iOS SWIFT SDK

说明手册(V1.6)



iOS Swift Bluetooth/WIFI 函式库使用说明

1. searchBLEDevice(callback)

- 函式说明：搜寻蓝牙 BLE 装置并回传
- 参数说明：
  - ➔ **callback(btDictionary: [String: CBPeripheral])**：函式，回传所有搜寻到的 BLE 装置，如：[“Printer_7544”: CBPeripheral, “Printer_7534”: CBPeripheral]

2. openPort()

★ Bluetooth

openPort (connectDecive, callback)

- 函式说明：指定蓝牙的装置和回传状态函式后联机
- 参数说明：
  - ➔ **connectDecive**：CBPeripheral 型别，指定联机的蓝牙 Peripheral 装置
  - ➔ **callback(msg: String)**：函式，回传联机状态

回传字符串	
“disconnected”	已断线
“connecting”	联机中
“connected”	已联机
“disconnecting”	断线中
“time_out”	逾时

★ WIFI

openPort(IP, port, callback)

- 函式说明：指定打印机的 IP 地址与端口和回传状态函式后，开启输出埠
- 参数说明：
  - ➔ **IP：** String 型别，指定联机的 IP 地址 ，如：”192.168.1.109”
  - ➔ **port：** Int 型别，指定联机的端口如：8899
  - ➔ **callback(msg: String)：** 函式，回传联机状态

回传字符串	
“connecting”	联机中
“connected”	已联机
“time_out”	逾时

3. closePort()

★ Bluetooth

closePort (callback)

- 函式说明：蓝牙断线
- 参数说明：
  - ➔ **connectDecive:** CBPeripheral 型别，指定已经联机的蓝牙 Peripheral 装置
  - ➔ **callback(msg: String)：**函式，回传断线讯息“disconnected”

★ WIFI

closePort (callback)

- 函式说明：关闭输出埠
- 参数说明：
  - ➔ **callback(msg: String)**：函式，回传断线讯息“disconnected”

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

- 函式说明：设定标签的宽度、高度、打印速度、打印热度、传感器类别、间隙/黑标垂直间距、间隙/黑标偏移距离
- 参数说明：

参数	型别	说明
width	Int	设定标签宽度，单位 mm
height	Int	设定标签高度，单位 mm
speed	Int	设定打印速度，1~15，代表每秒 1~15 吋打印速度(随机型不同会有不同打印最高上限，最高为每秒 15 吋打印速度)
density	Int	设定打印浓度，0~15，数字越大打印结果越黑
sensor	Int	设定使用传感器之类别； 0：表示使用间隙传感器(gap sensor) 1：表示使用黑标传感器(black mark sensor)
sensorDistance	Int	设定间隙/黑标垂直间距高度，单位 mm
sensorOffset	Int	设定间隙/黑标垂直间距高度，单位 mm，此参数若使用一般标签时均设为 0
callback	函式	未连机时回传”Connection doesn't exist.”

5. clearBuffer()

- 函式说明：清除图像缓冲

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

- 函式说明：使用打印机内建条形码打印
- 参数说明：

参数	型别	说明
x	Int	条形码 X 方向起始点，以点(dot)表示
y	Int	条形码 Y 方向起始点，以点(dot)表示
type	String	设定条形码类型(Code Type) ，请参考附件
height	Int	设定条形码高度，高度以点来表示

readable	Int	设定是否打印条形码码文 0：不打印 1：打印条形码码文置左 2：打印条形码码文置中 3：打印条形码码文置右
rotation	Int	设定条形码旋转角度 0：旋转0度 90：旋转90度 180：旋转180度 270：旋转270度
narrow	Int	设定条形码窄 bar 比例因子，请参考附件
wide	Int	设调条形码宽 bar 比例因子，请参考附件
content	String	设定欲打印之条形码内容
callback	函式	未连机时回传”Connection doesn't exist.”

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

- 函式说明：使用打印机内建二维码打印
- 参数说明：

参数	型别	说明
x	Int	二维码 X 方向起始点，以点(dot)表示
y	Int	二维码 Y 方向起始点，以点(dot)表示
eccLevel	String	容错等级，L： 7%, M： 15%, Q： 25%, H： 30%
cellWidth	Int	设定二维码长宽，设定范围为 1~10
rotation	Int	設定二维码旋转角度 0：旋转0度 90：旋转90度 180：旋转180度 270：旋转270度
content	String	设定欲打印之二维码内容
callback	函式	未连机时回传”Connection doesn't exist.”

8. formFeed()

- 函式说明：跳页，该函式需在 setup 后使用

9. noBackFeed()

- 函式说明：设定纸张不回吐

10. sendCommand (_ string)

- 函式说明：送内建指令到打印机
- 参数说明：
  - ➔ **string**：String 型别，设定指令内容，详细指令请参考 TSPL 使用手册

11. sendByteCmd(cmdData)

- 函式说明：传送 byte 形式的资料到打印机
- 参数说明：
  - ➔ **cmdData**：Data 型别

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

- 函式说明：使用打印机内建文字打印
- 参数说明：

参数	型别	说明
x	Int	文字 X 方向起始点，以点(dot)表示
y	int	文字 Y 方向起始点，以点(dot)表示
fontName	String	内建字型名称，共五种 1： 8*/12 dots 2： 12*20 dots 3： 16*24 dots 4： 24*32 dots 5： 32*48 dots TST24.BF2： 繁体中文24*24 dots

		TST16.BF2： 繁体中文16*16 dots TSS24.BF2： 简体中文24*24 dots TSS16.BF2： 简体中文16*16 dots
rotation	Int	设定文字旋转角度 0：旋转0度 90：旋转90度 180：旋转180度 270：旋转 270 度
x_scale	Int	设定文字 X 方向放大倍率，1~10
y_scale	Int	设定文字Y方向放大倍率，1~10
content	String	设定欲打印之文字内容
callback	函式	未连机时回传”Connection doesn't exist.”

13. printLabel(set, copy)

- 函式说明：打印标签内容
- 参数说明：
  - ➔ **set**：int 型别，设定打印标签式数(set)
  - ➔ **copy**：int 型别，设定打印标签份数(copy)

14. downloadPCX(filePath, fileName)

- 函式说明：下载单色 PCX 格式图文件至打印机
- 参数说明：
  - ➔ **filePath**：URL 类别，带入 PCX 文件所在位置
  - ➔ **fileName**：String 型别，文件名(须包含扩展名)
  - ➔ **callback(msg: String)**：函式，未连机时回传”Connection doesn't exist.”

