



iOS Software Development Kit

DK-AirCash 向け StarIO 使用方法

本 SDK には iOS デバイスのための Xcode Objective-C プロジェクトを含んでいます。

必要なツール:

- Xcode 7.0 以降
- StarIO iOS SDK

この SDK は、有線 LAN/無線 LAN インターフェースの場合、DK-AirCash ファームウェアバージョン 2.0 以降に対応しています。

Bluetooth インターフェースの場合、ファームウェアバージョン 1.0 から対応しています。

3.14.0 以降の StarIO.framework をご使用の際には、合わせて以下の framework を追加いただく必要があります。

- External Accessory framework
- Core Bluetooth framework

※既に 3.13.1 以前をご使用で StarIO.framework のバージョンアップを行う場合、新たに Core Bluetooth framework のプロジェクトへの追加が必要です。

詳しくは[こちら](#)をご参照ください。

StarIO SDK 対応 OS : iOS 7.0 以降**StarIO SDK 対応リスト**

デバイス	CPU
iPad 2	Armv7
iPad (第 3 世代)	Armv7
iPad (第 4 世代)	Armv7s
iPad Air	Arm64
iPad Air 2	Arm64
iPad mini	Armv7
iPad mini 2	Arm64
iPad mini 3	Arm64
iPad mini 4	Arm64
iPad Pro	Arm64
iPhone 4s	Armv7
iPhone 5	Armv7s
iPhone 5s	Arm64
iPhone 5c	Armv7s
iPhone 6	Arm64
iPhone 6 Plus	Arm64
iPhone 6s	Arm64
iPhone 6s Plus	Arm64
iPod touch (第 5 世代)	Armv7
iPod touch (第 6 世代)	Arm64

注) iPad、iPhone、iPod、iPod touch は、米国および他の国々で登録された Apple Inc.の商標です。iPad Air、iPad mini、は、Apple Inc. の商標です。“iPhone”の商標は、アイホン株式会社のライセンスにもとづき使用されています。iOS は、米国およびその他の国における Cisco 社の商標または登録商標であり、ライセンスにもとづき使用されています。

目 次

- ❖ [本書に関して](#)
- ❖ [Star POS デバイスを iOS デバイスに接続するには](#)
- ❖ [はじめに](#)
- ❖ [Star POS デバイスで SDK を使用する](#)
- ❖ [iOS SDK 概要](#)
- ❖ [StarIO framework](#)
- ❖ [StarIO メソッド概要](#)
 - [SMPort クラス](#)
 - [SMBluetoothManager クラス](#)
- ❖ [機能](#)
 - [Help](#)
 - [Get DK-AirCash Firmware Information](#)
 - [Get DK-AirCash Dip Switch Information](#)
 - [Get Printer Status](#)
 - [Get DK-AirCash Status](#)
 - [Sample Receipt + Open Cash Drawer via DK-AirCash](#)
 - [Open Cash Drawer via DK-AirCash](#)
 - [Bluetooth Pairing + Connect](#)
 - [Bluetooth Disconnect](#)
 - [DK-AirCash Bluetooth Setting](#)
- ❖ [StarIO を使用するアプリケーション開発のために](#)
- ❖ [追加リソース](#)
 - [スター精密グローバルサポートサイト](#)
 - [ASCII コード表](#)
- ❖ [SDK パッケージ 改訂履歴](#)

本書に関して

本マニュアルは、StarIO と Star DK-AirCash が通信を行う、iOS アプリケーションの作成方法を解説しています。

また、このマニュアルは、アプリケーション・システム開発者を対象に作成しており、利用者は Objective-C 言語の基礎を理解していることを前提としています。

この SDK は iOS 用に作成されています。

[スター精密グローバルサポートサイト](#)の Developers セクションには、その他のオペレーティングシステムとプログラミング言語に利用可能な SDK が用意されています。最新の SDK、テクニカルドキュメント、FAQ 及び、その他の追加情報については、Developers セクションをご確認ください。

表示マークの説明:

警告



潜在的な問題について説明します。

禁止



禁止事項について説明します。

メモ



重要な情報とヒントを提供します。

注意事項:

- 本マニュアルの内容は、予告無く変更する場合があります。
- スター精密株式会社は、正確な情報を提供するためにあらゆる措置を取っていますが、誤りや不作為について責任を負うものではありません。
- スター精密株式会社は、このマニュアルに記載されている情報の使用に起因するいかなる損害に対しても責任を負うものではありません。
- 本マニュアルの一部、あるいは全部を無断で複製・複製・転載することは、固くお断りします。

Star デバイスを iOS デバイスに接続するには

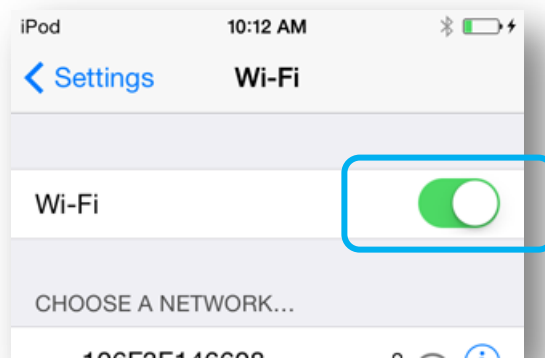
有線 LAN/無線 LAN インターフェイス

Star DK-AirCash は、工場出荷時の初期設定は DHCP が有効になっています。使用するネットワークが DHCP をサポートする場合、DK-AirCash が自動的に IP アドレスを取得できるように、必要なネットワーク構成を構築してください。

また、本製品の #9100 Multi Session を無効に設定して使用してください。

#9100 Multi Session の確認・変更、固定 IP アドレスの設定方法については、[こちら](#)の[リンク](#)の「対応 OS・環境一覧 > ユーティリティー一覧」より「イーサネットプリンター利用手引き」をご参照ください。固定 IP アドレスの設定に必要な製品固有の情報は、製品に付属のアドレス印字ラベルに記載されています。

1. Star POS デバイ스에 IP アドレスを割り当て、ネットワークに接続します。
2. 「設定」をタップします。
3. 「Wi-Fi」を ON に設定します。



4. Star POS デバイスと同じネットワークに接続します。

Bluetooth インターフェイス

Star POS デバイスは、工場出荷時の初期設定では各機種ごとに"Star Micronics","DK-AirCash"等、共通の Bluetooth デバイス名が設定されています。同じ Bluetooth デバイス名の機種を複数台配置して運用される場合、Bluetooth デバイス名の変更を行うと Star POS デバイスの判別が付けやすく便利です。

Bluetooth デバイス名の変更等、Star POS デバイスの有線 LAN/無線 LAN/Bluetooth 設定値は、スター精密の提供する Star Setting Utility を使用して変更することができます。Star Setting Utility は App Store よりダウンロードしてください。

◆ペアリング

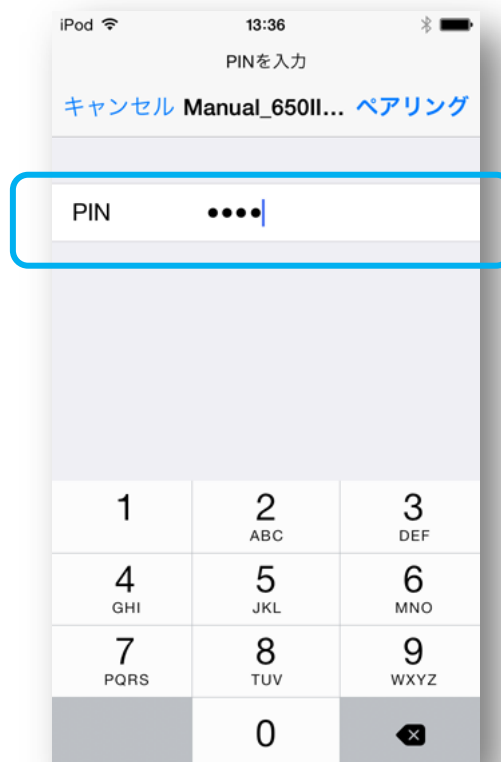
1. 設定を行う Star POS デバイスのインターフェイス選択が Bluetooth になっていることを確認して、デバイスの電源を投入します。
Star POS デバイスのセキュリティ方式が SSP の場合は、PAIR ボタンを 5 秒以上押してペアリング可能にします。
2. iOS の[設定]より、[Bluetooth]をタップします。



3. [Bluetooth]を "オン"に設定すると、iOS デバイスとペアリングが可能な Bluetooth デバイスの検索を行い、表示します。ペアリングを行う Star POS プリンター等をタップします。



4. PIN を入力します。（Star POS デバイスの Bluetooth セキュリティが PIN Code の場合）



5. 以下の表示が確認できればペアリング完了です。



◆Bluetooth 名称の変更

App Store から Star Bluetooth Utility をダウンロードして、iOS ポート名を変更することができます。必要に応じてご使用ください。

iOS ポート名はペアリング実行後、以下の手順で確認できます。

「設定」 - 「一般」 - 「情報」 Bluetooth アドレスの下に表示

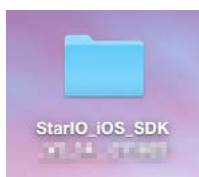
はじめに

iOS のプロジェクトをビルドするには、Xcode が必要です。これらのツールは、[Apple Developer サイト](#)や Mac App Store から入手可能です。実際に iOS デバイス上で動作するアプリケーションを生成するには、Apple の Developer Program への登録が必要です。（Developer Program は年一度の更新手続きを必要とします） Developer Program への登録を行わずに iTunes からこれらのツールを入手することは可能ですが、その場合、アプリケーションは iOS シミュレーター上で動かすことができるだけであって、実際の iOS デバイスにはインストールされません。

