iOS SWIFT SDK

說明手冊(V1.6)



iOS Swift Bluetooth/WIFI 函式庫使用說明

1. searchBLEDevice(callback)

- 函式說明：搜尋藍牙 BLE 裝置並回傳
- 參數說明：

➔ **callback(btDictionary: [String: CBPeripheral])**：函式，回傳所有搜尋到的 BLE 裝置，如：[“Printer_7544”: CBPeripheral, “Printer_7534”: CBPeripheral]

2. openPort()

★ Bluetooth

openPort (connectDecive, callback)

- 函式說明：指定藍牙的裝置和回傳狀態函式後連線
- 參數說明：

➔ **connectDecive**：CBPeripheral 型別，指定連線的藍牙 Peripheral 裝置

➔ **callback(msg: String)**：函式，回傳連線狀態

回傳字串	
“disconnected”	已斷線
“connecting”	連線中
“connected”	已連線
“disconnecting”	斷線中
“time_out”	逾時

★ WIFI

openPort(IP, port, callback)

- 函式說明：指定印表機的 IP 位址與連接埠和回傳狀態函式後，開啟輸出埠
- 參數說明：

  ➔ IP: String 型別，指定連線的 IP 位址 ，如：”192.168.1.109”

  ➔ port: Int 型別，指定連線的連接埠如：9100

  ➔ callback(msg: String): 函式，回傳連線狀態

回傳字串	
“connecting”	連線中
“connected”	已連線
“time_out”	逾時

3. closePort()

★ Bluetooth

closePort (callback)

- 函式說明：藍牙斷線
- 參數說明：

  ➔ **callback(msg: String)**：函式，回傳斷線訊息“disconnected”

★ WIFI

closePort (callback)

- 函式說明：關閉輸出埠
- 參數說明：

  ➔ **callback(msg: String)**：函式，回傳斷線訊息“disconnected”

4. setup(width, height, speed, density, sensor, sensorDistance, sensorOffset, callback)

- 函式說明：設定標籤的寬度、高度、列印速度、列印熱度、感應器類別、間隙/黑標垂直間距、間隙/黑標偏移距離
- 參數說明：

參數	型別	說明
width	Int	設定標籤寬度，單位 mm
height	Int	設定標籤高度，單位 mm
speed	Int	設定列印速度，1~15，代表每秒 1~15 吋列印速度(隨機型不同會有不同列印最高上限，最高為每秒 15 吋列印速度)
density	Int	設定列印濃度，0~15，數字越大列印結果越黑
sensor	Int	設定使用感應器之類別； 0：表示使用間隙感測器(gap sensor) 1：表示使用黑標感測器(black mark sensor)
sensorDistance	Int	設定間隙/黑標垂直間距高度，單位 mm
sensorOffset	Int	設定間隙/黑標垂直間距高度，單位 mm，此參數若使用一般

		標籤時均設為 0
callback	函式	未連線時回傳”Connection doesn't exist.”

5. clearBuffer()

- 函式說明：清除圖像緩衝

6. printBarcode(x, y, type, height, readable, rotation, narrow, wide, content, callback)

- 函式說明：使用印表機內建條碼列印
- 參數說明：

參數	型別	說明
x	Int	條碼 X 方向起始點，以點(dot)表示
y	Int	條碼 Y 方向起始點，以點(dot)表示
type	String	設定條碼類型(Code Type) ，請參考附件
height	Int	設定條碼高度，高度以點來表示
readable	Int	設定是否列印條碼碼文 0:不列印 1:列印條碼碼文置左 2:列印條碼碼文置中 3:列印條碼碼文置右
rotation	Int	設定條碼旋轉角度

		0：旋轉0度 90：旋轉90度 180：旋轉180度 270：旋轉270度
narrow	Int	設定條碼窄 bar 比例因子，請參考附件
wide	Int	設調條碼寬 bar 比例因子，請參考附件
content	String	設定欲列印之條碼內容
callback	函式	未連線時回傳”Connection doesn't exist.”

7. printQRCode(x, y, eccLevel, cellWidth, rotation, content, callback)

- 函式說明：使用印表機內建 QRcode 列印
- 參數說明：

參數	型別	說明
x	Int	QRcode X 方向起始點，以點(dot)表示
y	Int	QRcode Y 方向起始點，以點(dot)表示
eccLevel	String	容錯等級，L: 7%, M: 15%, Q: 25%, H: 30%
cellWidth	Int	設定 QRcode 長寬，設定範圍為 1~10
rotation	Int	設定 QRcode 旋轉角度 0：旋轉0度 90：旋轉90度

		180：旋轉180度 270：旋轉270度
content	String	設定欲列印之 QRcode 內容
callback	函式	未連線時回傳”Connection doesn't exist.”

8. formFeed()

- 函式說明：跳頁，該函式需在 setup 後使用

9. noBackFeed()

- 函式說明：設定紙張不回吐

10. sendCommand (_ string)

- 函式說明：送內建指令到印表機
- 參數說明：
  - ➔ **string**：String 型別，設定指令內容，詳細指令請參考 TSPL 使用手冊

11. sendByteCmd(cmdData)

- 函式說明：傳送 byte 形式的資料到印表機
- 參數說明：

➔ **cmdData**: Data 型別

12. printFont(x, y, fontName, rotation, x_scale, y_scale, content, callback)

- 函式說明：使用印表機內建文字列印
- 參數說明：

參數	型別	說明
x	Int	文字 X 方向起始點，以點(dot)表示
y	Int	文字 Y 方向起始點，以點(dot)表示
fontName	String	內建字型名稱，共五種 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots TST24.BF2: 繁體中文24*24 dots TST16.BF2: 繁體中文16*16 dots TSS24.BF2: 簡體中文24*24 dots TSS16.BF2: 簡體中文16*16 dots
rotation	Int	設定文字旋轉角度

		0：旋轉0度 90：旋轉90度 180：旋轉180度 270：旋轉 270 度
x_scale	Int	設定文字 X 方向放大倍率，1~10
y_scale	Int	設定文字Y方向放大倍率，1~10
content	String	設定欲列印之文字內容
callback	函式	未連線時回傳”Connection doesn't exist.”

13. printLabel(set, copy)

- 函式說明：列印標籤內容
- 參數說明：
  - ➔ **set**: Int 型別，設定列印標籤式數(set)
  - ➔ **copy**: Int 型別，設定列印標籤份數(copy)

14. downloadPCX(filePath, filename, callback)

- 函式說明：下載單色 PCX 格式圖檔至印表機
- 參數說明：
  - ➔ filePath：URL 類別，帶入 PCX 檔所在位置
  - ➔ fileName：String 型別，檔案名稱(須包含副檔名)
  - ➔ callback(msg: String)：函式，未連線時回傳”Connection doesn't exist.”

15. downloadBMP(filePath, filename, callback)

- 函式說明：下載單色 BMP 格式圖檔至印表機
- 參數說明：
  - ➔ **filePath**：URL 類別，帶入 BMP 檔所在位置
  - ➔ **fileName**：String 型別，檔案名稱(須包含副檔名)
  - ➔ **callback(msg: String)**：函式，未連線時回傳”Connection doesn't exist.”