15. downloadBMP(filePath, fileName)

- 函式说明：下载单色 BMP 格式图文件至打印机
- 参数说明：
  - ➔ **filePath**：URL 类别，带入 BMP 文件所在位置
  - ➔ **fileName**：String 型别，文件名(须包含扩展名)
  - ➔ **callback(msg: String)**：函式，未连机时回传”Connection doesn't exist.”

16. getSdkVersion ( )

- 函式说明：回传此 SDK 版本号
- 回传说明：回传版本号(String)

17. printerStatus(callback)

- 函式说明：带入回传状态函式后回传打印机状态
- 参数说明：
  - ➔ **callback(status: String)**：函式，回传打印机状态
- 回传字符串说明：

回传字符串	打印机状态
00	就绪
01	上盖开启
02	卡纸
03	卡纸且上盖开启
04	标签用尽
05	标签用尽且上盖开启
08	碳带用尽
09	碳带用尽且上盖开启
0A	碳带用尽且卡纸
0B	碳带用尽、卡纸且上盖开启
0C	碳带用尽且标签用尽
0D	碳带用尽、标签用尽且上盖开启

10	暂停
20	打印中
80	其他错误
time_out	逾时
Connection doesn't exist.	未连线

18. setDirectionAndMirror(direction,mirror)

- 函式说明：设定标签打印时的出纸方向与是否使用镜像打印
- 参数说明：

参数	型别	说明
direction	int	设定出纸方向，预设为 0 0：顶端出纸 1：底端出纸
mirror	int	设定是否镜像打印 0：否 1：是

19. setShift(shiftY)

- 函式说明：设定图像垂直位移距离，数值为正时，图像会往打印方向移动，数值为负时，图像会背离打印方向
- 参数说明：
  - ➔ **shiftY**：int 型别，垂直位移距离，单位为 dot

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

- 函式说明：将指定的区域于打印时反白
- 参数说明：

参数	型别	说明
x_start	int	指定 X 起始坐标位置，以点(dot)表示
y_start	int	指定 Y 起始坐标位置，以点(dot)表示
x_width	int	指定 X 坐标宽度，以点(dot)表示
y_height	int	指定 Y 坐标高度，以点(dot)表示

21. setOffset(offset)

- 函式说明：设定每次出纸后额外偏移的距离(通常与剥纸模式和裁切模式组合使用)
- 参数说明：
  - ➔ **offset**：double 型别，额外的出纸偏移，单位为 mm

22. setCutMode(mode, piece)

- 函式说明：设定裁切模式与张数
- 参数说明：

参数	型别	说明
mode	int	设定裁切方式，预设为 1 0：反切 1：正切
piece	int	设定裁切张数

23. setAfterPrintAction(mode)

- 函式说明：设定打印后动作
- 参数说明：

参数	型别	说明
mode	int	设定打印后动作，预设为 1 0：停在原地 1：撕纸 2：剥纸 3：裁切

24. genericDefault()

- 函式说明：将打印机之一般设定值初始化
- 参数说明：无

25. sensorDefault()

- 函式说明：将打印机之传感器设定值初始化
- 参数说明：无

26. switchWifiFrequency(frequency, callback)

- 函式说明：使用兼容 5G 频段 WIFI 模块时，可用于切换使用频段
- 参数说明：
  - ➔ Frequency：字符串型别，设定模块频段；2.4G：使用 2.4G 频段、5G：使用 5G 频段、 BOTH：使用双频频段
  - ➔ callback： 未连机时回传”Connection doesn't exist.”

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

- 函式说明：列印圖片檔案(僅支援 jpg, png)
- 参数说明：

参数	型别	说明
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	未连机时回传”Connection doesn't exist.”

28. setRealTimeStatus(status)

- 函式说明：开启或关闭实时命令，打印机第一次开机预设状态为关闭
- 参数说明：

参数	型别	说明
Status	String	实时命令开启状态 0： 关闭状态 1： 开启状态
callback	函式	未连线时回传：”Connection doesn't exist.”

29. getRealTimeStatus(callback)

- 函式说明：读取实时命令的开启或关闭状态
- 参数说明：

  ➔ callback(msg： String)：函式，回传读取的资料或显示逾时

- 回传说明：

回传	型别	说明
0	String	实时命令关闭状态回传 FB：0\r\n
1	String	实时命令关闭状态回传 FB：1\r\n
time_out	String	逾时
Connection doesn't exist.	String	未连线

30. getStatusNumber() -> String

- 函式说明：读取实时状态数字
- 参数说明：无
- 回传数字说明：

回传字符串	打印机状态
00	就绪
01	上盖开启
02	卡纸
03	卡纸且上盖开启
04	标签用尽
05	标签用尽且上盖开启
08	碳带用尽
09	碳带用尽且上盖开启
0A	碳带用尽且卡纸
0B	碳带用尽、卡纸且上盖开启
0C	碳带用尽且标签用尽
0D	碳带用尽、标签用尽且上盖开启
10	暂停
20	打印中
80	其他错误

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

- 函式说明：将数据写入 UHF 标签内存中
- 参数说明：

参数	型别	说明
dataFormat	String	设定 String 数据编码格式，默认为 H A：ASCII H：Hexadecimal
startBlockNo	Int	设定数据区块起始位置，默认为 2
byteSize	Int	设定写入数据byte长度，默认为1
Gen2MemoryBank	String	设定 Gen2 数据区段，默认为 E R：保留 E：EPC T：TID(Tag ID) U：User
dataString	String	欲写入之 String 数据
callback	函式	未连机时回传”Connection doesn't exist.”

2. EPCPWD_Action (action, password)

- 函式说明：将 UHF GNE2 的 EPC 资料区块上锁或解锁
- 参数说明：

参数	型别	说明
action	String	设定执行动作 U：解锁资料区块 L：上锁资料区块 O：永久解锁资料区块 P：永久上锁资料区块
password	String	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

3. TIDPWD_Action(action, password)

- 函式说明：将 UHF GNE2 的 TID 资料区块上锁或解锁
- 参数说明：

参数	型别	说明
action	String	设定执行动作 U：解锁资料区块 L：上锁资料区块 O：永久解锁资料区块 P：永久上锁资料区块
password	String	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

4. USERPWD_Action (action, password)

- 函式说明：将 UHF GNE2 的 USER 资料区块上锁或解锁
- 参数说明：

参数	型别	说明
action	String	设定执行动作 U：解锁资料区块 L：上锁资料区块 O：永久解锁资料区块 P：永久上锁资料区块
password	String	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

5. accessPWD_Action (action, password)