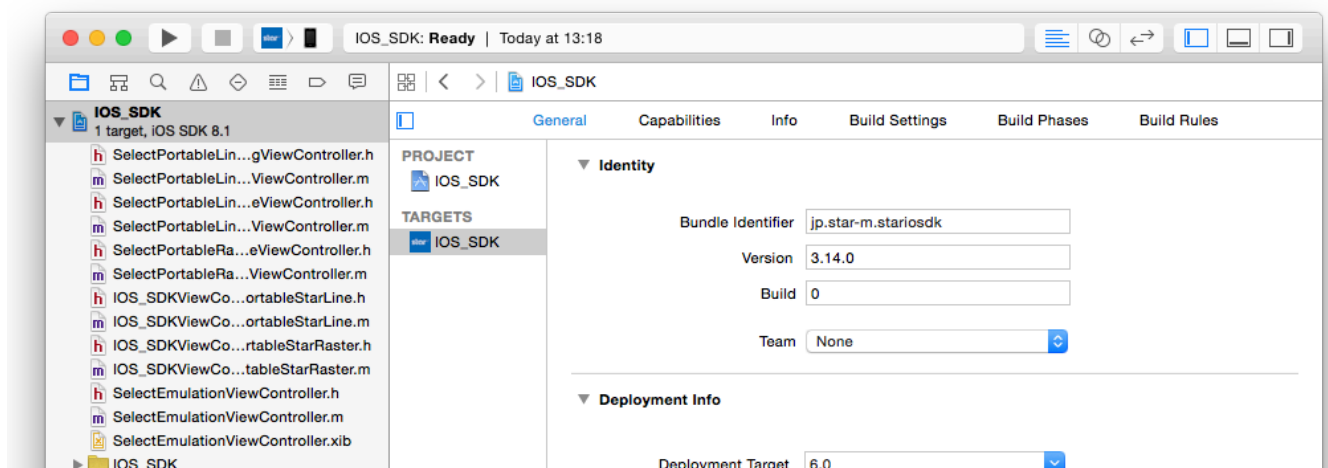
事前に、開発を行う Mac に Xcode のインストールを行ってください。万一、サポートまたは追加情報が必要な場合は、Apple Developer サイトの [Resources](#) セクションを参照してください。

Xcode で Star iOS SDK プロジェクトを開く方法：

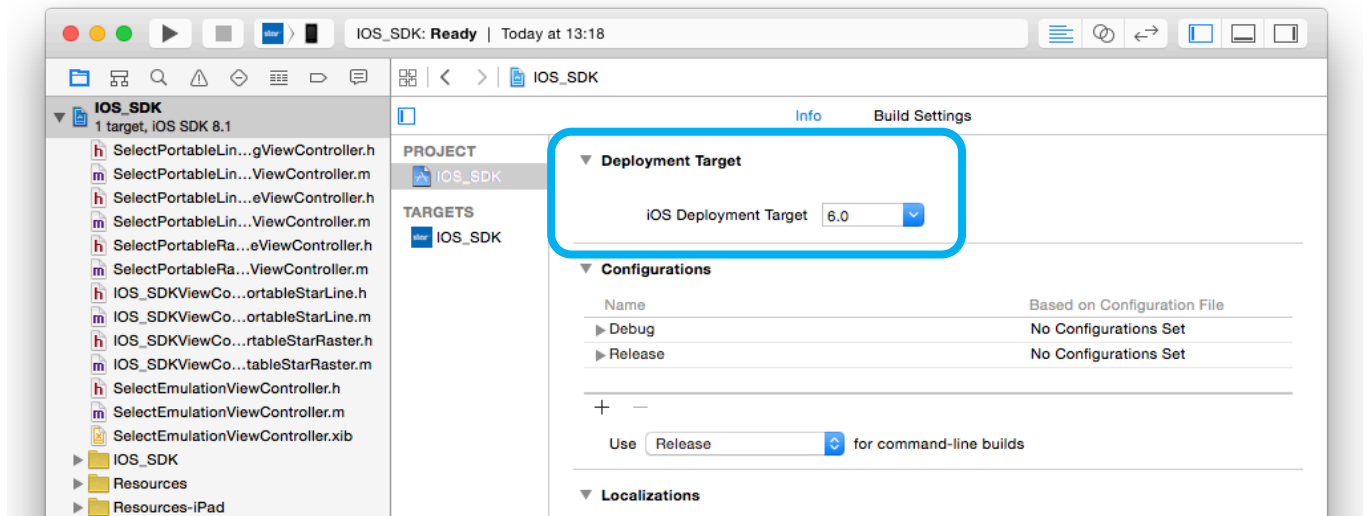
- 1) Star iOS SDK フォルダを解凍し、開きます。



- 2) IOS_SDK.xcodeproj を開きます。

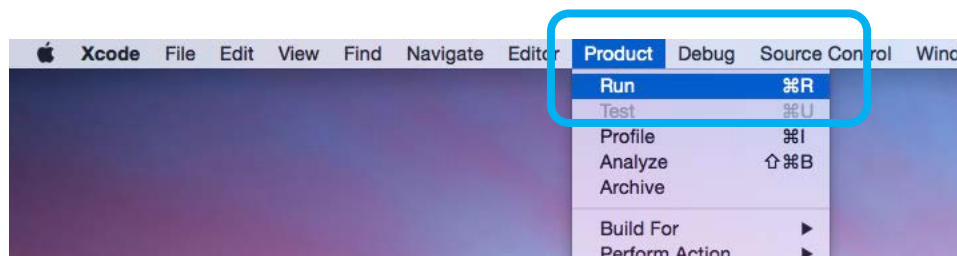


3) iOS Deployment Target の欄にて 6.0 以降を選択してください。



プロジェクトを実行する :

- 1) ショートカットの“⌘R”を使用するか、上部メニューバーの“Product - Run”をクリックして実行します。



Star デバイスで SDK を使用する

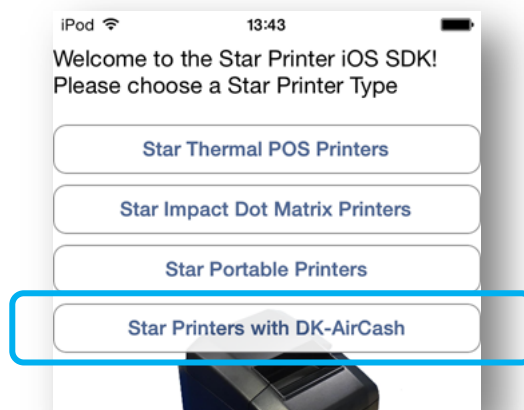
ポート名の設定:

StarIO は、デバイスと通信するために特定のポート名を使用します。ポート名は、以下の通り正しく設定しないとデバイスと通信できません。

Interface	Port Name
有線 LAN/無線 LAN I/F (TCP/IP)	TCP:“IP アドレス”
Bluetooth	BT:“iOS ポート名” ※

DK-AirCash の設定

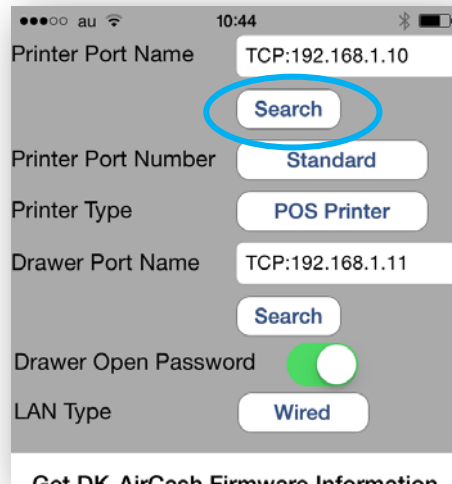
- 1) “Star Printers with DK-AirCash”をタップします。



有線 LAN/無線 LAN/Bluetooth デバイスの設定

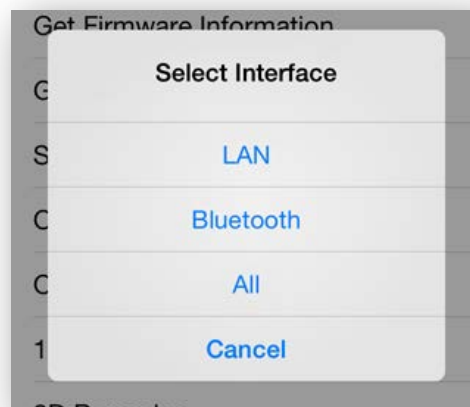
・接続するプリンターを検索して設定する

1) "Search"をタップします。



2) 接続するプリンターのインターフェイスをタップします。

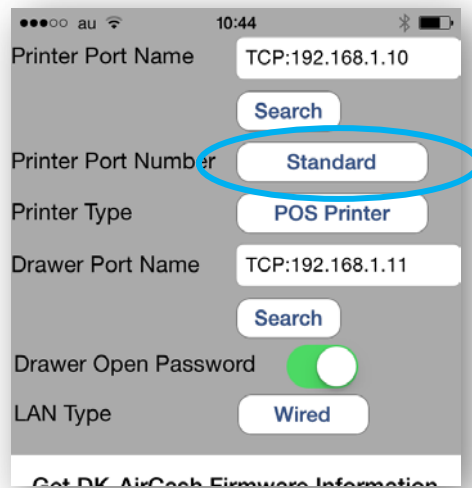
有線 LAN/無線 LAN の場合、接続可能な有線 LAN/無線 LAN プリンターを検索して表示します。Bluetooth の場合、ペアリングされた接続可能な Bluetooth プリンターのポート名を表示します。



3) 検索されたプリンター一覧から、接続するプリンターの名前をタップしてください。

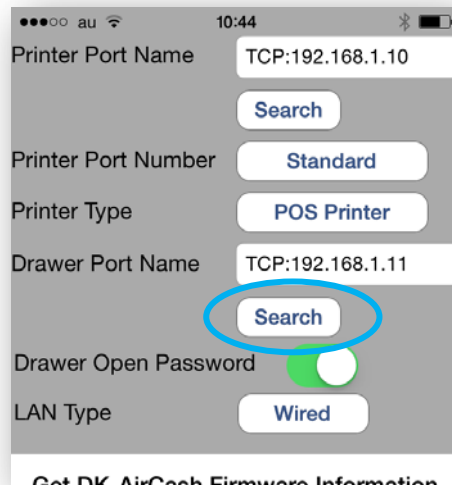


- 4) 有線 LAN/無線 LAN の場合は、必要に応じて、"Standard"ドロップダウンをクリックして、TCP ポートを設定します。このサンプルアプリケーションでは、9100 から 9109 までの任意のポートを選択することができます。
- Apple AirMac Express などのルータを使用する際にポートを設定する必要がある場合があります。