16. getSdkVersion ( )

- 函式說明：回傳此 SDK 版本號
- 回傳說明：回傳版本號(String)

17. printerStatus(callback)

- 函式說明：帶入回傳狀態函式後回傳印表機狀態
- 參數說明：
  - ➔ **callback(status: String)**：函式，回傳印表機狀態
- 回傳字串說明：

回傳字串	印表機狀態
00	就緒
01	上蓋開啟
02	卡紙
03	卡紙且上蓋開啟

04	標籤用盡
05	標籤用盡且上蓋開啟
08	碳帶用盡
09	碳帶用盡且上蓋開啟
0A	碳帶用盡且卡紙
0B	碳帶用盡、卡紙且上蓋開啟
0C	碳帶用盡且標籤用盡
0D	碳帶用盡、標籤用盡且上蓋開啟
10	暫停
20	列印中
80	其他錯誤
time_out	逾時
Connection doesn't exist.	未連線

18. setDirectionAndMirror(direction,mirror)

- 函式說明：設定標籤列印時的出紙方向與是否使用鏡像列印
- 參數說明：

參數	型別	說明
direction	int	設定出紙方向，預設為 0 0：頂端出紙 1：底端出紙
mirror	int	設定是否鏡像列印 0：否 1：是

19. setShift(shiftY)

- 函式說明：設定圖像垂直位移距離，數值為正時，圖像會往列印方向移動，數值為負時，圖像會背離列印方向
- 參數說明：

  ➔ **shiftY**：int 型別，垂直位移距離，單位為 dot

20. printReverse(x_start, y_start, x_width, y_height)

- 函式說明：將指定的區域於列印時反白
- 參數說明：

參數	型別	說明
x_start	int	指定 X 起始座標位置，以點(dot)表示
y_start	int	指定 Y 起始座標位置，以點(dot)表示
x_width	int	指定 X 座標寬度，以點(dot)表示
y_height	int	指定 Y 座標高度，以點(dot)表示

21. setOffset(offset)

- 函式說明：設定每次出紙後額外偏移的距離(通常與剝紙模式和裁切模式組合使用)
- 參數說明：

  ➔ offset：double 型別，額外的出紙偏移，單位為 mm

22. setCutMode(mode, piece)

- 函式說明：設定裁切模式與張數
- 參數說明：

參數	型別	說明
mode	int	設定裁切方式，預設為 1 0：反切 1：正切
piece	int	設定裁切張數

23. setAfterPrintAction(mode)

- 函式說明：設定列印後動作
- 參數說明：

參數	型別	說明
mode	int	設定列印後動作，預設為 1 0：停在原地 1：撕紙 2：剝紙 3：裁切

24. genericDefault()

- 函式說明：將印表機之一般設定值初始化
- 參數說明：無

25. sensorDefault()

- 函式說明：將印表機之感應器設定值初始化
- 參數說明：無

26. switchWifiFrequency(frequency, callback)

- 函式說明：使用兼容 5G 頻段 WIFI 模塊時，可用於切換使用頻段
- 參數說明：
  - ➔ Frequency：字串型別，設定模塊頻段 ；2.4G：使用 2.4G 頻段、5G：使用 5G 頻段、BOTH：使用雙頻頻段
  - ➔ callback: 未連線時回傳”Connection doesn't exist.”

27. printBitmap(imageData, x, y, width, height, mode, callback)

- 函式說明：列印圖片檔案(僅支援 jpg, png)
- 參數說明：

參數	型別	說明
imageData	Data	圖片資料
x	int	圖片 x 軸起始位置
y	int	圖片 y 軸起始位置

width	int	指定圖片寬度（不指定，則為圖片原始寬度）
height	int	指定圖片高度（不指定，則為圖片原始高度）
mode	int	打印模式選擇 ，0、1、2 為標準傳送，未壓縮檔案，速度較慢，3 為壓縮傳送，速度較快 0: OVERWRITE 1: OR 2: XOR 3: Lz77 壓縮
callback	函式	未連線時回傳”Connection doesn't exist.”

28. setRealTimeStatus(status, callback)

- 函式說明：開啟或關閉實時命令，印表機第一次開機預設狀態為關閉
- 參數說明：

參數	型別	說明
Status	String	實時命令開啟狀態 0: 關閉狀態 1: 開啟狀態
callback	函式	未連線時回傳：”Connection doesn't exist.”

29. getRealTimeStatus(callback)

- 函式說明：讀取實時命令的開啟或關閉狀態
- 參數說明：

  ➔ callback(msg: String)：函式，回傳讀取的資料或顯示逾時
- 回傳說明：

回傳	型別	說明
0	String	實時命令關閉狀態回傳 FB:0\r\n
1	String	實時命令開啟狀態回傳 FB:1\r\n
time_out	String	逾時
Connection doesn't exist.	String	未連線

30. getStatusNumber() -> String

- 函式說明：讀取實時狀態數字
- 參數說明：無
- 回傳數字說明：

回傳字串	印表機狀態
00	就緒
01	上蓋開啟
02	卡紙
03	卡紙且上蓋開啟
04	標籤用盡
05	標籤用盡且上蓋開啟
08	碳帶用盡

09	碳帶用盡且上蓋開啟
0A	碳帶用盡且卡紙
0B	碳帶用盡、卡紙且上蓋開啟
0C	碳帶用盡且標籤用盡
0D	碳帶用盡、標籤用盡且上蓋開啟
10	暫停
20	列印中
80	其他錯誤

1. writeUHF(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, dataString, callback)

- 函式說明：將資料寫入 UHF 標籤記憶體中
- 參數說明：

參數	型別	說明
dataFormat	String	設定 String 資料編碼格式，預設為 H A：ASCII H：Hexadecimal
startBlockNo	Int	設定資料區塊起始位置，預設為 2
byteSize	Int	設定寫入資料byte長度，預設為1
Gen2MemoryBank	String	設定 Gen2 資料區段，預設為 E R：保留 E：EPC T：TID(Tag ID) U：User
dataString	String	欲寫入之 String 資料
callback	函式	未連線時回傳”Connection doesn't exist.”

2. EPCPWD_Action(action, password)

- 函式說明：將 UHF GNE2 的 EPC 資料區塊上鎖或解鎖
- 參數說明：

參數	型別	說明
action	String	設定執行動作 U：解鎖資料區塊 L：上鎖資料區塊 O：永久解鎖資料區塊 P：永久上鎖資料區塊
password	String	密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

3. TIDPWD_Action(action, password)

- 函式說明：將 UHF GNE2 的 TID 資料區塊上鎖或解鎖
- 參數說明：

參數	型別	說明
action	String	設定執行動作 U：解鎖資料區塊 L：上鎖資料區塊 O：永久解鎖資料區塊 P：永久上鎖資料區塊
password	String	密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

4. USERPWD_Action(action, password)

- 函式說明：將 UHF GNE2 的 USER 資料區塊上鎖或解鎖
- 參數說明：

參數	型別	說明
action	String	設定執行動作 U：解鎖資料區塊 L：上鎖資料區塊 O：永久解鎖資料區塊 P：永久上鎖資料區塊
password	String	密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

5. accessPWD_Action(action, password)

- 函式說明：將 UHF GNE2 的存取密碼進行設定、上鎖或解鎖
- 參數說明：

參數	型別	說明
action	String	設定執行動作 U：解鎖存取/刪除密碼 L：上鎖存取/刪除密碼

		O：永久解鎖存取/刪除密碼 P：永久上鎖存取/刪除密碼 S：設定存取/刪除密碼
password	String	密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

6. killPWD_Action(action, password)

- 函式說明：將 UHF GNE2 的存取密碼進行設定、上鎖或解鎖
- 參數說明：

參數	型別	說明
action	String	設定執行動作 U：解鎖存取/刪除密碼 L：上鎖存取/刪除密碼 O：永久解鎖存取/刪除密碼 P：永久上鎖存取/刪除密碼 S：設定存取/刪除密碼
password	String	密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

7. set_RFIDPorcedure(tagType, rw_position, void_printout, tryEncode_times, error_handle, speed, retry_times, callback)

- 函式說明：RFID 設定
- 參數說明：

參數	型別	說明
tagType	Int	設定標籤類型，1~10，預設值為 8 1：EPC Class 1 Generation 2-Q , 8：EPC Class 1 Generation 2-R ,10：UHF-J
rw_position	Int	設標籤讀寫位置(標籤頂部起算)，範圍為 0~9999(dot)，預設為 0
void_printout	Int	設定無效列印長度(dot)，範圍為 0~標籤長度，預設為標籤長度
tryEncode_times	Int	設定最大無效標籤數，範圍為 0~10，預設為 3
error_handle	String	設定無效時採取的動作，預設為 N N：No action(繼續) P：Pause mode(暫停) E：Error mode(停止)
speed	Int	設定無效打印速度，範圍 2~10(IPS)，預設值 2(IPS)
retry_times	Int	設定標籤重試次數，範圍 0~10，預設值 6
callback	函式	未連線時回傳”Connection doesn't exist.”