- 函式说明：将 UHF GNE2 的存取密码进行设定、上锁或解锁
- 参数说明：

参数	型别	说明
action	String	设定执行动作 U：解锁存取密码 L：上锁存取密码

		O：永久解锁存取密码 P：永久上锁存取密码 S：设定存取密码
password	String	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

6. killPWD_Action (action, password)

- 函式说明：将 UHF GNE2 的删除密码进行设定、上锁或解锁
- 参数说明：

参数	型别	说明
action	String	设定执行动作 U：解锁删除密码 L：上锁删除密码 O：永久删除存取密码 P：永久删除存取密码 S：设定删除密码
password	String	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

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

- 函式说明：RFID 设定
- 参数说明：

数	型别	说明
tagType	Int	设定标签类型，1~10，默认值为 8 1：EPC Class 1 Generation 2-Q , 8：EPC Class 1 Generation 2-R ,10：UHF-J
rw_position	Int	设标签读写位置(标签顶部起算)，范围为 0~9999(dot)，预设为 0
void_printout	Int	设定无效打印长度(dot)，范围为0~标签长度，默认为标签长度
tryEncode_ti	Int	设定最大无效标签数，范围为 0~10，预设为 3

mes		
error_handle	String	设定无效时采取的动作，预设为 N N：No action(继续) P：Pause mode(暂停) E：Error mode(停止)
speed	Int	设定无效打印速度，范围 2~10(IPS)，默认值 2(IPS)
retry_times	Int	设定标签重试次数，范围 0~10，默认值 6
callback	函式	未连机时回传”Connection doesn't exist.”

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

- 函式说明：RFID 设定
- 参数说明：

数	型别	说明
tagType	Int	设定标签类型，1~10，默认值为 8 1：EPC Class 1 Generation 2-Q , 8：EPC Class 1 Generation 2-R ,10：UHF-J
rw_position	Int	设标签读写位置(标签顶部起算)，范围为 0~9999(dot)，预设为 0
void_printout	Int	设定无效打印长度(dot)，范围为0~标签长度，默认为标签长度
tryEncode_times	Int	设定最大无效标签数，范围为 0~10，预设为 3
error_handle	String	设定无效时采取的动作，预设为 N N：No action(继续) P：Pause mode(暂停) E：Error mode(停止)
speed	Int	设定无效打印速度，范围 2~10(IPS)，默认值 2(IPS)
retry_times	Int	设定标签重试次数，范围 0~10，默认值 6
dpi	String	设定打印机的 DPI 203： 203 dpi 300： 300 dpi 600： 600 dpi

callback	函式	未连机时回传”Connection doesn't exist.”

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

- 函式说明：将数据写入 HF 标签内存中
- 参数说明：

参数	型别	说明
dataFormat	String	设定 String 数据编码格式，默认为 H A：ASCII H：Hexadecimal
startBlockNo	Int	设定数据区块起始位置，默认为 2
byteSize	Int	设定写入数据byte长度，默认为1
dataString	String	欲写入之 String 数据
callback	函式	未连机时回传”Connection doesn't exist.”

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

- 函式说明：打印段落文字内容
- 参数说明：

参数	型别	说明
x	Int	文字 X 方向起始点，以点(dot)表示
y	Int	文字 Y 方向起始点，以点(dot)表示
width	Int	设定段落区块宽度，以点(dot)表示
height	Int	设定段落区块高度，以点(dot)表示
fontName	String	内建字型名称 1： 8*/12 dots 2： 12*20 dots 3： 16*24 dots 4： 24*32 dots 5： 32*48 dots TST24.BF2： 繁体中文24*24 TST16.BF2： 繁体中文16*16 TSS24.BF2： 简体中文24*24

		TSS16.BF2： 简体中文16*16
rotation	Int	设定文字旋转角度 0：旋转0度 90：旋转90度 180：旋转180度 270：旋转 270 度
x_scale	Int	设定文字 X 方向放大倍率，1~10
y_scale	Int	设定文字X方向放大倍率，1~10
space	Int	行距，以点(dot)表示
align	Int	对齐位置 0：预设(置左) 1：置左 2：置中 3：置右
content	String	设定欲打印之文字内容
callback	函式	未连机时回传”Connection doesn't exist.”

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

- 函式说明：读取 UHF Gen2/UHF GJB 卷标内存数据
- 参数说明：

参数	型别	说明
dataFormat	String	设定 String 数据编码格式，默认为 H A：ASCII H：Hexadecimal
startBlockNo	Int	设定数据区块起始位置，默认为 0
byteSize	Int	设定读取数据byte长度，默认为1
Gen2MemoryBank	String	设定 Gen2/GJB 数据区段，默认为 E R：保留 E：EPC T：TID(Tag ID) U：User
callback	函式	未连线时回传：”Connection doesn't exist.” 连线时回传： 1. 标签资料

		2. “Error code： 错误代码(参考错误代码页面)” 3. “time_out” 逾时

- 回传字符串说明：

dataFormat	回传字符串(范例)
A	标签资料以 ASCII 显示 (ex： 24051324000103456400)
H	标签资料以 Hexadecimal 显示 (ex： 32343035313332343030303130333435363430 30)

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

- 函式说明：以 Q 指令读取 UHF Gen2 标签记忆体 EPC 资料区段资料，需用字符串变数接收回传讯息
- 参数说明：

参数	型别	说明
dataFormat	String	设定 String 资料编码格式，预设为 H A：ASCII H：Hexadecimal
PCReturnStatus	Int	PC 返回状态，预设为 0 0：不回传 PC 值 1：回传 PC 值
CRCReturnStatus	Int	CRC-16 返回状态，预设为 0 0：不回传 CRC-16 1：回传CRC-16
callback	函式	未连线时回传：”Connection doesn't exist.” 连线时回传： 1. 标签资料 2. “Error code： 错误代码(参考错误代码页面)” 3. “time_out” 逾时

- 回传字符串说明：

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

dataFormat	回传字符串(范例)
A	标签资料以 ASCII 显示

	(ex： 24051324000103456400)
H	标签资料以 Hexadecimal 显示 (ex： 32343035313332343030303130333435363430 30)

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

dataFormat	回传字符串(范例)
A	标签资料以 ASCII 显示 (ex： 0513240001034564)
H	标签资料以 Hexadecimal 显示 (ex： 30353133323430303031303334353634)

13. rfid_calibration(callback)

- 函式说明：执行 RFID Tag 校准动作
- 參數說明：無

14. rfidSetupDefault()

- 函式说明：将 RFID 设定值初始化
- 参数说明：无

1. **set_GJB_Pwd_Action(passwordArea, action, newPassword, writePassword)**
   - 函式说明：设定 UHF GJB 各密码区域新密码
   - 参数说明：

