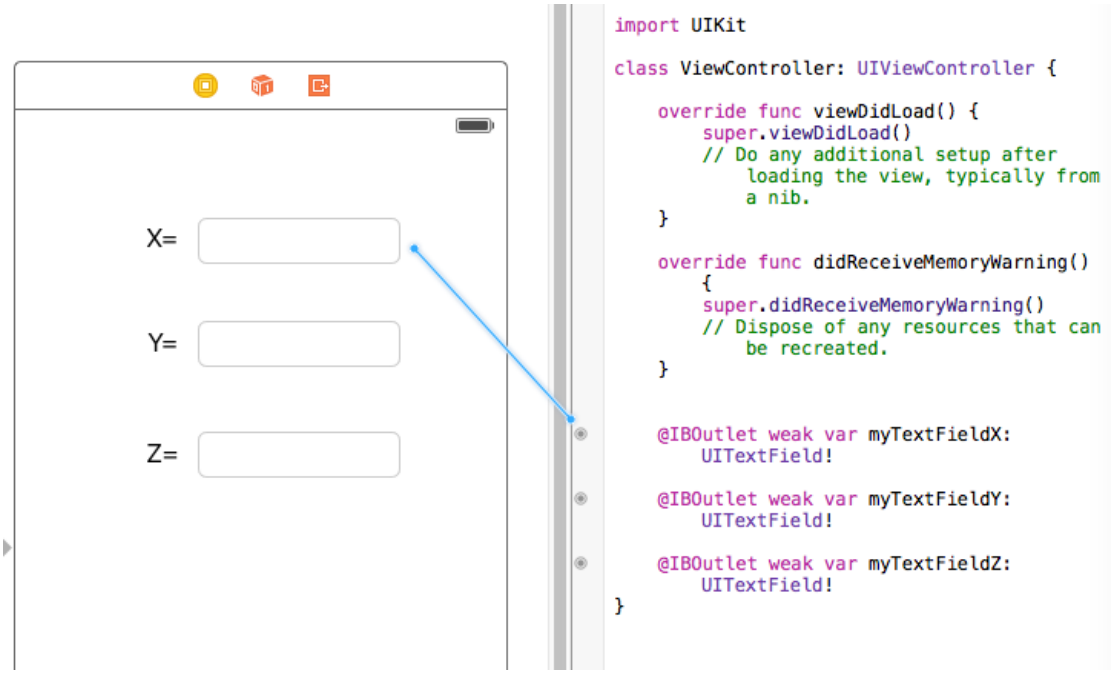
Step.3.

將編輯畫面分割為四部分，以方便連結元件與程式



Step.4

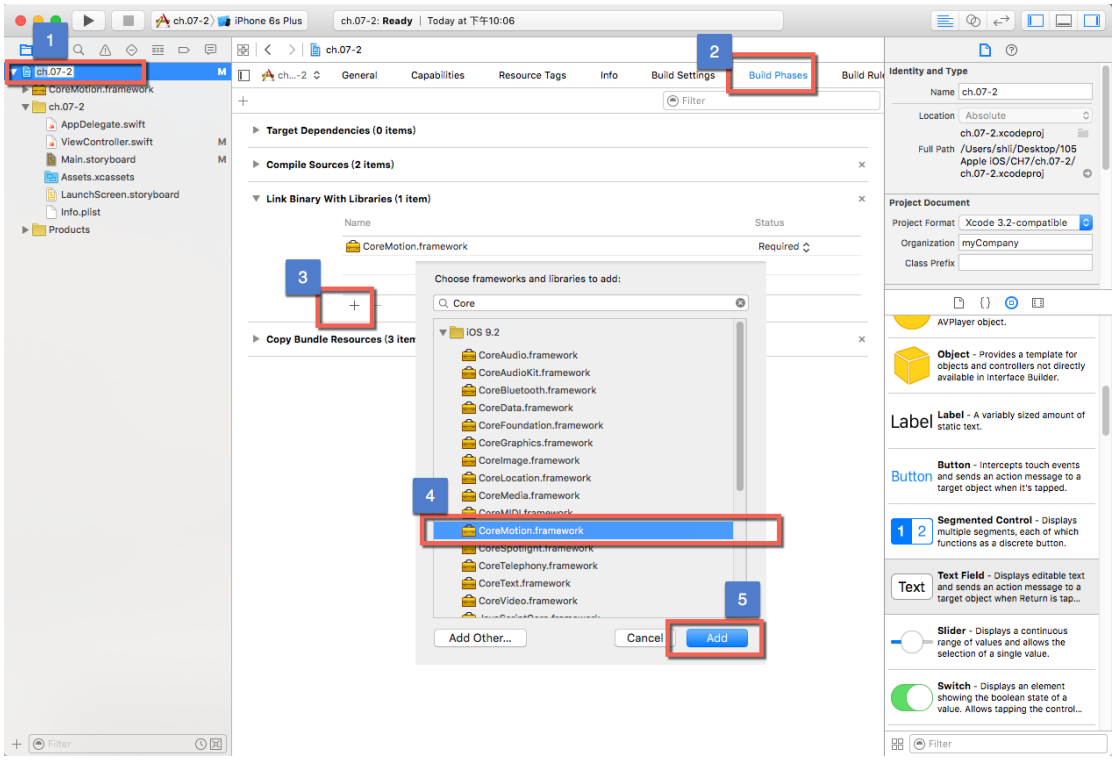
(1) 將【文字欄位】和【名稱連結】，分別命名 myTextFieldX, myTextFieldY, myTextFieldZ