8. setRFIDProcedure(tagType, rw_position, void_printout, tryEncode_times, error_handler, speed, retry_times, dpi, callback)

- 函式說明：RFID 設定
- 參數說明：

參數	型別	說明
tagType	Int	設定標籤類型，1~10，預設值為 8 1：EPC Class 1 Generation 2-Q , 8：EPC Class 1 Generation 2-R ,10：UHF-J
rw_position	Int	設標籤讀寫位置(標籤頂部起算)，預設為 0 範圍為： 203dpi:0 ~ 1251 (mm)、 300dpi:0 ~ 846 (mm)、 600dpi:0 ~ 423 (mm)。
void_printout	Int	設定無效列印長度(dot)，範圍為 0~標籤長度，預設為標籤長度
tryEncode_times	Int	設定最大無效標籤數，範圍為 0~10，預設為 3
error_handle	String	設定無效時採取的動作，預設為 N N：No action(繼續) P：Pause mode(暫停) E：Error mode(停止)
speed	Int	設定無效打印速度，範圍 2~10(IPS)，預設值 2(IPS)
retry_times	Int	設定標籤重試次數，範圍 0~10，預設值 6

dpi	String	設定打印機的 DPI 203: 203 dpi 300: 300 dpi 600: 600 dpi
callback	函式	未連線時回傳”Connection doesn't exist.”

9. writeHF(dataFormat, startBlockNo, byteSize, dataString, callback)

- 函式說明：將資料寫入 HF 標籤記憶體中
- 參數說明：

參數	型別	說明
dataFormat	String	設定 String 資料編碼格式，預設為 H A：ASCII H：Hexadecimal
startBlockNo	Int	設定資料區塊起始位置，預設為 2
byteSize	Int	設定寫入資料 byte 長度，預設為 1
dataString	String	欲寫入之 String 資料
callback	函式	未連線時回傳”Connection doesn't exist.”

10. printFontBlock (x, y ,width, height, fontName, rotation, x_scale, y_scale, space, align, content, callback)

- 函式說明：列印段落文字內容
- 參數說明：

參數	型別	說明
x	Int	文字 X 方向起始點，以點(dot)表示
y	Int	文字 Y 方向起始點，以點(dot)表示
width	Int	設定段落區塊寬度，以點(dot)表示
height	Int	設定段落區塊高度，以點(dot)表示
fontName	String	內建字型名稱 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots TST24.BF2: 繁體中文24*24 TST16.BF2: 繁體中文16*16 TSS24.BF2: 簡體中文24*24 TSS16.BF2: 簡體中文16*16
rotation	Int	設定文字旋轉角度 0：旋轉0度 90：旋轉90度 180：旋轉180度

		270：旋轉 270 度
x_scale	Int	設定文字 X 方向放大倍率，1~10
y_scale	Int	設定文字X方向放大倍率，1~10
space	Int	行距，以點(dot)表示
align	Int	對齊位置 0：預設(置左) 1：置左 2：置中 3：置右
content	String	設定欲列印之文字內容
callback	函式	未連線時回傳”Connection doesn't exist.”

11. readUHF(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, callback)

- 函式說明：讀取 UHF Gen2 標籤記憶體資料，需用字串變數接收回傳訊息
- 參數說明：

參數	型別	說明
dataFormat	String	設定 String 資料編碼格式，預設為 H A：ASCII H：Hexadecimal
startBlockNo	Int	設定資料區塊起始位置，預設為 0
byteSize	Int	設定讀取資料byte長度，預設為1

Gen2MemoryBank	String	設定 Gen2 資料區段，預設為 E R：保留 E：EPC T：TID(Tag ID) U：User
callback	函式	未連線時回傳：”Connection doesn't exist.” 連線時回傳： 1. 標籤資料 2. “Error code: 錯誤代碼(參考錯誤代碼頁面)” 3. “time_out” 逾時

■ 回傳字串說明：

dataFormat	回傳字串(範例)
A	標籤資料以 ASCII 顯示 (ex: 24051324000103456400)
H	標籤資料以 Hexadecimal 顯示 (ex: 32343035313332343030303130333435363430 30)

12. query_UHF(dataFormat, PCReturnStatus, CRCReturnStatus, callback)

■ 函式說明：以 Q 指令讀取 UHF Gen2 標籤記憶體 EPC 資料區段資料，需用字串變數接收回

傳訊息

■ 參數說明：

參數	型別	說明
dataFormat	String	設定 String 資料編碼格式，預設為 H A：ASCII H：Hexadecimal
PCReturnStatus	Int	PC 返回狀態，預設為 0 0：不回傳 PC 值 1：回傳 PC 值
CRCReturnStatus	Int	CRC-16 返回狀態，預設為 0 0：不回傳 CRC-16 1：回傳CRC-16
callback	函式	未連線時回傳：”Connection doesn't exist.” 連線時回傳： 1. 標籤資料 2. “Error code: 錯誤代碼(參考錯誤代碼頁面)” 3. “time_out” 逾時

■ 回傳字串說明：

以 query_UHF("A", 1, 1)和 query_UHF("H", 1, 1)為例：(PC 值和 CRC-16 皆回傳)

dataFormat	回傳字串(範例)
A	標籤資料以 ASCII 顯示 (ex: 24051324000103456400)
H	標籤資料以 Hexadecimal 顯示 (ex: 32343035313332343030303130333435363430 30)

以 query_UHF("A", 0, 0)和 query_UHF("H", 0, 0)為例：(PC 值和 CRC-16 皆不回傳)

dataFormat	回傳字串(範例)
A	標籤資料以 ASCII 顯示 (ex: 0513240001034564)
H	標籤資料以 Hexadecimal 顯示 (ex: 303531333234303030313033343536 34)

13. rfid_calibration(callback)

- 函式說明：執行 RFID Tag 校準動作
- 參數說明：**callback**: 未連線時回傳：”Connection doesn't exist.”

14. rfidSetupDefault()

- 函式說明：將 RFID 設定值初始化
- 參數說明：無

1. set_GJB_ Pwd_Action(passwordArea, newPassword, writePassword)

- 函式說明：設定 UHF GJB 各密碼區域新密碼
- 參數說明：

參數	型別	說明
passwordArea	String	設定密碼區域，預設為 W K：Kill W：Write R：Read S：Status
newPassword	String	設定密碼區域的新密碼，應為8 hex字元(0~9,A,B,C,D,E,F)
writePassword	String	寫入密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

2. writeGJB_UHF(dataFormat, startBlockNo, byteSize, GJBMemoryBank, dataString, writePassword, callback)

- 函式說明：將資料寫入 UHF GJB 標籤記憶體中
- 參數說明：

參數	型別	說明
dataFormat	String	設定 String 資料編碼格式，預設為 H A：ASCII H：Hexadecimal
startBlockNo	Int	設定資料區塊起始位置，GJB 預設為 1
byteSize	Int	設定寫入資料byte長度，預設為1
GJBMemoryBank	String	設定 GJB 資料區段，預設為 E R：安全區 E：EPC T：TID(Tag ID) U：User
dataString	String	欲寫入之 String 資料
writePassword	String	寫入密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)
callback	函式	未連線時回傳”Connection doesn't exist.”