参数	型别	说明
passwordArea	String	设定密码区域，预设为 W K：Kill W：Write R：Read S：Status
newPassword	String	设定密码区域的新密码，应为8 hex字元(0~9,A,B,C,D,E,F)
writePassword	String	写入密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)

2. **writeGJB_UHF(dataFormat, startBlockNo, byteSize, GJBMemoryBank, dataString, writePassword, callback)**
   - 函式说明：将资料写入 UHF GJB 标签记忆体中
   - 参数说明：

参数	型别	说明
dataFormat	String	设定 String 资料编码格式，预设为 H A：ASCII H：Hexadecimal
startBlockNo	Int	设定资料区块起始位置，GJB 预设为 1
byteSize	Int	设定写入资料byte长度，预设为1
GJBMemoryBank	String	设定 GJB 资料区段，预设为 E R：安全区 E：EPC T：TID(Tag ID) U：User
dataString	String	欲写入之 String 资料
writePassword	String	写入密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)
callback	函式	未连线时回传：”Connection doesn't exist.”

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

- 函式说明：读取 UHF GJB 标签记忆体资料，需用字符串变数接收回传讯息
- 参数说明：

参数	型别	说明
dataFormat	String	设定 String 资料编码格式，预设为 H A：ASCII H：Hexadecimal
startBlockNo	Int	设定资料区块起始位置，预设为 0
byteSize	Int	设定写入资料byte长度，预设为1
GJBMemoryBank	String	设定 GJB 资料区段，预设为 E R：安全区 E：EPC T：TID(Tag ID) U：User
readPassword	String	读取密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)
callback	函式	未连线时回传：”Connection doesn't exist.” 连线时回传： 1. 标签资料 2. “Error code： 错误代码(参考错误代码页面)” 3. “time_out” 逾时

- 回传字符串说明：

dataFormat	回传字符串(范例)
A	标签资料以 ASCII 显示 (ex： 24051324000103456400)
H	标签资料以 Hexadecimal 显示 (ex： 32343035313332343030303130333435363430 30)

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

- 函式说明：设定 UHF GJB 各资料区块读写状态
- 参数说明：

参数	型别	说明
GJBMemoryBank	String	设定 GJB 资料区段，预设为 E F：安全区 E：EPC T：TID(Tag ID) U：User
action	String	设定状态，预设为 A A=Lock0(可读可写) B=Lock1(可读不可写) C=Lock2(不可读可写) D=Lock3(不可读不可写)
statusPassword	String	状态密码，应为8 hex字元(0~9,A,B,C,D,E,F)
callback	函式	未连线时回传：”Connection doesn't exist.”

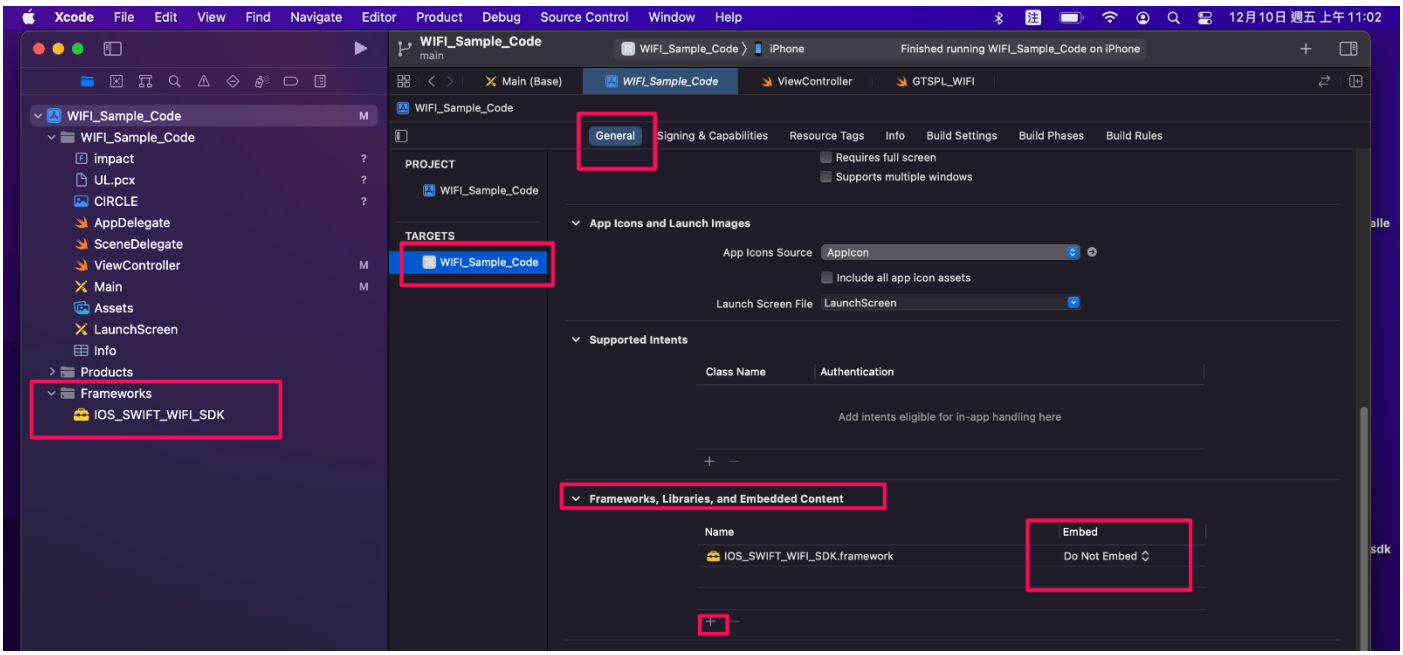
5. kill_GJB_Tag_UHF(KillPassword)