Step.5 。

(1) 點選右上方專案名稱，以開啟專案的基本設定。
(2) 切換到 Build Phase
(3) 下拉 Link Binary With Libraries，點選『+』以新增函式。
(4) 選擇 CoreMotion.Framework
(5) 點選『Add』以加入此函式庫。

，



Step.6 。

(1) 於程式中 加入引用 CoreMotion 函式庫。

```
import CoreMotion
```

(2) 建立加速感應器的物件

```
let motionManager: CMMotionManager = CMMotionManager()
```

(3) 設定偵測的間隔時間為 0.2 秒。

```
motionManager.accelerometerUpdateInterval = 0.2
```

(4) 偵測加速度狀態，並將個別方向的加速度顯示在文字欄位中。

```
func startAccelerometerUpdatesToQueue(_ queue:
NSOperationQueue, withHandler handler: CMAccelerometerHandler)
```

用來開始紀錄加速度的值

方向感應器(accelerometer)是藉由感應某個方向的慣性力大小來衡量其加速度與重力。藉由裝置內的方向感應器可以偵測三度空間中的移動或

重力。因此使用者可以利用方向感應器得知裝置目前的擺放方式(例如：橫放、倒放或背面朝上)。

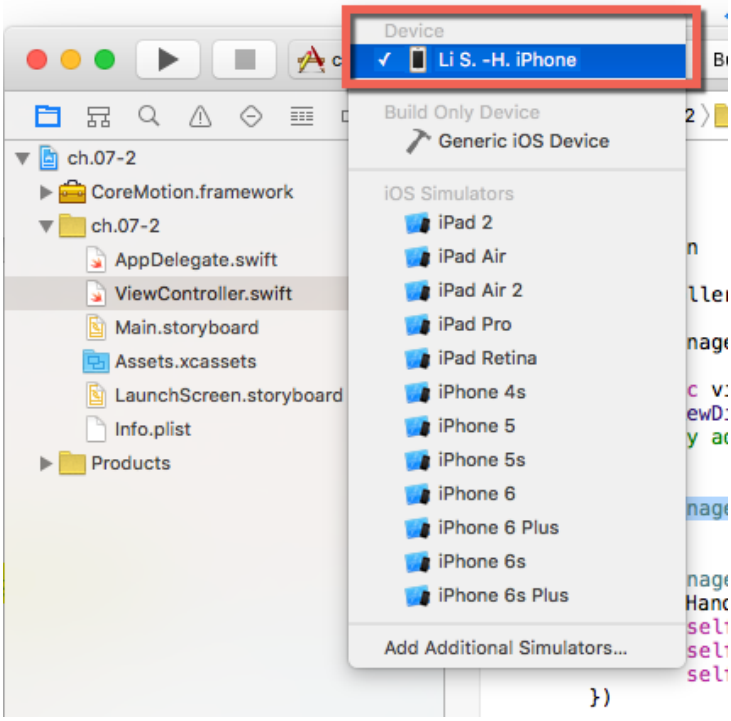
acceleration 屬性
acceleration.x 水平方向的加速度 (左:-1.0, 右：1.0)
acceleration.y 垂直方向的加速度 (下:-1.0, 上：1.0)
acceleration.z 前後方向的加速度 (後:-1.0, 前：1.0)

```
motionManager.startAccelerometerUpdatesToQueue
(NSOperationQueue.mainQueue(), withHandler:    {
(data, error) in
self.myTextFieldX.text = String(format: "%.2f", data!.acceleration.x)
self.myTextFieldY.text = String(format: "%.2f", data!.acceleration.y)
self.myTextFieldZ.text = String(format: "%.2f", data!.acceleration.z)
})
```