3. readGJB_UHF(dataFormat, startBlockNo, byteSize, GJBMemoryBank, readPassword, callback)

- 函式說明：讀取 UHF GJB 標籤記憶體資料，需用字串變數接收回傳訊息
- 參數說明：

參數	型別	說明
dataFormat	String	設定 String 資料編碼格式，預設為 H A：ASCII H：Hexadecimal
startBlockNo	Int	設定資料區塊起始位置，預設為 0
byteSize	Int	設定讀取資料byte長度，預設為1
GJBMemoryBank	String	設定 GJB 資料區段，預設為 E R：安全區 E：EPC T：TID(Tag ID) U：User
readPassword	String	讀取密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)
callback	函式	未連線時回傳：”Connection doesn't exist.” 連線時回傳： 4. 標籤資料 5. “Error code: 錯誤代碼(參考錯誤代碼頁面)”

		6. “time_out” 逾時

- 回傳字串說明：

dataFormat	回傳字串(範例)
A	標籤資料以 ASCII 顯示 (ex: 24051324000103456400)
H	標籤資料以 Hexadecimal 顯示 (ex: 32343035313332343030303130333435363430 30)

4. set_GJB_Status_UHF(GJBMemoryBank, action, statusPassword, callback)

- 函式說明：設定 UHF GJB 各資料區塊讀寫狀態
- 參數說明：

參數	型別	說明
GJBMemoryBank	String	設定 GJB 資料區段，預設為 E F：安全區 E：EPC T：TID(Tag ID) U：User
action	String	設定狀態，預設為 A

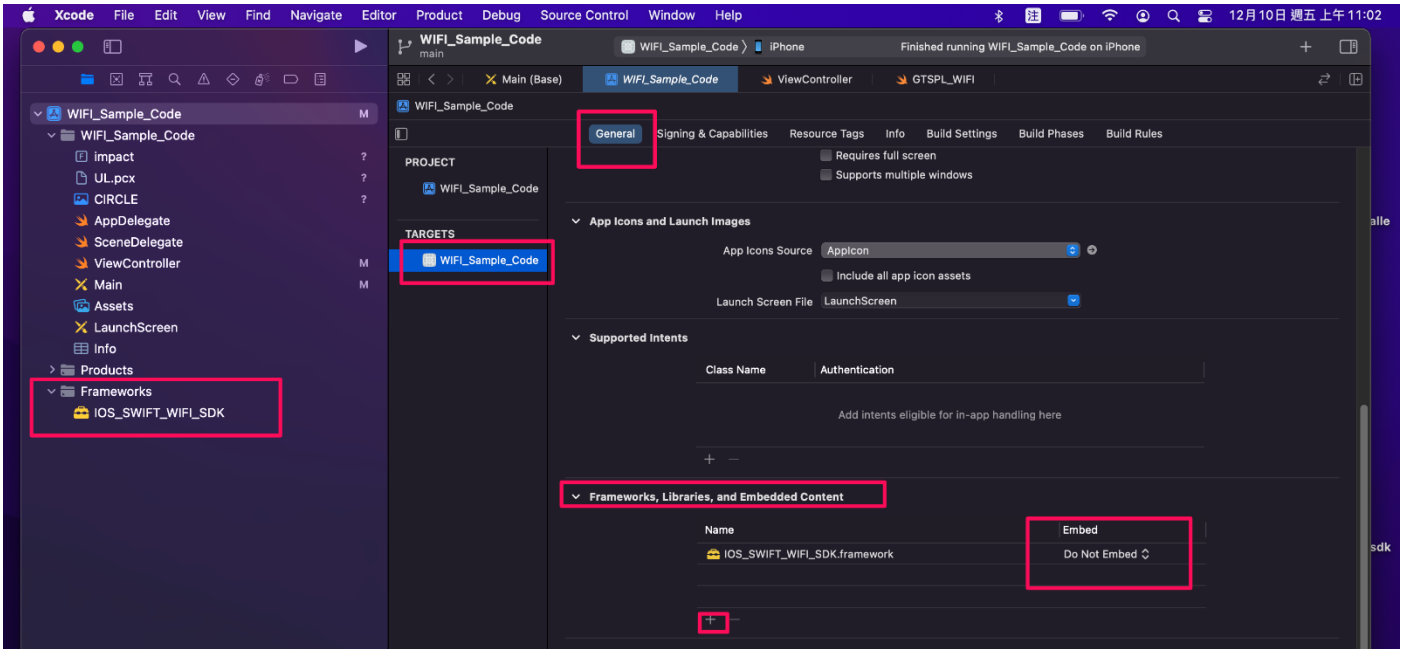
		A=Lock0(可讀可寫) B=Lock1(可讀不可寫) C=Lock2(不可讀可寫) D=Lock3(不可讀不可寫)
statusPassword	String	狀態密碼，應為8 hex字元(0~9,A,B,C,D,E,F)
callback	函式	未連線時回傳：”Connection doesn't exist.”

5. Kill_GJB_Tag_UHF(killPassword)

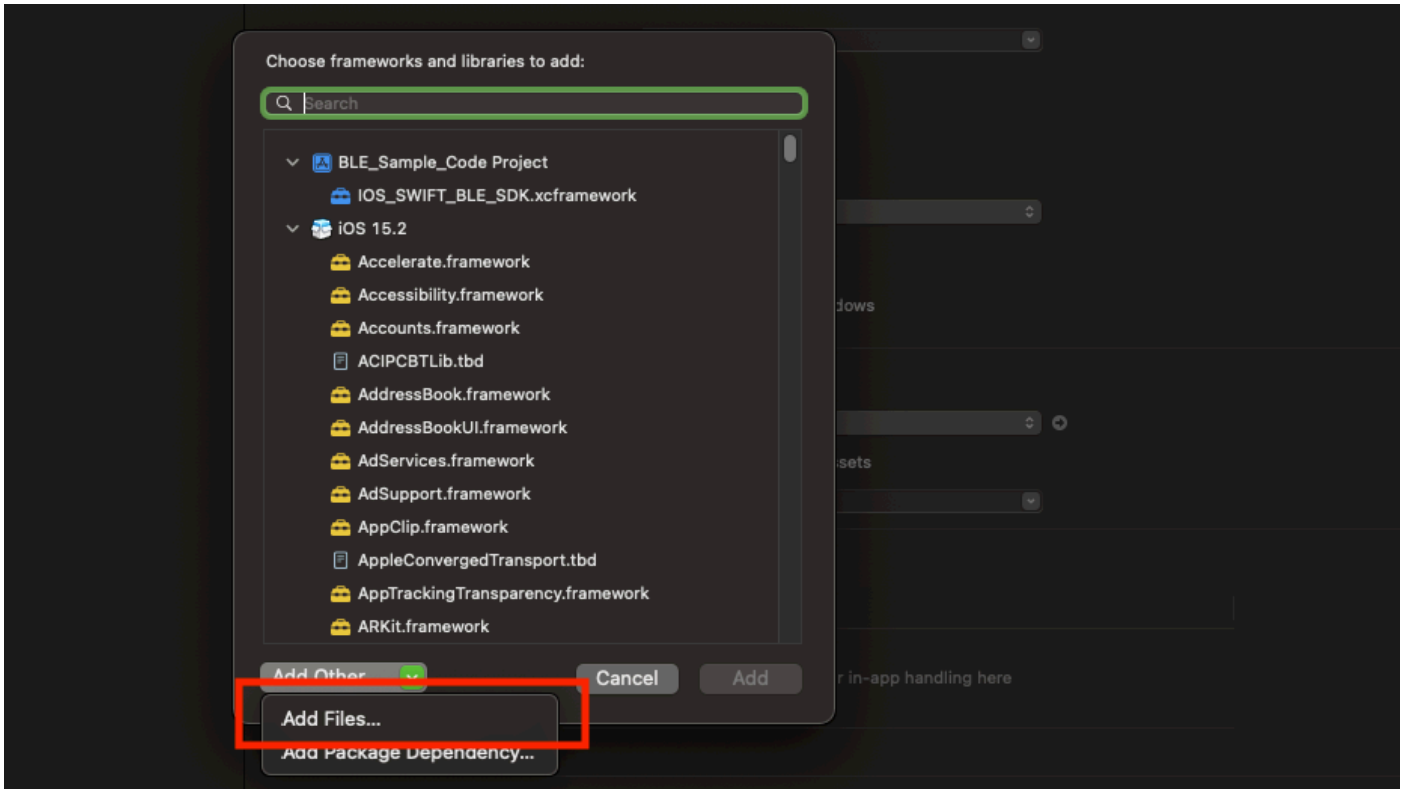
- 函式說明：刪除 UHF GJB 標籤
- 參數說明：

參數	型別	說明
killPassword	String	刪除密碼，應為 8 hex 字元(0~9,A,B,C,D,E,F)