- 函式说明：删除 UHF GJB 标签
- 参数说明：

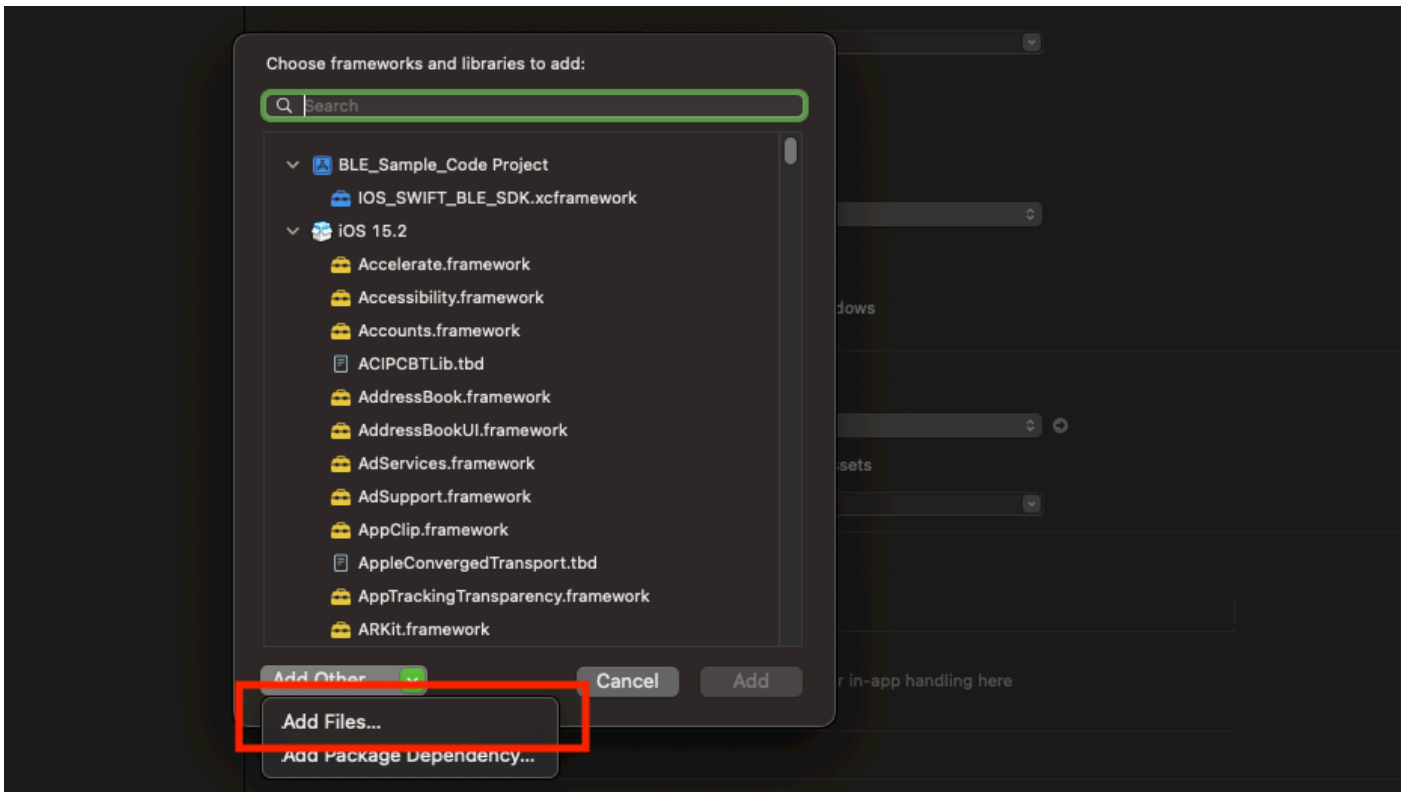
参数	型别	说明
killPassword	String	删除密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)

Swift SDK 导入说明

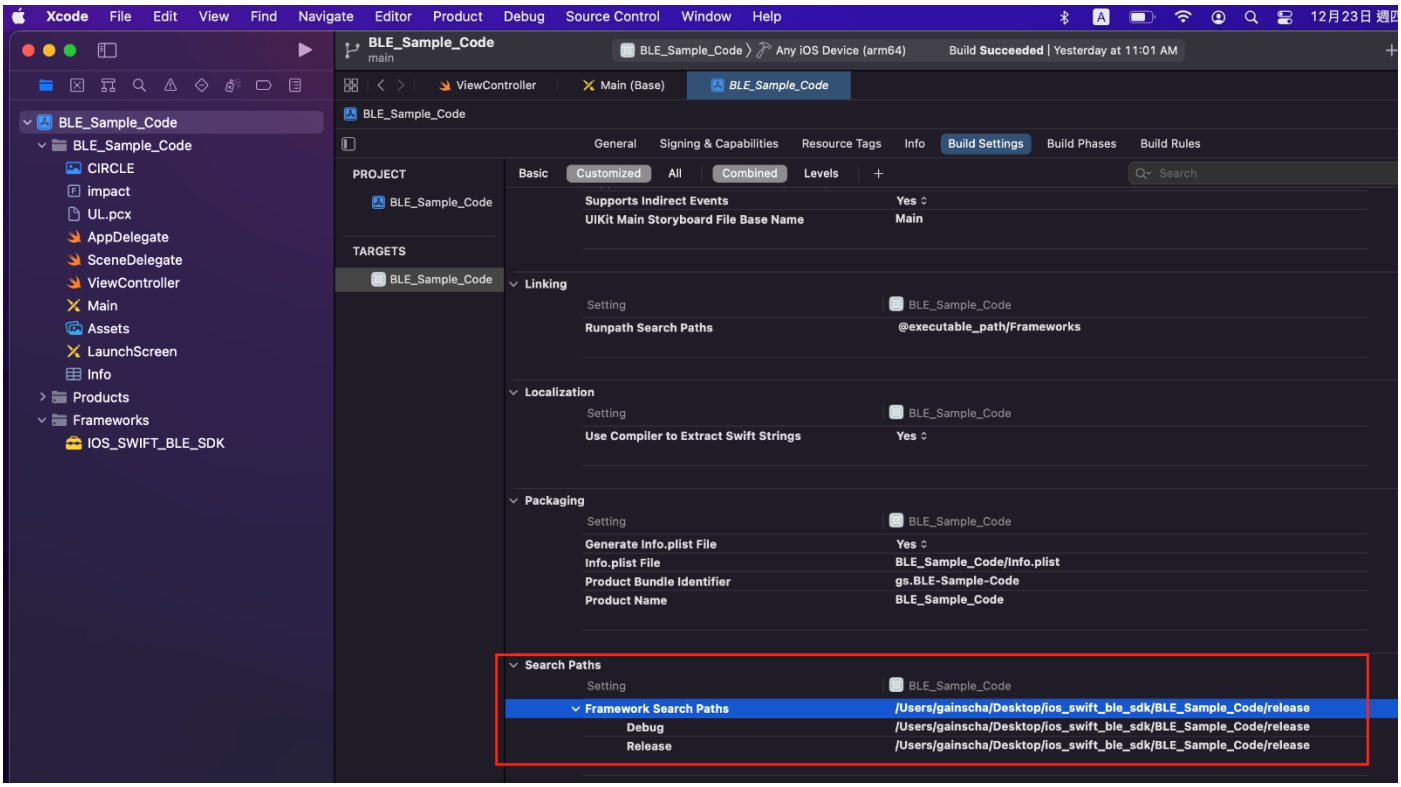
1.导入 Framework：Target => General => Frameworks, Libraires, and Embedded Content 中加入 WIFI 和 BLE XCFramework，注意 Embed 要选择 Do Not Embed



