

Airyzone SDK 版本：[v0.0712.1](#)

支援 iOS 版本：[ios 9.0](#) 以上

支援程式語言：[Swift 3.0](#) 以上, [Objective-C](#)

一、設定 Project

1. BLESDK.framework 置入 [Project](#)，並且加入至 [Embedded Binaries](#)：



2. 加入相依的 Frameworks，或使用 Cocoapods

a. AFNetworking

b. Alamofire

c. CoreBluetooth

d. iOSSDFULibrary

2.1 Pod File Content

```
source 'https://github.com/CocoaPods/Specs.git'

platform :ios, '9.0'

use_frameworks!

workspace 'YourWorkspace'

target 'YourTargets' do

  xcodeproj 'YourProject.xcodeproj'

  pod 'AFNetworking', '~> 3.0'

  pod 'Alamofire', '~> 4.3'

  pod 'iOSSDFULibrary', '~> 3.0'

end
```

二、開始使用 SDK

1.初始化 BeaconAPI

```
_ = BeaconAPI.sharedInstance.initAPI(APPID: AIRYZON_APP_ID, APPKey: AIRYZON_API_KEY)
```

APPID : 申請之 APP ID

APPKey : 申請之 APP Key

2.申請使用者

```
public func RegisterUser( UserID userID:String, FireBaseKey firebaseKey:String, completeHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID：使用者帳號

FireBaseKey：Firebase 的 api key，可以為 nil

建立 **ApiCompletionHandler** 實例

```
BeaconAPI.sharedInstance.RegisterUser(UserID: userID, FirebaseKey:yourFireBaseKey,  
completeHandler: { (response) -> Void in  
    if let dic:[String:Any] = response  
    {  
        let status:String = dic["status"] as! String  
        if status == "OK"  
        {  
            //註冊成功  
        }  
    }  
})
```

response 格式範例：

```
{  
  "status": "OK",  
  "msg ": ""  
}
```

status：值"OK"為申請成功，"Fail"為失敗

msg：若 status 為"OK"，則為空值，反之，會回傳錯誤訊息

3.查詢使用者裝置清單

```
public func getUserDeviceList( UserID userId:String, completionHandler handler:ApiCompletionHandler?) ->  
Void
```

UserID：使用者帳號

response 格式範例：

```
status：值"OK"為查詢成功  
msg：回傳訊息  
list    ：裝置清單
```

4. 搜尋裝置

```
public func scanBeacon(timer:TimeInterval, _ completionHandler: @escaping ScanCompletionHandler) -> Void
```

Timer : 需要 Scan 的秒數

建立 **ScanCompletionHandler** 實例

```
BeaconAPI.sharedInstance.scanBeacon(timer: 2.0, { (state:BeaconAPI.ScanState) -> Void in
    if state == BeaconAPI.ScanState.Success
    {
        // var discoveredPeripherals:[mBeacon]
        var discoveredPeripherals:[mBeacon] =
            BeaconAPI.sharedInstance.getBeaconList()
    }
    else if state == BeaconAPI.ScanState.Scanning
    {
        print("still scanning")
    }
})
```

mBeacon 類別

beaconDeviceName() : 裝置名稱

macId : 唯一識別碼 · Beacon 硬體編號

5.配對裝置

```
public func bindBeaconWithUser( UserID userId:String, BeaconID beaconId:String, NickName name:String,  
completeHandler handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

NickName : 裝置暱稱

response 格式範例 :

```
status : 值"OK"為綁定成功  
msg : 回傳訊息
```

6. 解除配對裝置

```
public func unBindBeaconWithUser( UserID userId:String, BeaconID beaconId:String, completionHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

response 格式範例 :

```
status : 值"OK"為解除綁定成功  
msg : 回傳訊息
```

7. 開通裝置

```
public func activateBeacon(_ beacon:mBeacon?, _ completionHandler: @escaping ActivateCompletionHandler)  
-> Void
```

建立 **ActivateCompletionHandler** 實例