1.導入 Framework：Target => General => Frameworks, Libraires, and Embedded Content 中加入 WIFI 和 BLE XCFramework，注意 Embed 要選擇 Do Not Embed

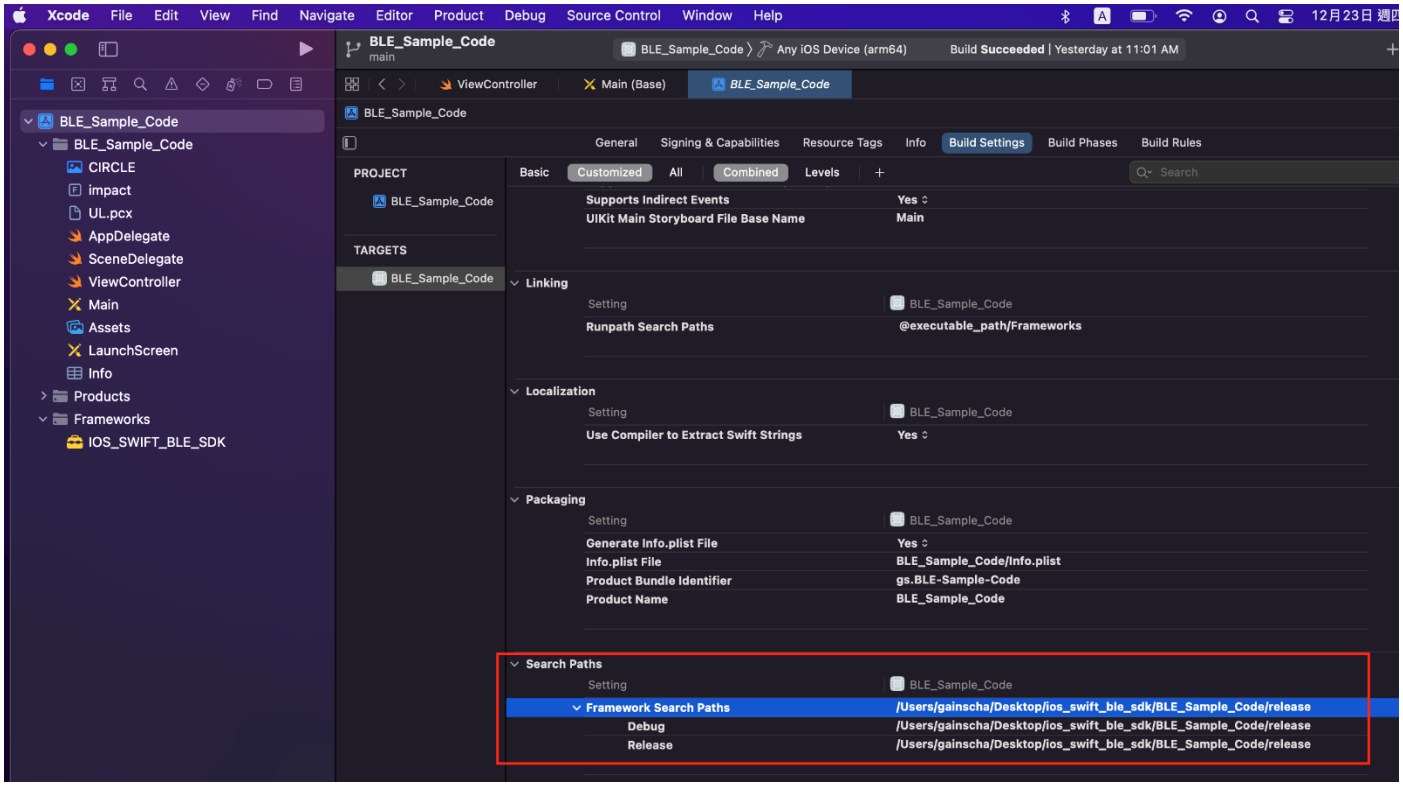


2.選 Add files，並選擇 .xcframework 的資料，按 open 鍵匯入

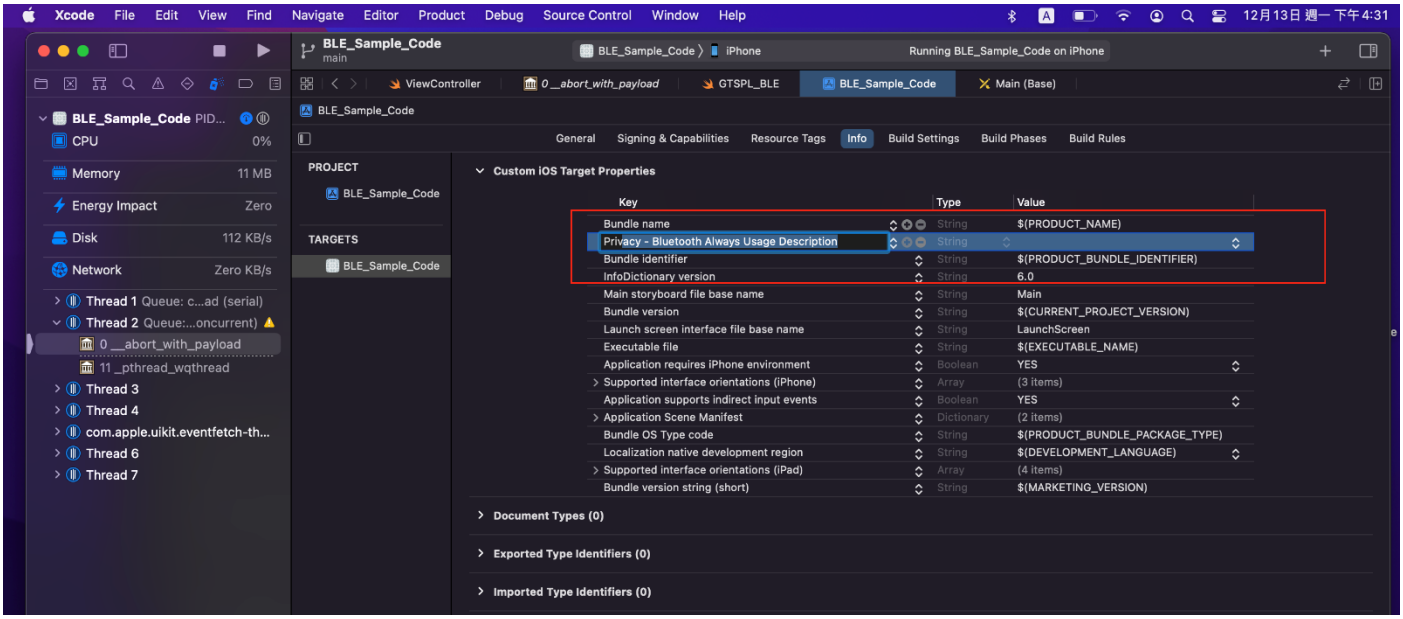


3.設定 Search Paths：Target => Building Settings => Combined => Search Paths，設定 Framework 所在