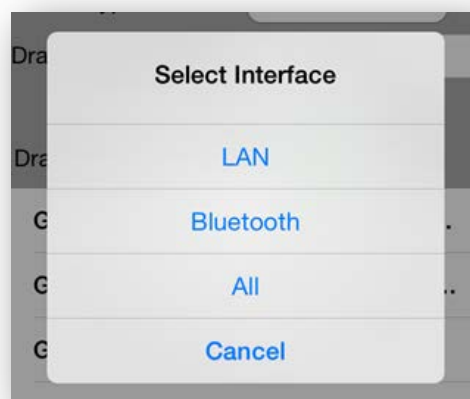
・ 接続する DK-AirCash を検索して設定する

1) "Search"をタップします。

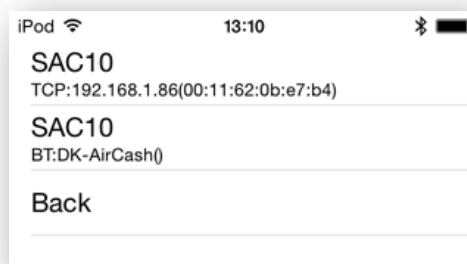


2) 接続する DK-AirCash のインターフェイスをタップします。

有線 LAN/無線 LAN の場合、接続可能な有線 LAN/無線 LAN 接続の DK-AirCash を検索して表示します。Bluetooth の場合、ペアリング済みで接続可能な Bluetooth 接続の DK-AirCash ポート名を表示します。



3) 検索された DK-AirCash 一覧から、接続するポート名をタップしてください。

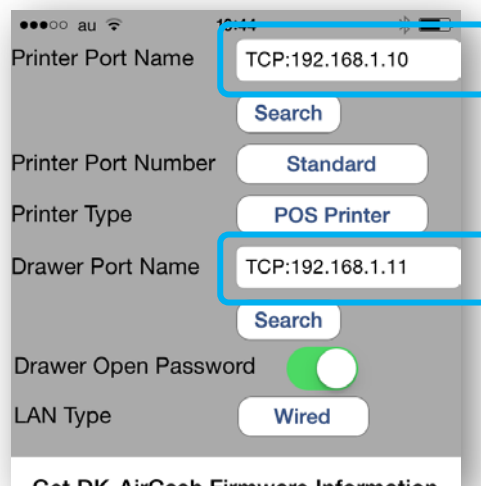


・接続するプリンター/DK-AirCash を手動で設定する

1) プリンターまたはドロワーの「PortName」に以下のように手動で入力します。

●TCP:<IP アドレス>

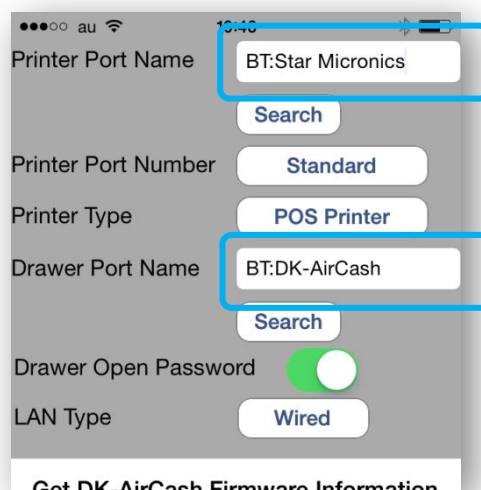
"TCP:"の後にプリンターの IP アドレスを追加します。（カッコは不要です）



The screenshot shows the 'Printer Port Name' field with the value 'TCP:192.168.1.10' entered. Below it is a 'Search' button. The 'Printer Port Number' is set to 'Standard'. The 'Printer Type' is set to 'POS Printer'. The 'Drawer Port Name' field has the value 'TCP:192.168.1.11' entered, with a 'Search' button below it. The 'Drawer Open Password' is toggled on. The 'LAN Type' is set to 'Wired'. At the bottom, there is a link to 'Get DK-AirCash Firmware Information'.

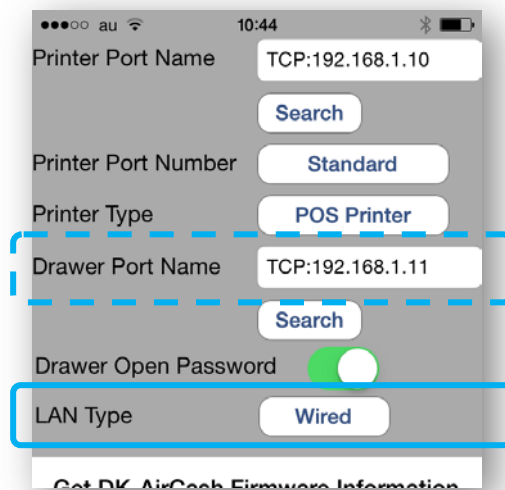
●BT: <Bluetooth デバイスのポート名>

"BT:"の後に Bluetooth デバイスのポート名を入力します。（カッコは不要です）



The screenshot shows the 'Printer Port Name' field with the value 'BT:Star Micronics' entered. Below it is a 'Search' button. The 'Printer Port Number' is set to 'Standard'. The 'Printer Type' is set to 'POS Printer'. The 'Drawer Port Name' field has the value 'BT:DK-AirCash' entered, with a 'Search' button below it. The 'Drawer Open Password' is toggled on. The 'LAN Type' is set to 'Wired'. At the bottom, there is a link to 'Get DK-AirCash Firmware Information'.

・有線 LAN/無線 LAN デバイス – 接続方法選択



Drawer Port Name に設定した DK-AirCash が有線 LAN/無線 LAN デバイスの場合、接続方法に合わせて、LAN Type を設定します。

有線 LAN で接続する場合は“Wired”を選択します。getPort メソッドの portSetting パラメータには""（空文字列）が設定されます。

無線 LAN で接続する場合は“Wireless”を選択します。getPort メソッドの portSetting パラメータには";wl"が設定されます。

iOS SDK 概要

SDK の主要コンポーネントについて簡単に説明します。

全ての機能が IOS_SDK project と IOS_SDK target に位置しています。

IOS_SDKViewControllerDKAirCash.m ファイルからプログラムを実行してください。このソース・コードが DK-AirCash の起点となります。

他のソース・ファイルをクリックすることにより、特定の機能がどのように働くか確かめてください。例えば、“code128.m”は GUI の中の 1D barcode Code128 に相当します。

全ての機能が DK-AirCash に利用可能ではありませんので、ご注意ください。各 SDK マニュアルの最初のページには、どの機能がサポートされているか記載されています。



StarIO Framework

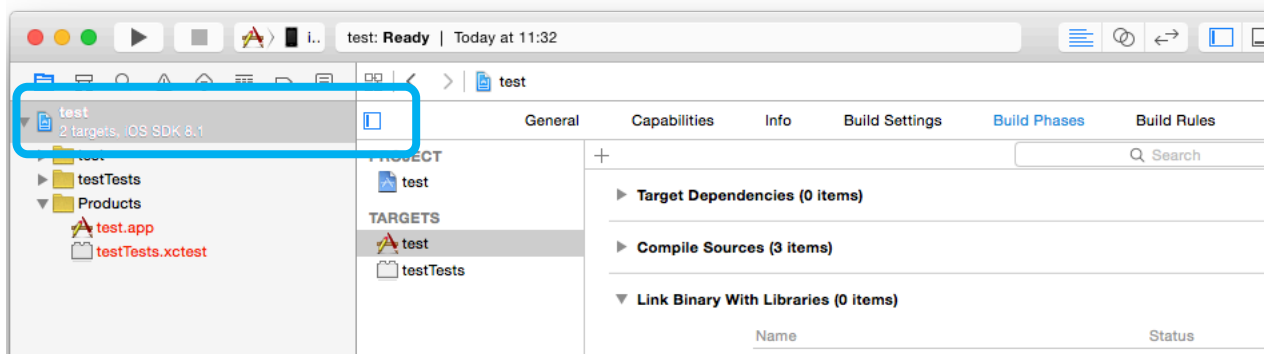
Star iOS SDK プロジェクトには、既に StarIO framework は含まれています。(この SDK をテストする場合、そのままご使用できます)