```
BeaconAPI.sharedInstance.activateBeacon(beacon, {(state:BeaconAPI.ActivateState) -> Void in  
    if state == BeaconAPI.ActivateState.Success  
    {  
        //開通成功  
    }  
}))
```


8. 設定丟失

```
public func setBeaconMissing( UserID userId:String, BeaconID beaconId:String, completeHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息
```

9. 取得丟失清單

```
public func getBeaconMissingList( UserID userId:String, completeHandler handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息  
list   : 丟失裝置清單
```

10. Beacon 碰撞回報

```
public func setBeaconCollision( BeaconID beaconId:String, Latitude lat:Float, Longitude lng:Float, completeHandler  
handler:ApiCompletionHandler?) -> Void
```

BeaconID : Beacon 硬體編號

Latitude : 使用者目前位置的緯度

Longitude : 使用者目前位置的經度

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息
```

11. 取得 Beacon 碰撞回報紀錄

```
public func getBeaconCollisionLog( BeaconID beaconId:String, completeHandler handler:ApiCompletionHandler?) ->  
Void
```

BeaconID : Beacon 硬體編號

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息  
list   : 裝置清單
```

12. 取消丟失狀態

```
public func cancelBeaconMissing( UserID userId:String, BeaconID beaconId:String, completionHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息
```

13. 裝置連線

```
public func connectBeacon(_ beacon:mBeacon?, _ completionHandler: @escaping ConnectCompletionHandler ) ->  
Void
```

beacon : mBeacon 實體

操作範例 :

```
BeaconAPI.sharedInstance.connectBeacon(beacon, { (state:BeaconAPI.ConnectState) -> Void in  
    if state == BeaconAPI.ConnectState.Connected  
    {  
        //Beacon Connected  
    }  
    else if state == BeaconAPI.ConnectState.ConnectFailed  
    {  
        //Beacon Connect Failed  
    }  
})
```

14. 裝置斷開連線 (需連線後才能操作)

```
public func disconnectBeacon(_ beacon:mBeacon?, _ completionHandler: @escaping  
DisconnectCompletionHandler) -> Void
```

beacon : mBeacon 實體

操作範例：

```
BeaconAPI.sharedInstance.disconnectBeacon(beacon, { (state:BeaconAPI.ConnectState) -> Void in  
    if state == BeaconAPI.ConnectState.Disconnected  
    {  
        //Beacon Disconnected  
    }  
    else if state == BeaconAPI.ConnectState.DisconnectFailed  
    {  
        //Beacon Disconnect Failed  
    }  
})
```

15. 使用蜂鳴器 (需連線後才能操作)

```
public func callBuzzer(_ beacon:mBeacon?)
```

beacon : mBeacon 實體

操作範例：

```
BeaconAPI.sharedInstance.callBuzzer(m_pBeacon)
```

16. 讀取電量 (需連線後才能操作)

```
public func readBattery(_ beacon:mBeacon?, _ completionHandler: @escaping ReadBatteryCompletionHandler)
```

beacon : mBeacon 實體

操作範例：

```
BeaconAPI.sharedInstance.readBattery(m_pBeacon,{ (responseDic) -> Void in  
  
    })
```

responseDic 格式範例：

status : 狀態 (low battery or battery level high)
volt : 電壓值(單位：v)
power : 電量百分比(單位：%)

17. 上傳紀錄 （需連線後才能操作）

```
public func updateDataLog(_ beacon:mBeacon?, _ userID:String, _ beaconID:String, _ completionHandler: @escaping UploadDataLogCompletionHandler)
```

beacon : mBeacon 實體

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

操作範例：

```
BeaconAPI.sharedInstance.updateDataLog(m_pBeacon, userID, beaconID, { (state) -> Void in
    if state == BeaconAPI.UploadDataStatus.success
    {
        //上傳成功
    }
    else
    {
        //上傳失敗
    }
})
```

18. 依指定日期取回紀錄

```
public func getDataByDate( UserID userID:String, BeaconID beaconId:String, Date date:String, completeHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

Date : 日期 (格式 : YYYY-MM-dd)