路徑



1.需先於 Target=> Info 設定 Properties：Privacy – Bluetooth Always Usage Description



2.匯入 GTSPL_SDK：

```
import IOS_SWIFT_BLE_SDK
```

3.範例程式：

```
//初始化 SDK
    let gsble = GTSPL_BLE()
//紀錄藍牙設備的陣列
    var BTDictionary = [String: CBPeripheral]()
//所選的藍牙設備名稱
    var mBleName:String?
//搜尋 BLE 裝置
```

```
        gsble.searchBLEDevice(){(btDictionary: [String: CBPeripheral]) in
            self.BTDictionary = btDictionary
         }
//BLE  裝置名稱
        let btdevice = BTDictionary[mBleName!]
//連線 BLE 裝置
        gsble.openPortBLE(connectDevice: btdevice!){(msg) in
            self.textView.text = msg
        }
//BLE 裝置斷線
        gsble.closePort(){(msg) in
            self.textView.text = msg
        }
//設定印表機
        gsble.setup(width: 105, height: 80, speed: 4, density: 6, sensor: 0, sensorDistance:
5, sensorOffset:5) { msg in
            self.textView.text = msg
        }
```

```
//列印 barcode、text、bitmap、區塊文字

    //先清除資料、再列印指定項目

    gsble.clearBuffer()

    //barcode

    gsble.printBarcode(x: 30, y: 30, type: "128", height: 100, human_readable: 1,
rotation: 0, narrow:2, wide: 2, content: "barcode987654321")

    //text

    gsble.printFont( x: 100, y: 180, fontname: "3", rotation: 0, xScale: 3, yScale: 3,
content: "12345678 print test")

    //區塊文字

    let paragraph="We stand behind our products with one of the most comprehensive
support programs in the Auto-ID industry."

    gsble.printFontBlock(x:15,y:15,width:790,height:90,fontname:"1",rotation:0,xScale:
8,yScale:8,space:20,align:0,content:paragraph)

    //指定 bitmap 的檔案 url

    let url = Bundle.main.url(forResource: "CIRCLE", withExtension:"bmp")!

    //傳送 bitmap 檔案到印表機中

    gsble.downloadBMP(filepath: url, filename: "CIRCLE.bmp")

    //將 bitmap 圖片排入列印順序指令
```

```
        gsble.sendcommand("PUTBMP 30,30,\"CIRCLE.bmp\"\r\n")

        //執行列印動作

        gsble.printLabel(set: 1, copy: 1)


//列印 QRcode

        //先清除資料

        gsble.clearBuffer()

        //形成 QRcode

        gsble.printQRCode(x: 50, y: 50, eccLevel: "H", cellWidth: 4, mode: "A", rotation: 0,
content: "QRcode987654321")

        //列印動作

        gsble.printLabel(set: 1, copy: 1)


//簡中列印

        cn: String = "默认简体中文测试"

        gsble.clearBuffer()

        gsble.printFont(x: 200, y: 200, fontName: "TSS24.BF2", rotation: 0, x_scale: 1,
y_scale: 1, content: cn)
```

```
        gsble.printLabel(set: 1, copy: 1)


//繁中列印

        cn: String = "預設繁體中文測試"

        gsble.clearBuffer()

        gsble.printFont(x: 200, y: 200, fontName: "TST24.BF2", rotation: 0, x_scale: 1,
y_scale: 1, content: cn)

        gsble.printLabel(set: 1, copy: 1)


//取得 sdk 版本

        let version = gsble.getSdkVersion()

//取得印表機狀態

        var statusStr = ""

         gsble.printerStatus(){(status) in

         switch status{

              case "00":

                   statusStr = "Normal"

              case "01":
```

```
        statusStr = "Head opened"
case "02":
        statusStr = "Paper Jam"
case "03":
        statusStr = "Paper Jam and head opened"
case "04":
        statusStr = "Out of paper"
case "05":
        statusStr = "Out of paper and head opened"
case "08":
        statusStr = "Out of ribbon"
case "09":
        statusStr = "Out of ribbon and head opened"
case "0A":
        statusStr = "Out of ribbon and paper jam"
case "0B":
        statusStr = "Out of ribbon, paper jam and head opened"
case "0C":
```

```
            statusStr = "Out of ribbon and out of paper"
        case "0D":
            statusStr = "Out of ribbon, out of paper and head opened"
        case "10":
            statusStr = "Pause"
        case "20":
            statusStr = "Printing"
        case "80":
            statusStr = "Other error"
        default:
            break
        }
    }
```

```
// 出紙方向、鏡像列印
gsble.setDirectionAndMirror(direction: 0, mirror: 0)
```

```
// 圖像垂直位移距離 例：偏移 36 dot
```

```
gsble.setShift(shiftY: 36)

// 反白列印（輸入區域）

gsble.printReverse(x_start: 0, y_start: 0, x_width: 100, y_height: 100)

// 出紙偏移 例：9.9mm

gsble.setOffset(offset: 9.9)

// 設定裁切模式與張數 例：正切、三張

gsble.setCutMode(mode: 1, piece: 3)

// 設定列印後動作 例：撕紙

gsble.setAfterPrintAction(mode: 1)

// 一般設定值初始化

gsble.genericDefault()

// 感應器設定值初始化
```

```
gsble.sensorDefault()

// 切換 wifi 2.4G/5G/Both，例：2.4G
gsble.switchWifiFrequency(frequency: “2.4G”) { msg in
    print(“Error message: \(msg)”)
}

// 列印圖片，轉單色，例：模式 1, 自訂大小為 200 dot X 100 dot
gsble.printBitmap(imageData: Data, x: 0, y: 0, width: 200, height: 100, mode: 1) { msg in
    print(“Error message: \(msg)”)
}

// RFID 設定，例：讀寫位置 10 dot
gsble.setRFIDProcedure(tagType: 8, rw_position: 10, void_printout: 10, tryEncode_times: 3,
error_handle: “E”, speed: 2, retry_times: 6) { msg in
    print(“Error message: \(msg)”)
}
```

```
// RFID 設定，例：讀寫位置 10 mm

gsble.setRFIDProcedure(tagType: 8, rw_position: 10, void_printout: 10, tryEncode_times: 3,
error_handle: “E”, speed: 2, retry_times: 6, dpi: “203”) { msg in

    print(“Error code: \(msg)”)

}


// RFID Tag 校準動作

gsble.rfid_calibration() { msg in

    print(“Error message” \(msg)” )

}


//  將 RFID 設定值初始化

gsble.rfidSetupDefault()


        //UHF GEN2 EPC 區資料上鎖帶 L, 解鎖帶 U

            gsble.EPCPWD_Action(action: "L", password: "12345678")

        //UHF GEN2 TID 區資料上鎖帶 L, 解鎖帶 U

            gsble.TIDPWD_Action(action: "L", password: "12345678")
```

```
//UHF GEN2 USER 區資料上鎖帶 L, 解鎖帶 U

    gsble.USERPWD_Action(action: "L", password: "12345678")

//UHF GEN2 存取密碼 上鎖帶 L, 解鎖帶 U

    gsble.accessPWD_Action(action: "L", password: "12345678")

//UHF GEN2 刪除密碼 上鎖帶 L, 解鎖帶 U

    gsble.killPWD_Action(action: "L", password: "12345678")



//UHF GEN2 設定存取密碼

    gsble.accessPWD_Action(action: "S", password: "12345678")

//UHF GEN2 設定刪除密碼

    gsble.killPWD_Action(action: "S", password: "12345678")

//UHF GEN2 寫入資料

    gsble.clearBuffer()

    gsble.writeUHF(dataFormat: "H", startBlockNo: 2, byteSize: 12, Gen2MemoryBank:
"E", dataString: "414142424343444445454646") { msg in

        self.textView.text = msg // 代表未與印表機連線

    }

    gsble.printLabel(set: 1, copy: 1)
```

```
// UHF GEN2 讀取資料

    gsble.readUHF(dataFormat: "H", startBlockNo: 0, byteSize: 12, Gen2MemoryBank:
"E") { msg in

    if msg == "Connection doesn't exist." {} //  代表未與印表機連線

    else if msg == “time_out { } //逾時

    else {}//  其他回傳，代表讀取到的資料



// UHF GEN2 Q 指令讀取資料

gsble.query_UHF(dataFormat: "H", PCReturnStatus: 1, CRCReturnStatus: 1) { msg in

    if msg == Connection doesn't exist."    {

            self.textView.text = msg //  代表未與印表機連線

            return

    } else if msg == “time_out” {

            self.textView.text = “time out” //逾時

            return

    } else {

            Self.textView.text = msg //  其他回傳，代表讀取到的資料
```

```
    }
}

// UHF GJB 設定密碼

    // 帶入寫入密碼，設定新的讀取密碼

    gsble.set_GJB_Pwd_Action(passwordArea: "R", newPassword: "87654321",
writePassword: "12345678")

    // 帶入寫入密碼，設定新的寫入密碼

    gsble.set_GJB_Pwd_Action(passwordArea: "W", newPassword: "87654321",
writePassword: "12345678")

    // 帶入寫入密碼，設定新的刪除密碼

    gsble.set_GJB_Pwd_Action(passwordArea: "K", newPassword: "87654321",
writePassword: "12345678")

    // 帶入寫入密碼，設定新的狀態密碼

    gsble.set_GJB_Pwd_Action(passwordArea: "S", newPassword: "87654321",
writePassword: "12345678")

// UHF GJB 帶入狀態密碼，設定不同資料區塊的寫入讀取狀態

    gsble.statusGJB_UHF(GJBMemoryBank: "E", action: "C", statusPassword:
"11112222")

// UHF GJB 帶入寫入密碼，將資料寫入指定資料區塊
```

```
        gsble.clearBuffer()

        gsble.writeGJB_UHF(dataFormat: "H", startBlockNo: 1, byteSize: 12,
    Gen2MemoryBank: "E", dataString: "40404141424243434444454 5", writePassword:
    "12345678") { msg in

            self.textView.text = msg    // 代表未與印表機連線

        }

        gsble.printLabel(set: 1, copy: 1)

    // UHF GJB 帶入讀取密碼，讀取資料

        gsble.readGJB_UHF(dataFormat: "H", startBlockNo: 0, byteSize: 12,
    Gen2MemoryBank: "E", readPassword: "33334444") { msg in

        if msg == "Connection doesn't exist." {} // 代表未與印表機連線

        else if msg == “time_out” {} // 逾時

        else { } // 其他回傳，代表讀取到的資料

    // UHF GJB 帶入刪除密碼，刪除標籤

        gsble.killGJB_Tag_UHF(killPassword: "11224455")

// 開啟實時命令

    gsble.setRealTimeStatus("1")

    isRealTimeOn = true    // 實時命令設定為 true

    timer = Timer.scheduledTimer(withTimeInterval:0.5, repeats: true, block: { _ in // 開始計
```

```
時，每 0.5 秒讀取一次
            if let statusNumber = self.gsble.getStatusNumber() {
                var statusStr = ""
                switch statusNumber {
                case 0:    statusStr = "Normal"
                // 以下略……。取得狀態代碼，對照表格取得目前狀態。
                self.realTimeShowLabel.text = statusStr // 顯示在畫面UILabel
// 關閉實時命令
        gsble.setRealTimeStatus("0")
        isRealTimeOn = false    // 實時命令設定為 false
        timer.invalidate() // 關閉計時器

// 讀取錯誤代碼清單
        let errorCode = gsble.getRFIDErrorCode()
```