ただし、新規のアプリケーションを作成する場合、StarIO メソッドを使用するためにプロジェクトに必要な framework を追加する必要があります。

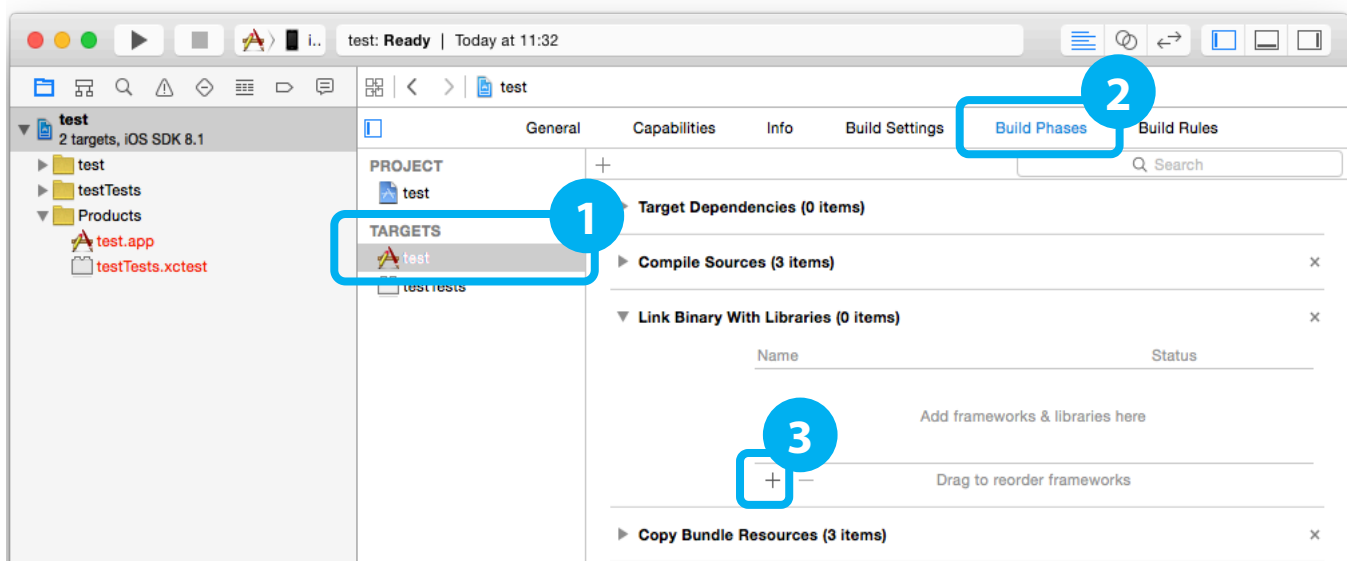
●新規のアプリケーションを作成するには

1. プロジェクトに StarIO.framework を追加する

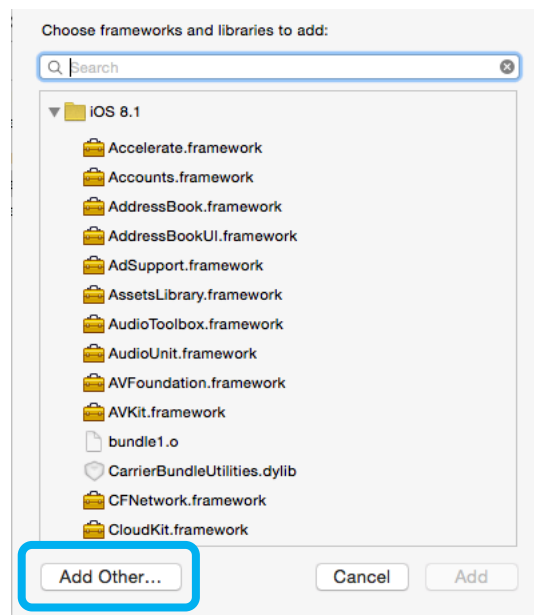
1. 新規に作成したプロジェクトをクリックします。



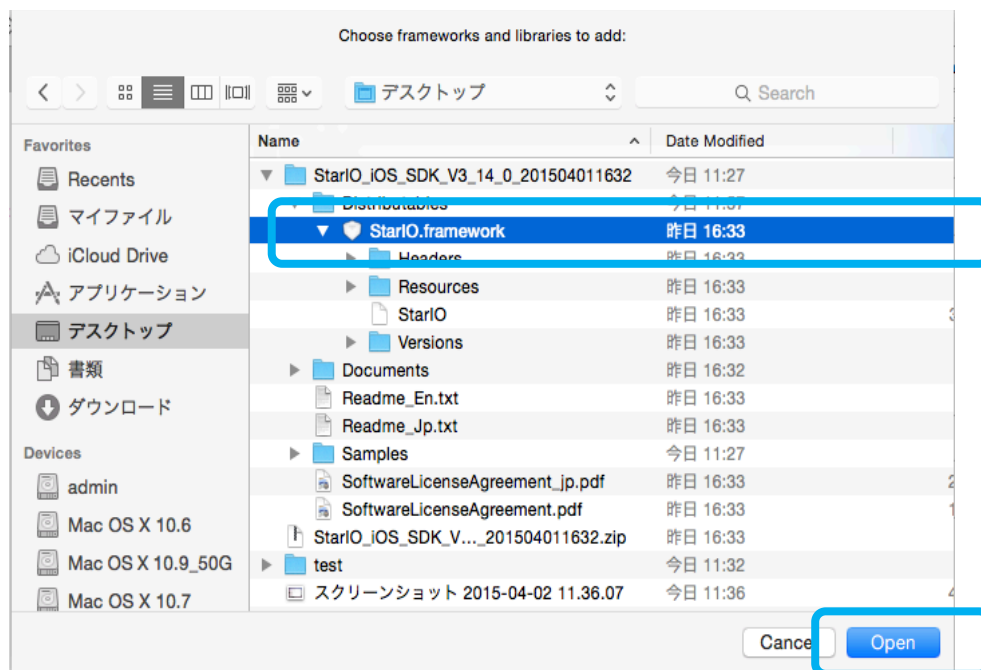
2. ターゲット → “Build Phases” → Link Binary With Libraries の“+” をクリックします。



3. “Add Other…”ボタンをクリックします。



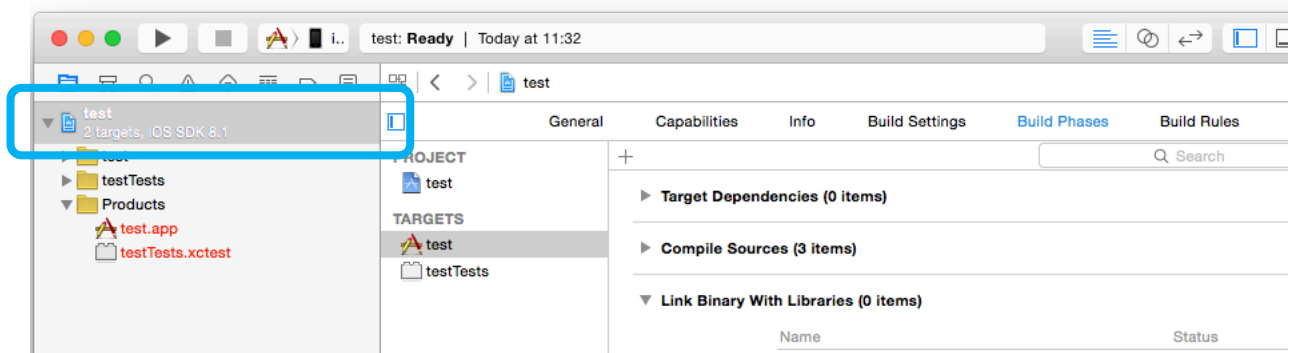
4. Star iOS SDK を解凍した場所の StarIO.framework フォルダを参照し、“Open”をクリックします。