操作範例 :

```
BeaconAPI.sharedInstance.getDataByDate(UserID: userID, BeaconID: beaconID, Date: "2017-06-19",  
completeHandler: { (response) -> Void in  
  
    })
```

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息  
list   : 紀錄清單  
        temperature 溫度  
        humidity 濕度  
        low  
        high  
        time 時間
```

19. 依指定年月取回紀錄

```
public func getDataByMonth( UserID userId:String, BeaconID beaconId:String, Month date:String, completeHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID : 使用者帳號

BeaconID : Beacon 硬體編號

Month : 日期 (格式 : YYYY-MM)

操作範例 :

```
BeaconAPI.sharedInstance.getDataByMonth(UserID: userID, BeaconID: beaconID, Month: "2017-06",  
completeHandler: { (response) -> Void in  
  
    })
```

response 格式範例 :

```
status : 值"OK"為成功  
msg : 回傳訊息  
list   : 紀錄清單  
        temperature 溫度  
        humidity 濕度  
        low  
        high  
        day 日期
```


20. 依指定年取回紀錄

```
public func getDataByYear( UserID userID:String, BeaconID beaconId:String, Year date:String, completeHandler  
handler:ApiCompletionHandler?) -> Void
```

UserID：使用者帳號

BeaconID：Beacon 硬體編號

Year：日期（格式：YYYY）

操作範例：

```
BeaconAPI.sharedInstance.getDataByYear(userID: userID, BeaconID: beaconID, Year: "2017", completeHandler:  
{ (response) -> Void in  
  
    })
```

response 格式範例：

```
status：值"OK"為成功  
msg：回傳訊息  
list  ：紀錄清單  
        temperature  溫度  
        humidity  濕度  
        low  
        high  
        month  月份
```

21. 讓裝置進入 Bootloader 模式 (需連線後才能操作)

```
public func enableBootLoader(_ beacon:mBeacon?, completionHandler  
handler:EnableBootLoaderCompletionHandler?) -> Void
```

beacon : mBeacon 實體

操作範例：

```
BeaconAPI.sharedInstance.enableBootLoader(m_pBeacon, completionHandler:  
{ (state:BeaconAPI.BootloaderState) -> Void in  
    if state == BeaconAPI.BootloaderState.enable  
    {  
        print("成功");  
    }  
})
```

注意事項：進入 Bootloader 後，Device 會自動斷線，避免在這裡來不及取得回傳的 Handler status，所以需要事先監聽 Notification

```
NotificationCenter.default.addObserver(self, selector:  
#selector(handleNotificationForBleDisconnected), name:  
Notifications.POST_NOTIFICATION_DISCONNECT, object: nil)
```

22. 執行 OTA 功能 (需連線後才能操作)

```
public func performDFU(_ aFileURL:URL)
```

aFileURL : firmware 的 file url path

操作範例：

```
if let resourceUrl = Bundle.main.url(forResource: "collar_dev_104", withExtension: "zip",
subdirectory: "")
{
    //需先設定委派，並且需要實作 BeaconAPIDelegate

    BeaconAPI.sharedInstance.delegate = self

    BeaconAPI.sharedInstance.performDFU(resourceUrl)
}

//實作 BeaconAPIDelegate
extension YourViewController : BeaconAPIDelegate
{
    func dfuProgressUpdate(for part: Int,
                           outOf totalParts: Int,
                           to progress: Int,
                           currentSpeedBytesPerSecond: Double,
                           avgSpeedBytesPerSecond: Double) -> Void
    {
        dfuUploadStatus.text = String(format: "Part: %d/%d\nSpeed: %.1f KB/s\nAverage Speed:
%.1f KB/s",part, totalParts, currentSpeedBytesPerSecond/1024, avgSpeedBytesPerSecond/1024)
    }

    func dfuStatusUpdate(to state: BeaconAPI_DFUState) -> Void
    {
        dfuStatusLabel.text = state.description()

        print("Changed state to: \(state.description())")
    }
}
```

注意事項：當 BeaconAPI_DFUState 回傳值為 completed，表示 DFU 結束