1.匯入 GTSPL_SDK：

```
import IOS_SWIFT_WIFI_SDK
```

2.範例程式：

```
//初始化 SDK
    let gswifi = GTSPL_WIFI()
//連線 WIFI 裝置
    let ip = IPText.text
    let port:Int? = Int(PortText.text!)
    gswifi.openport(IP:ip!, Port:port!){(msg) in
        print(msg)
    }
//WIFI 裝置斷線
    gswifi.closePort()
//設定印表機
    gswifi.setup(width: 105, height: 80, speed: 4, density: 6, sensor: 0, sensorDistance:
3, sensorOffset: 3)
//列印 barcode、text、bitmap
```

```
//先清除資料、再列印指定項目

gswifi.clearBuffer()

//barcode

gswifi.printBarcode(x: 30, y: 30, type: "128", height: 100, human_readable: 1, rotation: 0, narrow:2, wide: 2, content: "barcode987654321")



//text

gswifi.printFont( x: 100, y: 180, fontname: "3", rotation: 0, xScale: 3, yScale: 3, content: "12345678 print test")

//區塊文字

let paragraph="We stand behind our products with one of the most comprehensive support programs in the Auto-ID industry."

gswifi.printFontBlock(x:15,y:15,width:790,height:90,fontname:"1",rotation:0,xScale:8,yScale:8,space:20,align:0,content:paragraph)



//指定 bitmap 的檔案 url

let url = Bundle.main.url(forResource: "CIRCLE", withExtension:"bmp")!

//傳送 bitmap 檔案到印表機中
```

```
        gswifi.downloadBMP(filePath: url, fileName: "CIRCLE.bmp")

        //將 bitmap 圖片排入列印順序指令

        gswifi.sendCommand("PUTBMP 30,30,\"CIRCLE.bmp\"\r\n")

        //執行列印動作

        gswifi.printLabel(set: 1, copy: 1)



//列印 QRcode

        //先清除資料

        gswifi.clearBuffer()

        // 產生 QRCode

        gswifi.printQRCode(x: 50, y: 50, eccLevel: "H", cellWidth: 4, rotation: 0, content:
"QRcode987654321")

        gswifi.printLabel(set: 1, copy: 1)

//簡中列印

        String stString="默认简体中文测试";

        gswifi.clearBuffer()

        gswifi.printFont(x: 200, y: 200, fontName: "TSS24.BF2", rotation: 0, x_scale: 1,
y_scale: 1, content: cn)
```

```
        gswifi.printLabel(set: 1, copy: 1)

//繁中列印

        cn: String = "預設繁體中文測試"

        gswifi.clearBuffer()

        gswifi.printFont(x: 200, y: 200, fontName: "TST24.BF2", rotation: 0, x_scale: 1,
y_scale: 1, content: cn)

        gswifi.printLabel(set: 1, copy: 1)



//取得 sdk 版本

        let version = gswifi.getSdkVersion()

//取得印表機狀態

        let status = gswifi.printerStatus()

        var statusStr = ""

         switch status{

              case "00":

                   statusStr = "Normal"

              case "01":

                   statusStr = "Head opened"
```

```
case "02":
    statusStr = "Paper Jam"
case "03":
    statusStr = "Paper Jam and head opened"
case "04":
    statusStr = "Out of paper"
case "05":
    statusStr = "Out of paper and head opened"
case "08":
    statusStr = "Out of ribbon"
case "09":
    statusStr = "Out of ribbon and head opened"
case "0A":
    statusStr = "Out of ribbon and paper jam"
case "0B":
    statusStr = "Out of ribbon, paper jam and head opened"
case "0C":
    statusStr = "Out of ribbon and out of paper"
```

```
            case "0D":
                statusStr = "Out of ribbon, out of paper and head opened"
            case "10":
                statusStr = "Pause"
            case "20":
                statusStr = "Printing"
            case "80":
                statusStr = "Other error"
            default:
                break
            }
```

```
// 出紙方向、鏡像列印
gswifi.setDirectionAndMirror(direction: 0, mirror: 0)
```

```
// 圖像垂直位移距離 例：偏移 36 dot
gswifi.setShift(shiftY: 36)
```

```
// 反白列印（輸入區域）

gswifi.printReverse(x_start: 0, y_start: 0, x_width: 100, y_height: 100)


// 出紙偏移 例：9.9mm

gswifi.setOffset(offset: 9.9)


// 設定裁切模式與張數 例：正切、三張

gswifi.setCutMode(mode: 1, piece: 3)


// 設定列印後動作 例：撕紙

gswifi.setAfterPrintAction(mode: 1)


// 一般設定值初始化

gswifi.genericDefault()


// 感應器設定值初始化

gswifi.sensorDefault()
```

```
// 切換 wifi 2.4G/5G/Both，例：2.4G

gswifi.switchWifiFrequency(frequency: “2.4G”) { msg in

    print(“Error message: \(msg)”)

}


// 列印圖片，轉單色，例：模式 1, 自訂大小為 200 dot X 100 dot

gswifi.printBitmap(imageData: Data, x: 0, y: 0, width: 200, height: 100, mode: 1) { msg in

    print(“Error message: \(msg)”)

}


// RFID 設定，例：讀寫位置 10 dot

gswifi.setRFIDProcedure(tagType: 8, rw_position: 10, void_printout: 10, tryEncode_times: 3,
error_handle: “E”, speed: 2, retry_times: 6) { msg in

    print(“Error message: \(msg)”)

}


// RFID 設定，例：讀寫位置 10 mm

gswifi.setRFIDProcedure(tagType: 8, rw_position: 10, void_printout: 10, tryEncode_times: 3,
```

```
error_handle: “E”, speed: 2, retry_times: 6, dpi: “203”) { msg in
        print(“Error code: \(msg)”)
}

// RFID Tag 校準動作
gswifi.rfid_calibration() { msg in
        print(“Error message” \(msg)” )
}

//  將 RFID 設定值初始化
gswifi.rfidSetupDefault()

            //UHF GEN2 EPC 區資料上鎖帶 L,  解鎖帶 U
                gswifi.EPCPWD_Action(action: "L", password: "12345678")