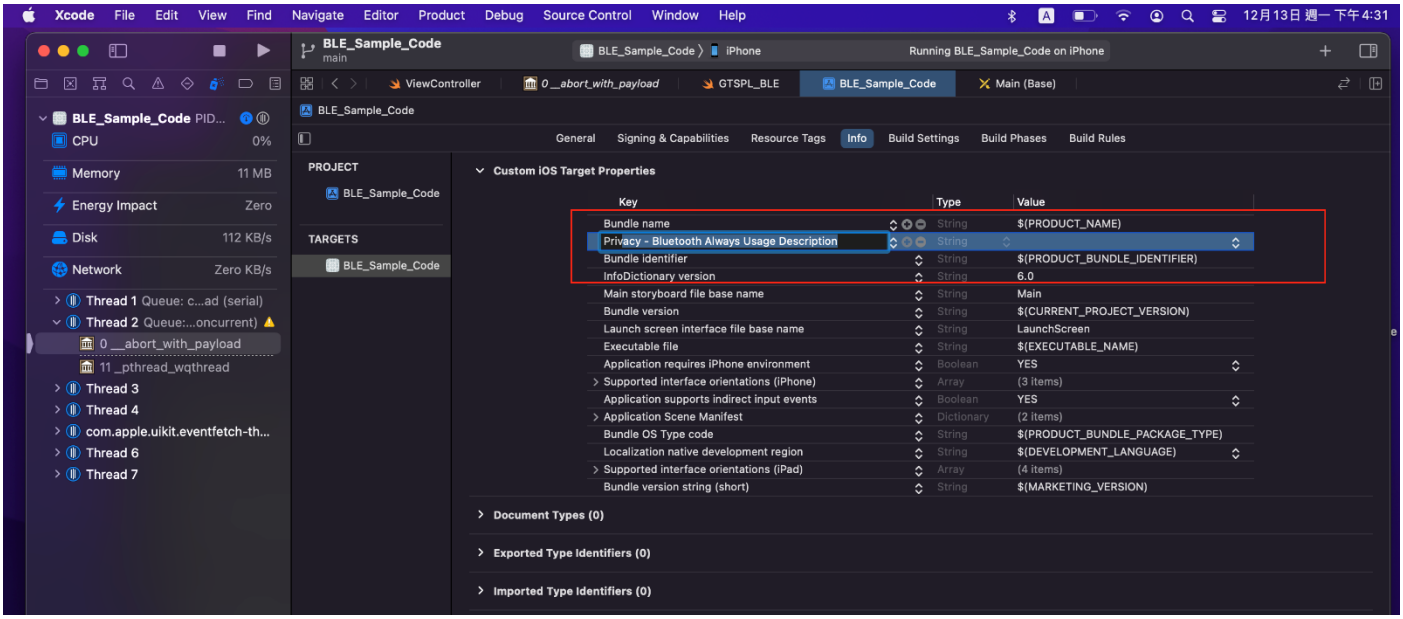
2. 选 add files，并选择 .xcframework 的资料，按 open 键汇入



3.设定 Search Paths：Target => Building Settings => Combined => Search Paths，设定 Framework 所在路径



1.需先于 Target=> Info 设定 Properties：Privacy – Bluetooth Always Usage Description



2.汇入 GTSPL_SDK：

```
import IOS_SWIFT_BLE_SDK
```

3.范例程序：

```
//初始化 SDK
    let gsble = GTSPL_BLE()
//纪录蓝牙设备的阵列
    var BTDictionary = [String：  CBPeripheral]()
//所选的蓝牙设备名称
    var mBleName：String?
//搜寻 BLE 装置
    gsble.searchBLEDevice(){(btDictionary：  [String：  CBPeripheral]) in
        self.BTDictionary = btDictionary
     }
//BLE  装置名称
    let btdevice = BTDictionary[mBleName!]
```

```
//联机 BLE 装置
    gsble.openportBLE(connectDevice： btdevice!){(msg) in
        self.mTextView.text = msg
    }
//BLE 装置断线
    gsble.closePort(){(msg) in
            self.textView.text = msg
      }
//设定打印机
    gsble.setup(width： 105, height： 80, speed： 4, density： 6, sensor： 0,
sensordistance： 5, sensoroffset：5) { msg in
            self.textView.text = msg
    }
//打印 barcode、text、bitmap、区段文字
    //先清除资料、再打印指定项目
    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")
    //区段文字
    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 档案 路径
        let url = Bundle.main.url(forResource： "CIRCLE", withExtension："bmp")!
        //传送 bitmap 档案到打印机中
        gsble.downloadBMP(filepath： url, filename： "CIRCLE.bmp")
        //将 bitmap 图片排入打印顺序指令
        gsble.sendcommand("PUTBMP 30,30,\"CIRCLE.bmp\"\r\n")
        //打印
        gsble.printLabel(set： 1, copy： 1)

//打印二维码
        //先清除资料
        gsble.clearBuffer()
        //产生二维码
        gsble.printQRCode(x： 50, y： 50, eccLevel： "H", cellWidth： 4, mode： "A",
rotation： 0, content： "QRcode987654321")
        // 打印
        gsble.printLabel(set： 1, copy： 1)

//简中打印
        String stString="默认简体中文测试";
        gsble.clearBuffer()
        gsble.printFont(x： 200, y： 200, fontname： "TSS24.BF2", rotation： 0, xScale：
1, yScale： 1, content： cn)
        gsble.printLabel(set： 1, copy： 1)
```

```
//繁中打印
    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()
//取得打印机状态
    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
        }
    }

// 出纸方向、镜像列印
gsble.setDirectionAndMirror(direction： 0, mirror： 0)
```

```
// 图像垂直位移距离 例：偏移 36 dot
gsble.setShift(shiftY： 36)

// 反白列印（输入区域）
gsble.printReverse(x_start： 0, y_start： 0, x_width： 100, y_height： 100)

// 出纸偏移 例：9.9mm
gsble.setOffset(offset： 9.9)

// 设定裁切模式与张数 例：正切、三张
gsble.setCutMode(mode： 1, piece： 3)

// 设定列印后动作 例：撕纸
gsble.setAfterPrintAction(mode： 1)

// 一般设定值初始化
gsble.genericDefault()

// 感应器设定值初始化
gsble.sensorDefault()

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