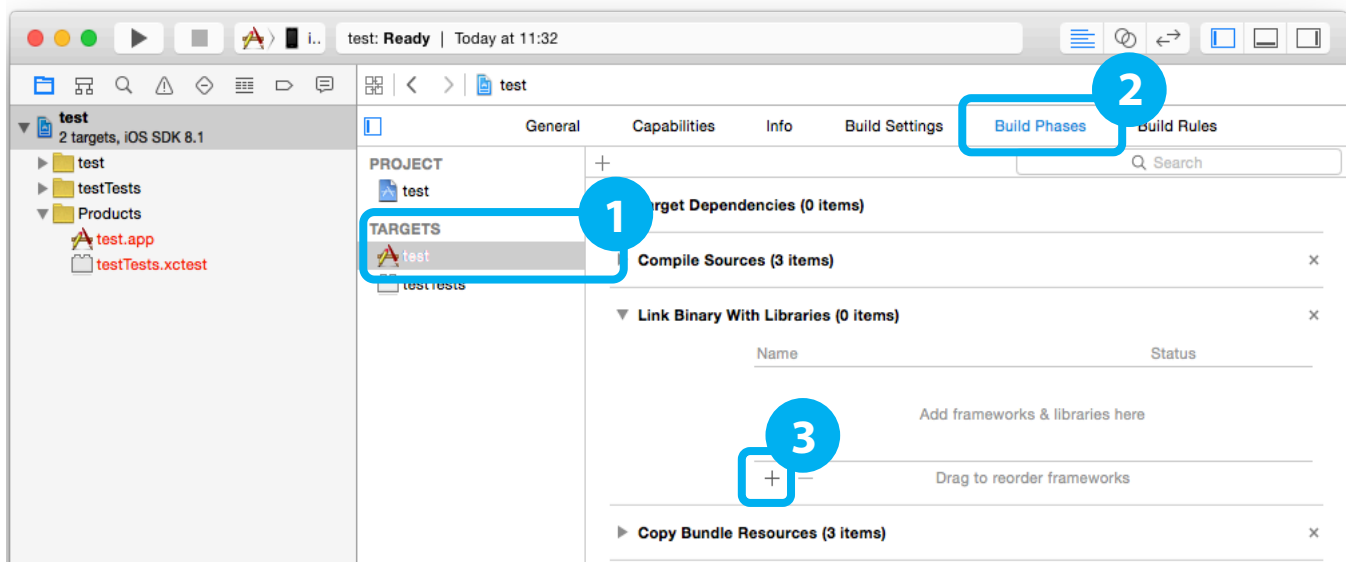
5. StarIO framework はプロジェクトに追加されます。

2. その他の framework を追加する

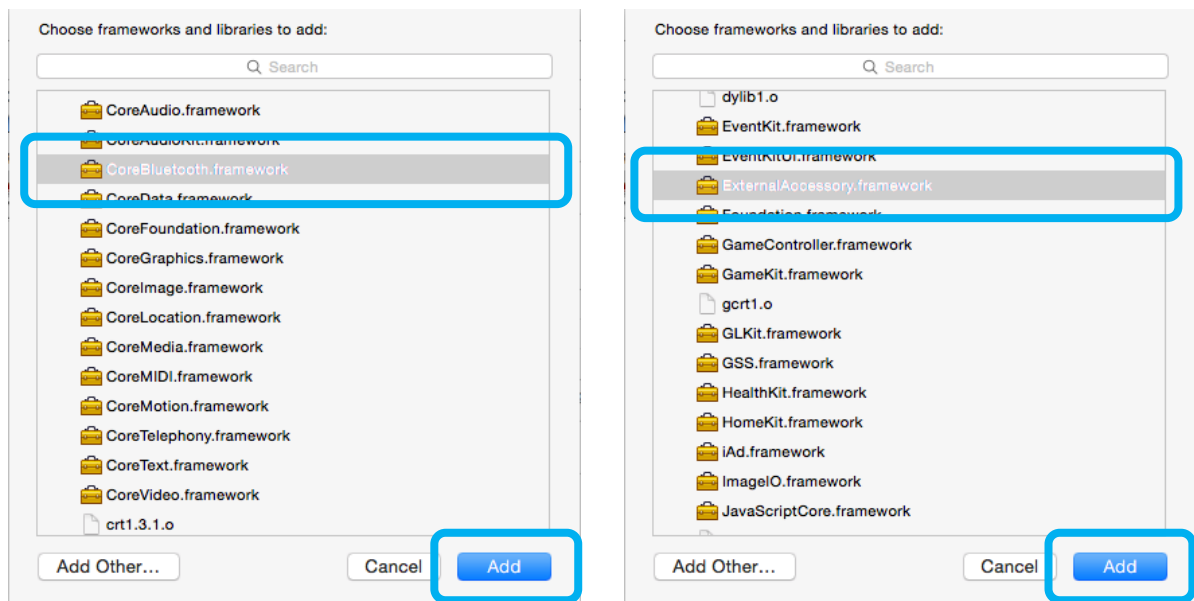
1. 新規に作成したプロジェクトをクリックします。



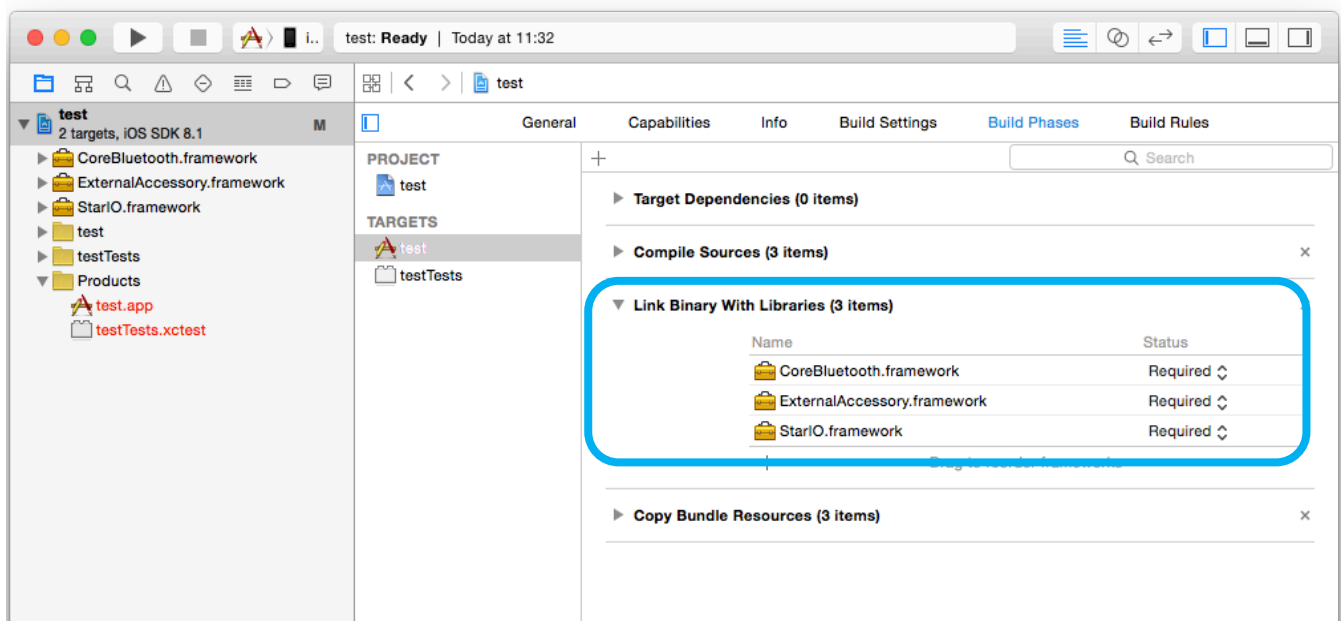
2. ターゲット → “Build Phases” → Link Binary With Libraries の“+” をクリックします。



3. External Accessory framework, Core Bluetooth framework をそれぞれ追加します。
framework を選択し、“Add”をクリックします。



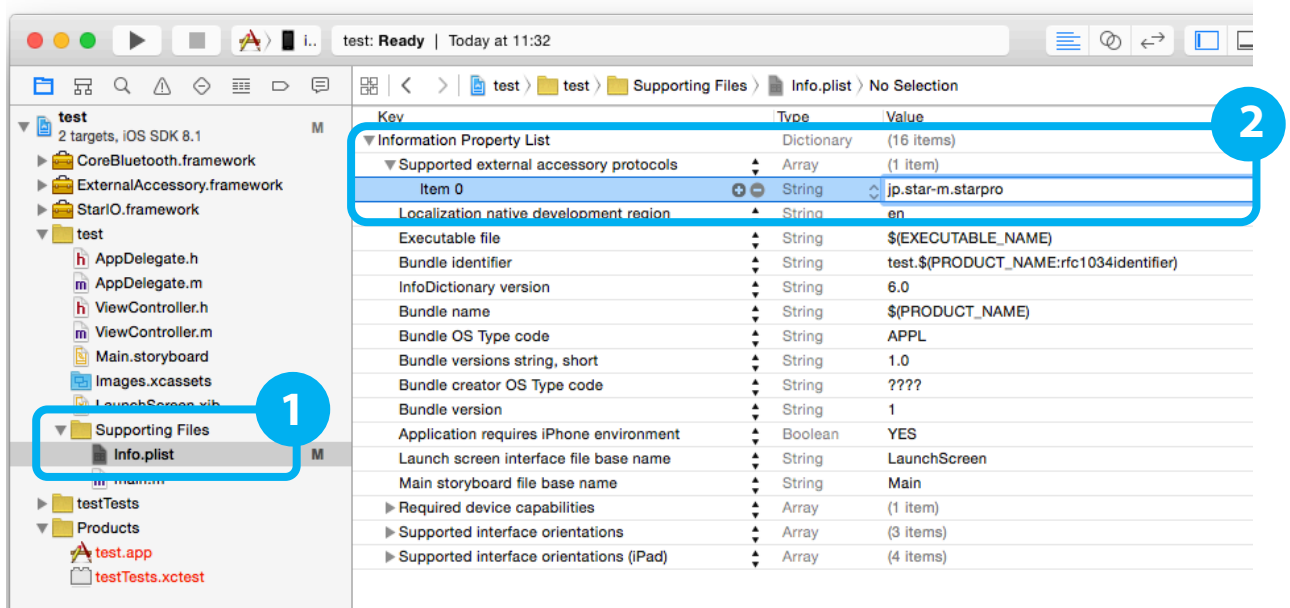
4. 必要な framework が追加されているか確認してください。



3.Information Property List へ項目を設定する（ Bluetooth を使用する場合）

※Bluetooth プリンターを使用しない場合は、この設定を行わないでください。

1. Information Property List(デフォルトでは“Info.plist”)をクリックします。
2. 項目一覧の[Key]に“Supported external accessory protocols”を追加して、項目名左側の▽をクリックして表示される“Item 0”の[Value]に “jp.star-m.starpro” を設定します。



3. Information Property List の設定は完了です。

●プロジェクトの StarIO.framework をバージョンアップするには

1. 現在使用している StarIO.framework を削除します。
2. 同じ場所に新しい StarIO.framework をコピーします。
3. Xcode プロジェクトをクリーンします。
Xcode プロジェクトを開き、メニューから[Build]-[Clean]を選択します。
4. Xcode プロジェクトをビルドします。



現在使用している StarIO.framework を削除せずに、参照先を新しい StarIO.framework に変更する場合には、必ず Xcode プロジェクトの “framework search path” の設定を確認してください。

“framework search path” の先頭に古い StarIO.framework のパスが残っていると、引き続き今までの StarIO.framework が使用されてしまいます。

StarIO メソッド概要

SMPort クラス:

●プロパティ

portName	プリンターの通信ポートを取得します。
portSettings	ポート設定を取得します。
timeoutMillis	内部制御と API のタイムアウト時間を取得します。
endCheckedBlockTimeoutMillis	endCheckedBlock メソッドのタイムアウト時間を取得、設定します。

- (NSString *)**portName**

プリンターの通信ポートを取得します。

- (NSString *)**portSettings**

ポート設定を取得します。

- (u_int32_t)**timeoutMillis**

内部制御と API のタイムアウト時間を取得します。（単位：ミリ秒）

@property(assign, readonly) u_int32_t **endCheckedBlockTimeoutMillis**

endCheckedBlock メソッドのタイムアウト時間を取得します。[単位: ミリ秒]

印刷に時間がかかる場合、この値を大きくする事で endCheckedBlock メソッドの印刷完了待ち時間を伸ばす事ができます。

初期値は、getPort メソッドで指定したタイムアウト時間となります。

●メソッド

getPort

```
+ (SMPort *) getPort: (NSString *) portName : (NSString *) portSettings : (u_int32_t) timeoutMillis
```

getPort は、プリンターポートのオープンに使用されます。

引数:

portName - プリンターへの通信ポートを指定

例) @"TCP:192.168.1.2" (Ethernet の場合)

 @"BT:Star Micronics" (Bluetooth の場合)

 @"BT:00:11:62:1b:4d:f4" (Bluetooth で MAC アドレスを指定する場合)

Note: Bluetooth の MAC アドレス指定は iOS6 のみ使用可能で、iOS7 以降では使用できません。

◆ Apple AirMac Express のプリンター共有機能を使用する場合
portName には、AirMac Express の IP アドレスを指定してください。
 例) @"TCP:192.168.1.2"

portSettings - 無線 LAN を使用する場合に@";wl"を指定
 それ以外の場合は空文字列 (@"") を指定

例) @"" (Bluetooth ・ 有線 LAN の場合)

 @"wl" (無線 LAN の場合)

timeoutMillis - 内部制御と API の通信タイムアウト値を指定

Note: timeoutMillis は API が制限された時間内で完了することを保証しますが、正確なタイムアウトの長さを保証するものではありません。

◆ Apple AirMac Express のプリンター共有機能を使用する場合
portSettings には、ポート番号を指定します。
 @"9100"から@"9109"を順に指定し、接続に成功した値を使用してください。
 例) @"9100"