            //UHF GEN2 TID 區資料上鎖帶 L,  解鎖帶 U
                gswifi.TIDPWD_Action(action: "L", password: "12345678")
            //UHF GEN2 USER 區資料上鎖帶 L,  解鎖帶 U
```

```
        gswifi.USERPWD_Action(action: "L", password: "12345678")
//UHF GEN2 存取密碼 上鎖帶 L, 解鎖帶 U
        gswifi.accessPWD_Action(action: "L", password: "12345678")
//UHF GEN2 刪除密碼 上鎖帶 L, 解鎖帶 U
        gswifi.killPWD_Action(action: "L", password: "12345678")


//UHF GEN2 設定存取密碼
        gswifi.accessPWD_Action(action: "S", password: "12345678")
//UHF GEN2 設定刪除密碼
        gswifi.killPWD_Action(action: "S", password: "12345678")
//UHF GEN2 寫入資料
        gswifi.clearBuffer()
        gswifi.writeUHF(dataFormat: "H", startBlockNo: 2, byteSize: 12, Gen2MemoryBank:
"E", dataString: "414142424343444445454646") { msg in
            self.textView.text = msg   // 代表未與印表機連線
        }
        gswifi.printLabel(set: 1, copy: 1)
// UHF GEN2 讀取資料
```

```
gswifi.readUHF(dataFormat: "H", startBlockNo: 0, byteSize: 12, Gen2MemoryBank: "E") { msg in

    if msg == "Connection doesn't exist." {} // 代表未與印表機連線

    if msg == “time_out” {} // 逾時

    else { }// 其他回傳，代表讀取到的資料

// UHF GJB 設定密碼

    // 帶入寫入密碼，設定新的讀取密碼

    gswifi.set_GJB_Pwd_Action(passwordArea: "R", newPassword: "87654321", writePassword: "12345678")

    // 帶入寫入密碼，設定新的寫入密碼

    gswifi.set_GJB_Pwd_Action(passwordArea: "W", newPassword: "87654321", writePassword: "12345678")

    // 帶入寫入密碼，設定新的刪除密碼

    gswifi.set_GJB_Pwd_Action(passwordArea: "K", newPassword: "87654321", writePassword: "12345678")

    // 帶入寫入密碼，設定新的狀態密碼

    gswifi.set_GJB_Pwd_Action(passwordArea: "S", newPassword: "87654321", writePassword: "12345678")

// UHF GJB 帶入狀態密碼，設定不同資料區塊的寫入讀取狀態
```

```
        gswifi.statusGJB_UHF(GJBMemoryBank: "E", action: "C", statusPassword:
"11112222")

    // UHF GJB 帶入寫入密碼，將資料寫入指定資料區塊

        gswifi.clearBuffer()

        gswifi.writeGJB_UHF(dataFormat: "H", startBlockNo: 1, byteSize: 12,
Gen2MemoryBank: "E", dataString: "404041414242434344444545", writePassword:
"12345678") { msg in

            self.textView.text = msg    {}    // 代表未與印表機連線

        }

        gswifi.printLabel(set: 1, copy: 1)

    // UHF GJB 帶入讀取密碼，讀取資料

        gswifi.readGJB_UHF(dataFormat: "H", startBlockNo: 0, byteSize: 12,
Gen2MemoryBank: "E", readPassword: "33334444") { msg in

        if msg == "Connection doesn't exist." {} // 代表未與印表機連線

        if msg == “time_out” {} // 逾時

        else { } // 其他回傳，代表讀取到的資料

    // UHF GJB  帶入刪除密碼，刪除標籤

        gswifi.killGJB_Tag_UHF(killPassword: "11224455")

// 開啟實時命令
```

```
gswifi.setRealTimeStatus("1")

isRealTimeOn = true    // 實時命令設定為 true

timer = Timer.scheduledTimer(withTimeInterval:0.5, repeats: true, block: { _ in // 開始計時，每 0.5 秒讀取一次

    if let statusNumber = self.gsble.getStatusNumber() {

        var statusStr = ""

        switch statusNumber {

        case 0:    statusStr = "Normal"

        // 以下略……。取得狀態代碼，對照表格取得目前狀態。

        self.realTimeShowLabel.text = statusStr // 顯示在畫面UILabel

// 關閉實時命令

gswifi.setRealTimeStatus("0")

isRealTimeOn = false    // 實時命令設定為 false

timer.invalidate() // 關閉計時器


// 讀取錯誤代碼清單

let errorCode = gswifi.getRFIDErrorCode()
```

Code Type	Description	Narrow : Width					Max. data length
		1:1	1:2	1:3	2:5	3:7	
128	Code 128, switching code subset automatically.	V					
128M	Code 128, switching code subset manually.	V					
EAN128	EAN128, switching code subset automatically.	V					
EAN128M	EAN128M, switching code subset manually.	V					
25	Interleaved 2 of 5.		V	V	V		Length is even
25C	Interleaved 2 of 5 with check digit.		V	V	V		Length is odd
25S	Standard 2 of 5.		V	V	V		
25I	Industrial 2 of 5.		V	V	V		
39	Code 39, switching standard and full ASCII mode automatically.		V	V	V		
39C	Code 39 with check digit.		V	V	V		
93	Code 93.			V			
EAN13	EAN 13.	V					12
EAN13+2	EAN 13 with 2 digits add-on.	V					14
EAN13+5	EAN 13 with 5 digits add-on.	V					17
EANB	EAN 8.	V					7
EANB+2	EAN 8 with 2 digits add-on.	V					96
EANB+5	EAN 8 with 5 digits add-on.	V					12
CODA	Codabar.		V	V	V		

POST	Postnet.	V					5,9,11
UPCA	UPC-A.	V					11
UPCA+2	UPC-A with 2 digits add-on.	V					13
UPA+5	UPC-A with 5 digits add-on.	V					16
UPCE	UPC-E.	V					6
UPCE+2	UPC-E with 2 digits add-on.	V					8
UPE+5	UPC-E with 5 digits add-on.	V					11
MSI	MSI.		V	V	V		
MSIC	MSI with check digit.		V	V	V		
PLESSEY	PLESSEY.		V	V	V		
CPOST	China post.					V	
ITF14	ITF14.		V	V	V		13
EAN14	EAN14.	V					13
11	Code 11.		V	V	V		
TELEPEN	Telepen. *Since V6.89EZ.		V	V	V		
TELEPENN	Telepen number. *Since V6.89EZ.		V	V	V		
PLANET	Planet. *Since V6.89EZ.	V					
CODE49	Code 49. *Since V6.89EZ.	V					
DPI	Deutsche Post Identcode. *Since V6.91EZ.		V	V	V		11
DPL	Deutsche Post Leitcode. *Since V6.91EZ.		V	V	V		13
LOGMAR	A special use of Code 39. *Since		V	V	V		

S	V6.88EZ.						