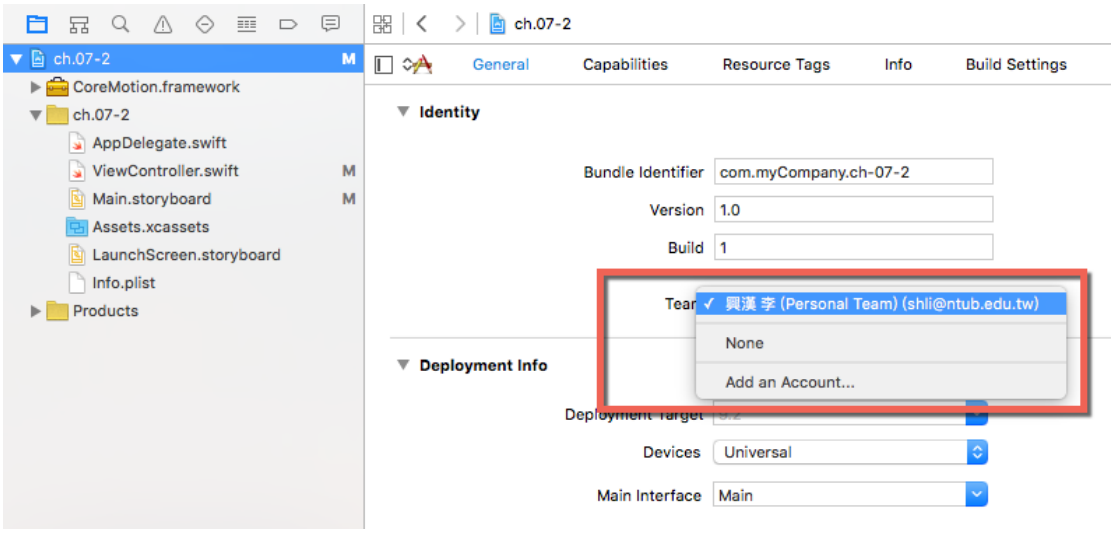
```swift
import UIKit
import CoreMotion
class ViewController: UIViewController {

    let motionManager: CMMotionManager = CMMotionManager()

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.

        motionManager.accelerometerUpdateInterval = 0.2

        motionManager.startAccelerometerUpdatesToQueue(NSOperationQueue.mainQueue(),
            withHandler:  { (data, error) in
                self.myTextFieldX.text = String(format: "%.2f", data!.acceleration.x)
                self.myTextFieldY.text = String(format: "%.2f", data!.acceleration.y)
                self.myTextFieldZ.text = String(format: "%.2f", data!.acceleration.z)
        })
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }


    @IBOutlet weak var myTextFieldX: UITextField!

    @IBOutlet weak var myTextFieldY: UITextField!

    @IBOutlet weak var myTextFieldZ: UITextField!


}
```

Step.7 。

(1) 執行程式前，請先選擇實體手機



(2) 若手機裝置出現安裝錯誤，則需要先確定是否已經將手機登入的 Apple ID 以經加入到開發者帳號中



活用【CoreMotion】進階

使用 CoreMotion 其他的偵測裝置讀取數值

```
//加速計　　讀取 X Y Z 軸的傾斜狀況
motionManager.startAccelerometerUpdatesToQueue(NSOperationQueue.mainQueue(), withHandler:  { (data, error) in
                //輸出數值
        })
```

acceleration 屬性

acceleration.x	水平方向的加速度
acceleration.y	垂直方向的加速度
acceleration.z	前後方向的加速度

```
//陀螺儀　　讀取 X Y Z 軸的旋轉狀況
motionManager.startGyroUpdatesToQueue(NSOperationQueue.mainQueue(), withHandler:  { (data, error) in
                //輸出數值
        })
```