戻り値:

SMPort クラスインスタンス。ポートオープンに失敗した場合、既にオープン済みであった場合は、nil が返されます。



getPort を実行した後は、必ず **releasePort** してから次の **getPort** を行ってください。releasePort をせずに次の **getPort** を行くと、nil が返されます。

//The following would be an actual usage of getPort:

```
SMPort *port = nil;
NSString *portName = @"TCP:192.168.0.5";
NSString *portSettings = @"";
@try
{
    port = [SMPort getPort:portName :portSettings :10000];
}
@catch (PortException)
{
    //There was an error opening the port
}
@finally
{
    [SMPort releasePort:port];
}
```



getPort を使用する場合は、常に **try catch** を使用してください。上記の例のような **try catch** を使用しなければ、通信エラーでポートオープンができなかった場合、プログラムはクラッシュする可能性があります。

searchPrinter

```
+ (NSArray *) searchPrinter;
+ (NSArray *) searchPrinter: (NSString *) target
```

searchPrinter は、有線 LAN/無線 LAN 上の DK-AirCash を検索し、検索結果を NSArray 型で返します。

戻り値の NSArray には、PortInfo クラスのインスタンスが含まれます。

戻り値の PortInfo クラスは、ポート名、DK-AirCash の MAC アドレス、DK-AirCash モデル名を持ち、それぞれ portName, macAddress, modelName プロパティにて取得することができます。

portName は、getPort の引数として使用することができます。

引数 target を指定すると、有線 LAN/無線 LAN もしくは Bluetooth デバイスのいずれかのみを検索することができます。

引数:

target	– @”TCP:”を指定した場合：	DK-AirCash (有線 LAN/無線 LAN)を検索
	@”BT:”を指定した場合：	DK-AirCash(Bluetooth)を検索



本 API はデバイスを確実に検出する事を保証するものではありません。

Bluetooth の MAC アドレス取得は iOS6 のみ使用可能で、iOS7 以降では使用できません。

//The following would be an actual usage of searchPrinter:

```
NSArray *portArray = [[SMPort searchPrinter] retain];
for (int i = 0; i < portArray.count; i++) {
    PortInfo *port = [portArray objectAtIndex:i];
    NSLog(@"Port Name: %@", port.portName);
    NSLog(@"MAC Address : %@", port.macAddress);
    NSLog(@"Model Name: %@", port.modelName);
}
[portArray release];
```

上記の例では、ネットワーク上の DK-AirCash を検索して一覧を取得し、その内容をログに出します。

readPort

```
- (u_int32_t) readPort: (u_int8_t *) readBuffer : (u_int32_t *) offset : (u_int32_t) size;
```

このメソッドは、デバイスからデータを読み込みます。DK-AirCash から Raw byte を読み取る必要のある場合のみ、ご使用ください。



Raw Status の取得にこのメソッドを使用しないでください。
Status の取得は、getParsedStatus::メソッドを使用してください。

引数:

- readbuffer - データが読み込まれるバイト配列のバッファ
- offset - ReadBuffer にデータを書き込み始める場所を指定
- size - 読み込むバイト数の合計

戻り値:

実際に読み込まれたバイト数。データが全て読み取れなかった時でも、この関数は成功します。アプリケーションは、期待されるデータが全て読み取れるまで、この関数を複数回、呼び出す必要があります。または、しきい値に達するまで再試行をするようにします。

例外:

`PortException` - 通信エラーが発生したとき

releasePort

```
+ (void) releasePort: (SMPort *) port;
```

このメソッドは、指定されたポートへの接続をクローズします。

引数:

port - 以前に初期化されたポートを表すポートタイプ



getPort を実行した後は、必ず releasePort してから次の getPort を行ってください。releasePort をせずに次の getPort を行くと、nil が返されます。

writePort

```
-(u_int32_t) writePort: (u_int8_t const *) writeBuffer: (u_int32_t) offset: (u_int32_t) size;
```

このメソッドは、デバイスにデータを書き込みます。DK-AirCash にコマンドの送信を行う場合に使用します。

コマンド送信終了の確認を行うため、このメソッドの前後で beginCheckedBlock / endCheckedBlock を使用してください。

サンプルコードは[こちら](#)をご参照ください。

※安全なプログラミングをするために [try catch](#) を使用してください。

SDK の“PrintTextWithPortName”には、DK-AirCash にデータが送信されたことを確認する方法を示すコードが記述されています。

引数:

- | | |
|-------------|----------------------------------|
| writeBuffer | - 出力データを格納する Byte 配列のバッファ |
| offset | - writeBuffer からデータを読み込み始める場所を指定 |
| size | - 書き込むバイト数の合計 |

戻り値:

実際に書き込まれたバイト数。データが全て書き込めなかった時でも、この関数は成功します。アプリケーションは、期待されるデータが全て書き込まれるまで、この関数を複数回、呼び出す必要があります。または、しきい値に達するまで再試行をするようにします。

例外:

- [PortException](#) - 通信エラーが発生したとき

getParsedStatus

```
-(void) getParsedStatus: (void *) starPrinterStatus: (u_int32_t) level;
```

このメソッドは、StarIO で詳細なステータスを DK-AirCash から取得します。

戻り値:

[StarPrinterStatus](#) 構造体は、現在の DK-AirCash のステータスを保持します。

例外:

[PortException](#) – 通信エラーが発生したとき

このメソッドは [StarPrinterStatus](#) と呼ばれる StarIO の構造体を使用します。

この構造体は、ブーリアン型とバイナリーの両方の形式でプリンターステータスを保持します。

下記を行うことにより、プロジェクトで [StarPrinterStatus](#) オブジェクトを作成してください。

```
StarPrinterStatus_2 printerStatus;
[port getParsedStatus: &printerStatus : 2];

if (printerStatus.offline == SM_TRUE)
{
    if (printerStatus.coverOpen == SM_TRUE) {
        //There was a cover open error
    }
    else if (printerStatus.receiptPaperEmpty == SM_TRUE) {
        //There was a receipt paper empty error
    }
    else {
        //There was a offline error
    }
}
else {
    //If False, then the printer is online.
}
```

StarPrinterStatus 構造体 ステータスリスト

メンバ名	説明	型	詳細	DK-AirCash 対応
blackMarkError	ブラックマーク エラー	SM_BOOLEAN	DK-AirCash は非対応。	
compulsionSwitch	コンパルジョン SW	SM_BOOLEAN	ドロフのコンパルジョン SW が押されて いると SM_TRUE となる。 通常は SM_FALSE。	✓
coverOpen	カバーの状態	SM_BOOLEAN	DK-AirCash は非対応。	
cutterError	オートカッター エラー	SM_BOOLEAN	DK-AirCash は非対応。	
etbAvailable	ETB 使用可否	SM_BOOLEAN	ETB が使用可能な場合に SM_TRUE と なる。使用できない場合は SM_FALSE。	✓
etbCounter	ETB カウンタ	UCHAR	現在の ETB カウンタの値。	✓
headThermistorError	ヘッドサーミス タエラー	SM_BOOLEAN	DK-AirCash は非対応。	
offline	ON-LINE/OFF- LINE 状態	SM_BOOLEAN	オフラインの場合に SM_TRUE となる。 オンライン時は SM_FALSE。	✓
overTemp	印字ヘッド高温 による停止中	SM_BOOLEAN	DK-AirCash は非対応。	
presenterPaperJamError	プレゼンター紙 ジャムエラー	SM_BOOLEAN	DK-AirCash は非対応。	
presenterState	プレゼンタ用紙 位置	UCHAR	DK-AirCash は非対応。	
raw	ステータスのバ イト列	UCHAR[63]	ステータスのバイト列 (例: HEX 23 86 00 00 00 00 00 00)	✓
rawLength	raw の長さ	CHAR	raw の長さ	✓
receiptPaperEmpty	用紙エンド	SM_BOOLEAN	DK-AirCash は非対応。	
receiptPaperNearEmptyInner	用紙ニアエンド (内側)	SM_BOOLEAN	DK-AirCash は非対応。	
receiveBufferOverflow	受信バッファオ ーバーフロー	SM_BOOLEAN	DK-AirCash は非対応。	
unrecoverableError	復帰不可能エラ ー	SM_BOOLEAN	復帰不可能エラー(ヘッドサーミスタエラ ー、オートカッターエラー、電源電圧エ ラー等)が発生した場合に SM_TRUE と なる。通常は SM_FALSE。	✓
voltageError	電源電圧エラー	SM_BOOLEAN	DK-AirCash は非対応。	

beginCheckedBlock

```
-(void) beginCheckdBlock: (void *) starPrinterStatus: (u_int32_t) level;
```

このメソッドは、endCheckedBlock メソッドとセットで使用してデータ送信終了の確認を行います。

データ送信の直前に beginCheckedBlock を実行します。

引数:

- starPrinterStatus – StarPrinterStatus 構造体へのポインタ
 (StarPrinterStatus,StarPrinterStatus_1,StarPrinterStatus_2 の
 指定が可能だが、通常は StarPrinterStatus_2 を指定)
- level – StarPrinterStatus 構造体のレベル
 (0,1,2 の指定が可能だが、通常は 2 を指定)

サンプルコードは[こちら](#)をご参照ください。

endCheckedBlock

```
-(void) endCheckdBlock: (void *) starPrinterStatus: (u_int32_t) level;
```

このメソッドは、beginCheckedBlock メソッドとセットで使ってデータ送信終了の確認を行います。データ送信の直後に endCheckedBlock を実行します。

タイムアウト時間(*1) 内に処理が完了しなかった場合や、エラーが発生した場合は、例外 PortException をスローします。

- (*1) タイムアウト時間は、endCheckedBlockTimeoutMillis プロパティの値が使用されます。初期値は getPort で指定したタイムアウト時間となります。
endCheckedBlockTimeoutMillis の値は、印刷時間より長くなるよう調整してください。
また、10 秒未満の値が設定された場合にはタイムアウトは 10 秒になります。

引数:

- starPrinterStatus – StarPrinterStatus 構造体へのポインタ
 (StarPrinterStatus,StarPrinterStatus_1,StarPrinterStatus_2 の指定が可能だが、通常は StarPrinterStatus_2 を指定)
level – StarPrinterStatus 構造体のレベル
 (0,1,2 の指定が可能だが、通常は 2 を指定)

成功時:

StarPrinterStatus のステータスを最新のものに更新して終了します。

例外:

PortException – 通信エラー*が発生したとき

- *例) - コマンド送信自体の失敗（オフライン等）
 - タイムアウト時間内に DK-AirCash からの終了の応答がない

```

unsigned char command[] = {0x07};
uint bytesWritten = 0;

StarPrinterStatus_2 starPrinterStatus;
SMPort *port = nil;

@try
{
    port = [SMPort getPort:@"TCP:192.168.1.10" :@"":10000 ];

    //Start checking the completion of printing
    [port beginCheckedBlock:&starPrinterStatus :2];

    if (starPrinterStatus.offline == SM_TRUE)
    {
        //There was an error writing to the port
    }
    while (bytesWritten < sizeof (command))    {
        bytesWritten += [port writePort: command : bytesWritten : sizeof(command) - bytesWritten];
    }

    //End checking the completion of printing
    [port endCheckedBlock:&starPrinterStatus :2];

    if (starPrinterStatus.offline == SM_TRUE)
    {
        //There was an error writing to the port
    }
}
@catch (PortException)
{
    //There was an error writing to the port
}
@finally
{
    [SMPort releasePort:port];
}

```

disconnect

- (BOOL) disconnect

このメソッドは、指定された Bluetooth デバイスへのコネクションを切断します。
コネクションの切断後、Bluetooth デバイスは再び他の iOS 端末から接続する事ができるようになります。

このメソッドは、以下の場合に失敗となります。

- getPort で指定したタイムアウト時間内に切断が完了しなかった場合

Ethernet デバイスに対しては何も行いません。

戻り値:

成功した場合は YES を、失敗した場合は NO を返します。

有線 LAN/無線 LAN デバイスに対して実行した場合は常に YES を返します。

getFirmwareInformation

-(NSDictionary *) getFirmwareInformation:

このメソッドは、プリンターからファームウェア情報を取得します。

戻り値:

モデル名とファームウェアバージョンを NSDictionary 型で返します。

例外:

[PortException](#) – 取得に失敗したとき

Note:

- ・ 取得に失敗した場合、空文字を返します。

StarIOVersion

```
+(NSString *) StarIOVersion
```

このメソッドは、StarIO のバージョンを取得します。

戻り値:

StarIO バージョン

getDipSwitchInformation

```
-(NSDictionary *) getDipSwitchInformation
```

このメソッドは、DK-AirCash からディップスイッチ情報を取得します。

戻り値:

DK-AirCash のディップスイッチ設定情報を NSDictionary 型で返します。

例外:

[PortException](#) – 取得に失敗したとき

Note:

- ・取得に失敗した場合、空文字を返します。
- ・ファームウェアバージョン 2.0 以上で対応しています。

SMBluetoothManager クラス:

Bluetooth インターフェイスの各種設定を行うためのクラスです。

SMPort クラスと同時に使用しないでください。

●プロパティ

portName	接続先デバイスの portName を取得します。
deviceType	接続先デバイスの種類を取得します。
opened	ポートがオープンされているかを示します。
deviceName	現在の Bluetooth デバイス名を取得、設定します。
iOSPortName	StarIO で使用するポート名を取得、設定します。
autoConnect	自動接続機能の有効/無効を取得、設定します。
security	Bluetooth のセキュリティ（SSP もしくは PIN コードモード）を取得、設定します。
pinCode	Bluetooth ペアリング時に使用する PIN コードを設定します。

`@property(nonatomic, readonly) NSString *portName`

SMBluetoothManeger のインスタンスを作成します。

`@property(nonatomic, readonly) SMDeviceType deviceType`

接続先デバイスの種類を取得します。

`@property(nonatomic, readonly) BOOL opened`

ポートがオープンされているかを示します。

open メソッドが成功すると YES になります。

その後 close メソッドを呼び出すと NO になります。

`@property(nonatomic, retain) NSString *deviceName`

現在の Bluetooth デバイス名を取得、設定します。

loadSetting メソッドを呼び出した際に現在の設定値が読み込まれます。

設定するには、本プロパティを変更後 apply メソッドを実行します。

使用可能文字列： 0-9, a-z, A-Z

@property(nonatomic, retain) NSString *iOSPortName

StarIO で使用するポート名を取得、設定します。

loadSetting メソッドを呼び出した際に現在の設定値が読み込まれます。

設定するには、本プロパティを変更後 apply メソッドを実行します。

使用可能文字列：

0-9 a-z A-Z ; : ! ? # \$ % & , . @ _ - = スペース / * + ~ ^ [{ () }) | \

@property(nonatomic, assign) BOOL autoConnect

自動接続機能の有効/無効を取得、設定します。

loadSetting メソッドを呼び出した際に現在の設定値が読み込まれます。

設定するには、本プロパティを変更後 apply メソッドを実行します。



security プロパティが PIN コード設定の場合は、この値に NO を設定してください。

@property(nonatomic, assign) SMBluetoothSecurity security

Bluetooth のセキュリティ（SSP もしくは PIN コード）を取得、設定します。

loadSetting メソッドを呼び出した際に現在の設定値が読み込まれます。

設定するには、本プロパティを変更後 apply メソッドを実行します。



PIN コード設定を使用する場合は、autoConnect プロパティに NO を設定してください。

@property(nonatomic, retain) NSString *pinCode

Bluetooth インターフェイスの PIN コードを設定します。

現在の設定値を取得することはできません。

現在の PIN コードから値を変更しない場合は nil を指定します。

使用可能文字列： 0-9, a-z, A-Z

●メソッド

initWithPortName

```
-(id) initWithPortName: (NSString *) portName deviceType: (SMDeviceType) deviceType
```

SMBluetoothManager のインスタンスを作成します。

引数:

- | | |
|------------|---|
| portName | – 接続するデバイスのポート名
例) "BT:Star Micronics" |
| deviceType | – 接続するデバイスの種類
SMDeviceTypeDesktopPrinter |

戻り値:

成功時は SMBluetoothManager のインスタンスを返します。
失敗時は nil を返します。

open

```
- (BOOL) open
```

Bluetooth プリンターとの接続を開きます。

戻り値:

成功した場合は YES を、失敗した場合は NO を返します。

apply

- (BOOL) apply

deviceName, iOSPortName, autoConnect, security, pinCode プロパティの値をデバイスに適用します。

戻り値:

成功した場合は YES を、失敗した場合は NO を返します。



apply メソッドで適用した値は、デバイスの電源再投入・再ペアリングを行った後に有効になります。

close

- (void) close

Bluetooth プリンターとの接続を閉じます。

loadSetting

- (BOOL) loadSetting:

Bluetooth インターフェイスカードの設定情報を取得します。

戻り値:

成功した場合は YES を、失敗した場合は NO を返します。

StarIO iOS SDK 機能

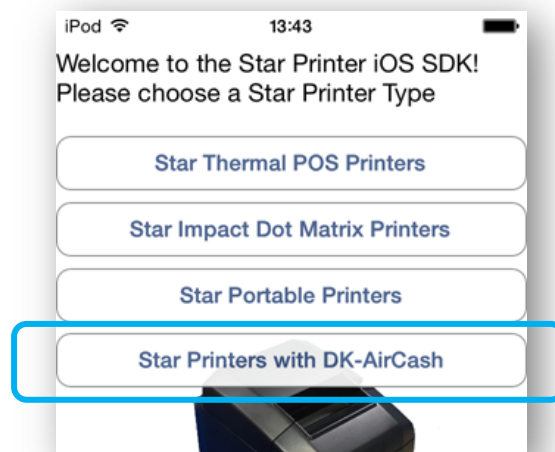
SDK 機能と StarIO のコマンドについての概要

これらのコマンドは、全て Star ラインモードコマンド仕様書に記載されています。

また、この SDK では詳細情報の参照先が、Star ラインモードコマンド仕様書のページやセクションとなっています。そのため、特定のコマンドに関する詳細な情報が必要な場合は、Star ラインモードコマンド仕様書をダウンロードし参照してください。

デバイスとコマンド種類の選択

- 1) "Star Printers with DK-AirCash"をタップします。



サンプル機能対応リスト

Command Samples Include:

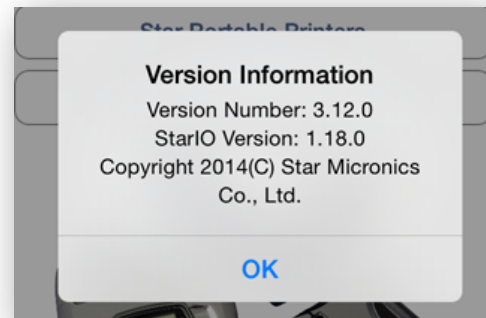
- [Get StarIO Version](#)
- [Port Discovery](#)
- [Get DK-AirCash Firmware Information](#)
- [Get DK-AirCash Dip Switch Information](#)
- [Get Printer Status](#)
- [Get DK-AirCash Status](#)
- [Sample Receipt + Open Cash Drawer via DK-AirCash](#)
- [Open Cash Drawer via DK-AirCash](#)
- [Bluetooth Pairing + Connect](#)
- [Bluetooth Disconnect](#)
- [DK-AirCash Bluetooth Setting](#)