```
// 列印图片，转单色，例：模式 1, 自订大小为 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 设定，例：读写位置 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 设定，例：读写位置 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 校准动作
gsble.rfid_calibration() { msg in
    print(“Error message” \(msg)” )
}

// 将 RFID 设定值初始化
gsble.rfidSetupDefault()
```

```
//UHF GEN2 EPC 区资料上锁带 L, 解锁带 U
    gsble.EPCPWD_Action(action： "L", password： "12345678")
//UHF GEN2 标记编号 区资料上锁带 L, 解锁带 U
    gsble.TIDPWD_Action(action： "L", password： "12345678")
//UHF GEN2 使用者区 资料上锁带 L, 解锁带 U
    gsble.USERPWD_Action(action： "L", password： "12345678")
//UHF GEN2 存取密码 上锁带 L, 解锁带 U
    gsble.accessPWD_Action(action： "L", password： "12345678")
//UHF GEN2 删除密码 上锁带 L, 解锁带 U
    gsble.killPWD_Action(action： "L", password： "12345678")

//UHF GEN2 设定存取密码
    gsble.accessPWD_Action(action： "S", password： "12345678")
//UHF GEN2 设定删除密码
    gsble.killPWD_Action(action： "S", password： "12345678")
//UHF GEN2 写入资料
    gsble.clearBuffer()
    gsble.writeUHF(dataFormat： "H", startBlockNo： 2, byteSize： 12,
Gen2MemoryBank： "E", dataString： "414142424343444445454646") { msg in
        self.textView.text = msg   // 代表打印机未连机
    }
    gsble.printLabel(set： 1, copy： 1)
// UHF GEN2 读取资料
    gsble.readUHF(dataFormat： "H", startBlockNo： 0, byteSize： 12,
Gen2MemoryBank： "E") { msg in
```

```
                if msg == "Connection doesn't exist" {} // 代表打印机未连机
                if msg == “time_out” {} // 逾时
                else { }// 回传读取到的资料


        // UHF GEN2 Q 指令读取资料
            gsble.query_UHF(dataFormat： "H", PCReturnStatus： 1, CRCReturnStatus： 1)
{ msg in
                if msg == Connection doesn't exist."  {
                    self.textView.text = msg // 代表未与打印机联机
                    return
            } else if msg == “time_out” {
                    self.textView.text = “time out” // 逾时
                    return
            } else {
                    Self.textView.text = msg // 其他回传，代表读取到的资料
            }
        }


        // UHF GJB 设定密码
            // 带入写入密码，设定新的读取密码
            gsble.set_GJB_Pwd_Action(passwordArea： "R", newPassword： "87654321",
        writePassword： "12345678")
            // 带入写入密码，设定新的写入密码
            gsble.set_GJB_Pwd_Action(passwordArea： "W", newPassword： "87654321",
        writePassword： "12345678")
            // 带入写入密码，设定新的删除密码
            gsble.set_GJB_Pwd_Action(passwordArea： "K", newPassword： "87654321",
        writePassword： "12345678")
            // 带入写入密码，设定新的状态密码
```

```
        gsble.set_GJB_Pwd_Action(passwordArea： "S", newPassword： "87654321",
    writePassword： "12345678")
    // UHF GJB 带入状态密码，设定不同资料区块的写入读取状态
        gsble.statusGJB_UHF(GJBMemoryBank： "E", action： "C", statusPassword：
    "11112222")
    // UHF GJB 带入写入密码，将资料写入指定资料区块
        gsble.clearBuffer()
        gsble.writeGJB_UHF(dataFormat： "H", startBlockNo： 1, byteSize： 12,
    Gen2MemoryBank： "E", dataString： "40404141424243434444454 5",
    writePassword： "12345678") { msg in
            if msg == "Connection doesn't exist" {} // 代表打印机未连机
        }
        gsble.printLabel(set： 1, copy： 1)
    // UHF GJB 带入读取密码，读取资料
        gsble.readGJB_UHF(dataFormat： "H", startBlockNo： 0, byteSize： 12,
    Gen2MemoryBank： "E", readPassword： "33334444") { msg in
            if msg == "Connection doesn't exist" {} // 代表打印机未连机
            } else if msg == “time_out” {} // 逾时                    }
            else { } // 回传读取到的资料
    // UHF GJB 带入删除密码，删除标签
        gsble.killGJB_Tag_UHF(killPassword： "11224455")

// 开启实时命令
    gsble.setRealTimeStatus("1")
    isRealTimeOn = true   // 实时命令设定为 true
    timer = Timer.scheduledTimer(withTimeInterval：0.5, repeats： true, block： { _ in // 开
```

```
始计时，每 0.5 秒读取一次
            if let statusNumber = self.gsble.getStatusNumber() {
                var statusStr = ""
                switch statusNumber {
                case 0：    statusStr = "Normal"
                // 以下略……。取得状态代码，对照表格取得目前状态。
                self.realTimeShowLabel.text = statusStr // 显示在画面UILabel
// 关闭实时命令
    gsble.setRealTimeStatus("0")
    isRealTimeOn = false    // 实时命令设定为 false
    timer.invalidate() // 关闭计时器

// 读取错误代码清单
    let errorCode = gsble.getRFIDErrorCode()
```

1.汇入 GTSPL_SDK：

```
import IOS_SWIFT_WIFI_SDK
```

2.范例程序：

```
//初始化 SDK
    let gswifi = GTSPL_WIFI()
//联机 WIFI 装置
    let ip = IPText.text
    let port：Int? = Int(PortText.text!)
    gswifi.openport(IP：ip!, Port：port!){(msg) in
        print(msg)
    }
//WIFI 装置断线
    gswifi.closeport()
//设定打印机
    gswifi.setup(width： 105, height： 80, speed： 4, density： 6, sensor： 0,
sensordistance： 5, sensoroffset：5)
//打印 barcode、text、bitmap、区段文字
    //先清除资料、再打印指定项目
    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")
//区段文字
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)

//bitmap 档案 路径
let url = Bundle.main.url(forResource： "CIRCLE", withExtension："bmp")!
//传送 bitmap 档案到打印机中
gswifi.downloadBMP(filepath： url, filename： "CIRCLE.bmp")
//将 bitmap 图片排入打印顺序指令
gswifi.sendcommand("PUTBMP 30,30,\"CIRCLE.bmp\"\r\n")
//打印
gswifi.printLabel(set： 1, copy： 1)

//打印二维码
//先清除资料
gswifi.clearBuffer()
//产生二维码
gswifi.printQRCode(x： 50, y： 50, eccLevel： "H", cellWidth： 4, mode： "A",
rotation： 0, content： "QRcode987654321")
gswifi.printLabel(set： 1, copy： 1)
//简中打印
String stString="默认简体中文测试";
```