Gyro 屬性

rotationRate.x	水平方向的旋轉速率
rotationRate.y	垂直方向的旋轉速率
rotationRate.z	前後方向的旋轉速率

```
//磁力儀　　讀取 X Y Z 軸的磁感應強度
motionManager.startMagnetometerUpdatesToQueue(NSOperationQueue.mainQueue(), withHandler:  { (data, error) in
                //輸出數值
        })
```

Magnetometer 屬性

magneticField.x	水平方向的磁力值
magneticField.y	垂直方向的磁力值
magneticField.z	前後方向的磁力值

```
//裝置移動     讀取  Pitch Roll Yaw
motionManager.startDeviceMotionUpdatesToQueue(NSOperationQueue.mainQueue(), withHandler:    { (data, error) in
                    //輸出數值
            })
```

DeviceMotion 屬性

attitude.pitch 俯仰弧度值

attitude.roll 滾動弧度值

attitude.yaw 左右搖擺弧度值

```swift
import UIKit

import CoreMotion

class ViewController: UIViewController {

    let motionManager: CMMotionManager = CMMotionManager()


    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.

        motionManager.accelerometerUpdateInterval = 0.2

        //加速計   讀取 ＸＹＺ軸的傾斜狀況
        motionManager.startAccelerometerUpdatesToQueue(NSOperationQueue.mainQueue(),
            withHandler:  { (data, error) in
            self.x1.text = String(format: "%.2f", data!.acceleration.x)
            self.y1.text = String(format: "%.2f", data!.acceleration.y)
            self.z1.text = String(format: "%.2f", data!.acceleration.z)
        })

        //陀螺儀   讀取 ＸＹＺ軸的旋轉狀況
        motionManager.startGyroUpdatesToQueue(NSOperationQueue.mainQueue(), withHandler:
            { (data, error) in
            self.x2.text = String(format: "%.2f", data!.rotationRate.x)
            self.y2.text = String(format: "%.2f", data!.rotationRate.y)
            self.z2.text = String(format: "%.2f", data!.rotationRate.z)
        })

        //磁力儀   讀取 ＸＹＺ軸的磁感應強度
        motionManager.startMagnetometerUpdatesToQueue(NSOperationQueue.mainQueue(),
            withHandler:  { (data, error) in
            self.x3.text = String(format: "%.2f", data!.magneticField.x)
            self.y3.text = String(format: "%.2f", data!.magneticField.y)
            self.z3.text = String(format: "%.2f", data!.magneticField.z)
        })

        //裝置移動   讀取 Pitch Roll Yaw
        motionManager.startDeviceMotionUpdatesToQueue(NSOperationQueue.mainQueue(),
            withHandler:  { (data, error) in
            self.pitch.text = String(format: "%.2f", data!.attitude.pitch)
            self.roll.text = String(format: "%.2f", data!.attitude.roll)
            self.yaw.text = String(format: "%.2f", data!.attitude.yaw)
        })

    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }


    @IBOutlet weak var x1: UITextField!
    @IBOutlet weak var y1: UITextField!
    @IBOutlet weak var z1: UITextField!

    @IBOutlet weak var x2: UITextField!
    @IBOutlet weak var y2: UITextField!
    @IBOutlet weak var z2: UITextField!

    @IBOutlet weak var x3: UITextField!
    @IBOutlet weak var y3: UITextField!
    @IBOutlet weak var z3: UITextField!

    @IBOutlet weak var pitch: UITextField!
    @IBOutlet weak var roll: UITextField!
    @IBOutlet weak var yaw: UITextField!

}
```

自 我 練 習

實作執行後結果：

設計一個球滾動的遊戲，球會隨著重力的位置而移動，因此，當手機改變方向時，球會在視窗內滾動。