```
        gswifi.clearBuffer()
        gswifi.printFont(x： 200, y： 200, fontname： "TSS24.BF2", rotation： 0, xScale：
1, yScale： 1, content： cn)
        gswifi.printLabel(set： 1, copy： 1)
//繁中打印
        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)
        gsble.printLabel(set： 1, copy： 1)

//取得 sdk 版本
        let version = gswifi.getSdkVersion()
//取得打印机状态
        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
    }

// 出纸方向、镜像列印
```

```
gsble.setDirectionAndMirror(direction： 0, mirror： 0)

// 图像垂直位移距离 例：偏移 36 dot
gsble.setShift(shiftY： 36)

// 反白列印（输入区域）
gsble.printReverse(x_start： 0, y_start： 0, x_width： 100, y_height： 100)

// 出纸偏移 例：9.9mm
gsble.setOffset(offset： 9.9)

// 设定裁切模式与张数 例：正切、三张
gsble.setCutMode(mode： 1, piece： 3)

// 设定列印后动作 例：撕纸
gsble.setAfterPrintAction(mode： 1)

// 一般设定值初始化
gsble.genericDefault()

// 感应器设定值初始化
gsble.sensorDefault()

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

```
}

// 列印图片，转单色，例：模式 1, 自订大小为 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 设定，例：读写位置 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 设定，例：读写位置 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 校准动作
gsble.rfid_calibration() { msg in
    print("Error message" \(msg)" )
}
```

```
// 将 RFID 设定值初始化
gsble.rfidSetupDefault()

        //UHF GEN2 EPC 区资料上锁带 L, 解锁带 U
            gswifi.EPCPWD_Action(action： "L", password： "12345678")
        //UHF GEN2 标记编号 区资料上锁带 L, 解锁带 U
            gswifi.TIDPWD_Action(action： "L", password： "12345678")
        //UHF GEN2 使用者区 资料上锁带 L, 解锁带 U
            gswifi.USERPWD_Action(action： "L", password： "12345678")
        //UHF GEN2 存取密码 上锁带 L, 解锁带 U
            gswifi.accessPWD_Action(action： "L", password： "12345678")
        //UHF GEN2 删除密码 上锁带 L, 解锁带 U
            gswifi.killPWD_Action(action： "L", password： "12345678")

        //UHF GEN2 设定存取密码
            gswifi.accessPWD_Action(action： "S", password： "12345678")
        //UHF GEN2 设定删除密码
            gswifi.killPWD_Action(action： "S", password： "12345678")
        //UHF GEN2 写入资料
            gswifi.clearBuffer()
            gswifi.writeUHF(dataFormat： "H", startBlockNo： 2, byteSize： 12,
        Gen2MemoryBank： "E", dataString： "414142424343444445454646") { msg in
                self.textView.text = msg   // 代表打印机未连机
            }
            gswifi.printLabel(set： 1, copy： 1)
        // UHF GEN2 读取资料
```

```
        gswifi.readUHF(dataFormat： "H", startBlockNo： 0, byteSize： 12,
    Gen2MemoryBank： "E")
        { msg in
            if msg == "Connection doesn't exist" {} // 代表打印机未连机
            if msg == “time_out” {} // 逾时
            else { }// 回传读取到的资料
    // UHF GEN2 Q 指令读取资料
        gsble.query_UHF(dataFormat： "H", PCReturnStatus： 1, CRCReturnStatus： 1)
{ msg in
            if msg == Connection doesn't exist."  {
                self.textView.text = msg // 代表未与打印机联机
                return
            } else if msg == “time_out” {
                self.textView.text = “time out” // 逾时
                return
            } else {
                Self.textView.text = msg // 其他回传，代表读取到的资料
            }
        }

    // UHF GJB 设定密码
        // 带入写入密码，设定新的读取密码
        gswifi.set_GJB_Pwd_Action(passwordArea： "R", newPassword： "87654321",
    writePassword： "12345678")
        // 带入写入密码，设定新的写入密码
        gswifi.set_GJB_Pwd_Action(passwordArea： "W", newPassword： "87654321",
    writePassword： "12345678")
        // 带入写入密码，设定新的删除密码
        gswifi.set_GJB_Pwd_Action(passwordArea： "K", newPassword： "87654321",
    writePassword： "12345678")
```

```
        // 带入写入密码，设定新的状态密码
        gsble.set_GJB_Pwd_Action(passwordArea： "S", newPassword： "87654321",
writePassword： "12345678")
// UHF GJB 带入状态密码，设定不同资料区块的写入读取状态
        gswifi.statusGJB_UHF(GJBMemoryBank： "E", action： "C", statusPassword：
"11112222")
// UHF GJB 带入写入密码，将资料写入指定资料区块
        gswifi.clearBuffer()
        gswifi.writeGJB_UHF(dataFormat： "H", startBlockNo： 1, byteSize： 12,
Gen2MemoryBank： "E", dataString： "404041414242434344444545",
writePassword： "12345678") { msg in
            if msg // 代表打印机未连机
        }
        gsble.printLabel(set： 1, copy： 1)
// UHF GJB 带入读取密码，读取资料
        gswifi.readGJB_UHF(dataFormat： "H", startBlockNo： 0, byteSize： 12,
Gen2MemoryBank： "E", readPassword： "33334444") { msg in
            if msg == "Connection doesn't exist" {} // 代表打印机未连机
            else if msg == “time_out” {}   // 逾时
            else { }// 回传读取到的资料
// UHF GJB 带入删除密码，删除标签
        gswifi.killGJB_Tag_UHF(killPassword： "11224455")
// 开启实时命令
        gswifi.setRealTimeStatus("1")
        isRealTimeOn = true   // 实时命令设定为 true
        timer = Timer.scheduledTimer(withTimeInterval：0.5, repeats： true, block： { _ in
```

```
// 开始计时，每 0.5 秒读取一次
            if let statusNumber = self.gsble.getStatusNumber() {
                var statusStr = ""
                switch statusNumber {
                case 0：    statusStr = "Normal"
                // 以下略……。取得状态代码，对照表格取得目前状态。
                self.realTimeShowLabel.text = statusStr // 显示在画面UILabel
// 关闭实时命令
    gswifi.setRealTimeStatus("0")
    isRealTimeOn = false    // 实时命令设定为 false
    timer.invalidate() // 关闭计时器
// 读取错误代码清单
    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		
PLESSE	PLESSEY.		V	V	V		

Y							
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
LOGMARS	A special use of Code 39. *Since V6.88EZ.		V	V	V		