Help



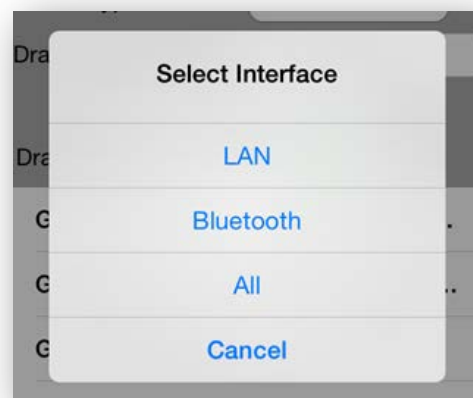
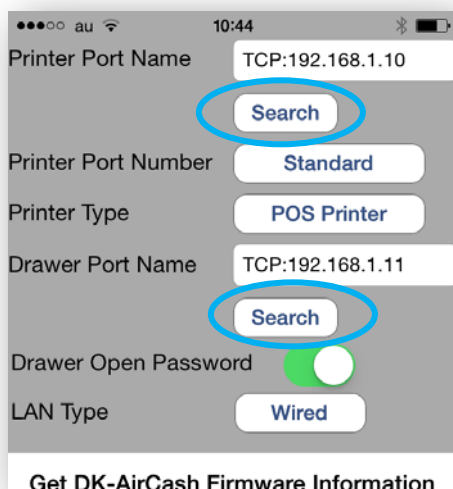
Help は、StarIO ポートの設定に関する情報を表示します。

Get StarIO Version



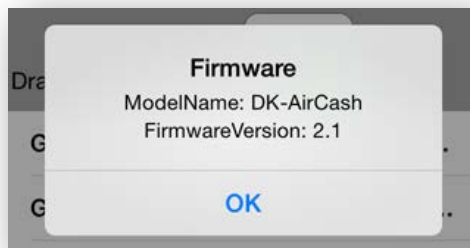
About をタップすると、StarIO のバージョンを表示します。

Port Discovery



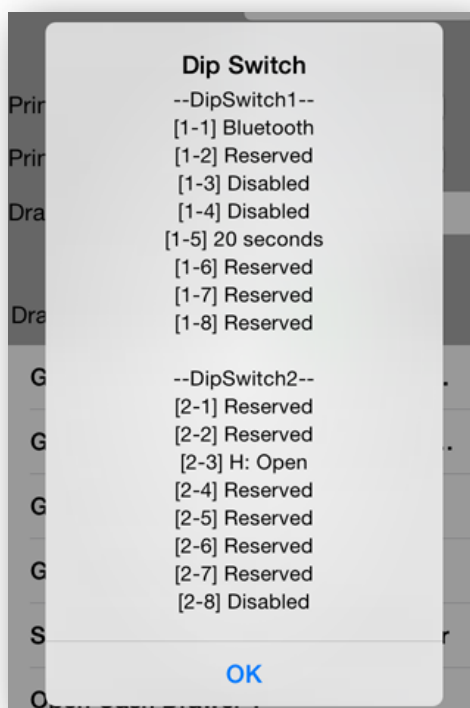
ネットワークに接続されている DK-AirCash を自動的に検索します。検索結果より、接続したいデバイスを選択してください。[詳細はこちらをご参照ください。](#)

Get DK-AirCash Firmware Information

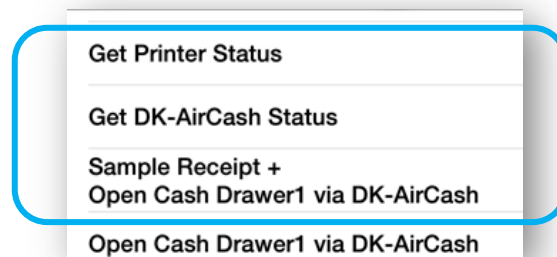


Drawer Port Name に設定された DK-AirCash のファームウェア情報を表示します。

Get DK-AirCash Dip Switch Information

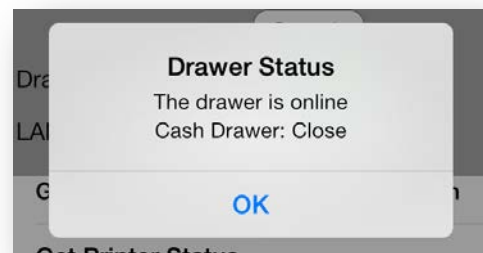
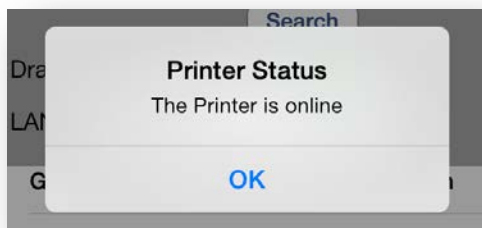


Drawer Port Name に設定された DK-AirCash のディップスイッチ設定情報を表示します。



Get Printer Status / Get DK-AirCash Status

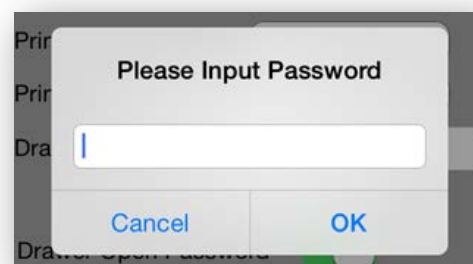
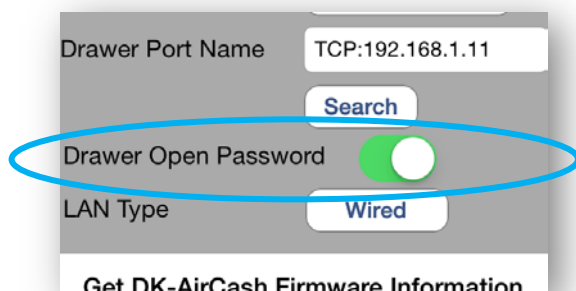
接続されたプリンターまたはキャッシュドロワのステータスを表示します。



Sample Receipt + Open Cash Drawer via DK-AirCash

プリンターが接続されていればサンプルレシートの印刷をします。また、キャッシュドロワが接続されていればキャッシュドロワをオープンします。

Drawer Open Password に ON が設定されていれば、キャッシュドロワオープンの前にパスワードの入力を要求します。設定されているパスワードは「1234」です。



プリンターが接続されていない場合、キャッシュドロワは作動しません。



Open Cash Drawer1 via DK-AirCash /

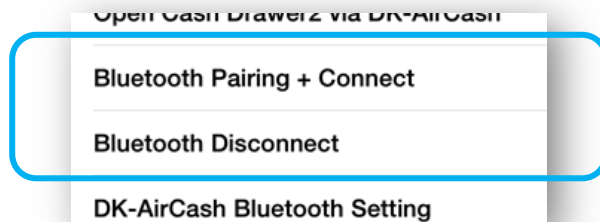
Open Cash Drawer2 via DK-AirCash

キャッシュドロアが接続されていればキャッシュドロアをオープンします。

Drawer Open Password に ON が設定されていれば、キャッシュドロワーオープンの前にパスワードの入力を要求します。設定されているパスワードは「1234」です。



Open Cash Drawer1 と Open Cash Drawer2 を同時に実行することはできません。



Bluetooth Pairing + Connect

タップすると近くにある接続可能な全ての Bluetooth デバイスの一覧が表示されます。

選択すると、セキュリティが SSP(Default)の場合は、そのデバイスとのペアリング、コネクションが行われます。セキュリティが PIN コード設定の場合は、コネクション機能のみ対応となります。あらかじめデバイスとペアリングをしている必要があります。

StarIO の機能は使用していません。

*Note:*この機能には iOS6 以降が必要です。

Bluetooth Disconnect

タップするとペアリング+コネクションが行われた Star 製プリンターの一覧が表示されます。選択すると、そのデバイスとのコネクションが切断され、他のホストからそのデバイスに対して接続できるようになります。

この機能は“disconnect”メソッドを使用しています。



モバイルプリンターには対応していません。
(POS プリンター、DK-AirCash のみ対応)

DK-AirCash Bluetooth Setting

iPod 13:14

Device Name DK-AirCash

iOS Port Name DK-AirCash

Auto Connect ☒

Security SSP

Change PIN Code No Change

New PIN Code

Back Apply

Drawer Port Name に指定された DK-AirCash に接続して、Bluetooth インターフェ이스の各種設定を変更します。



変更された値は、デバイスの電源再投入・再ペアリングを行った後に有効になります。

StarIO を使用するアプリケーション開発のために

#1: 大規模なプロジェクトをコーディングしている場合、全ての印刷メソッドを抽象化してクラスを作成してください。これはソース・コードの再利用に役立ちます。また、特定のコードを見つけるのが容易になり、時間の節約にもなります。StarIO を唯一のクラスに存在させることによって、オブジェクト指向プログラミングを実現できます。

#2: ASCII と Unicode、16 進と 10 進、及び、Byte と Char の違いと定義が何であることを確認してください。1 バイトは、通常、8 桁の 2 進数(1 と 0)で 8 ビットです。これらのバイトは、ちょうど 8 ビットのバイナリーデータです。しかし、byte は int または char でもありえます。3 つの異なる変数の型は、基本的に同じ方法でデータを保持しますが、わずかな違いがあります。印刷ジョブのデータを格納する変数を選択する際、byte の代わりに char、int、または string で試してみてください。

ASCII から Unicode(また逆も同様に)への変換は、時として安全ではありません。そのため、Encoding クラスがどのように機能するのを確認してください。Unicode で見られる間違いの例として、“Culture-sensitive searching and casing”、“Surrogate pairs”、“Combining characters”、“Normalization”などがあります。

#3: 16 進ダンプモードについて 作成したアプリケーションのデバッグの際、デバイスの 16 進ダンプモードを使用してください。16 進ダンプモードを使用することで、アプリケーションから送信したデータをデバイスが正しく受信できたか確認する事ができます。16 進ダンプモードへの設定は、各デバイスの製品仕様書を参照ください。

#4: StarIO コマンド・コードをリバース・エンジニアリングしないでください。全ての StarIO コマンドは、コマンド仕様書より参照可能です。また、本 SDK を活用することで、アプリケーション作成時の工数を大幅に削減可能です。

#5: 本 SDK に記載されていないコマンドについては、SDK のコードサンプルを参照してください。また、[スター精密グローバルサポートサイト](#)の Developers セクションにアクセスすることで、より詳細な情報を入手可能です。

#6: iOS 以外の OS(例: Android) に対応した SDK をお探しの場合には、[スター精密グローバルサポートサイト](#)の Developers セクションをご覧ください。

追加リソース

以下リンクよりプログラマーマニュアルを入手してください。

[スター精密グローバルサポートサイト](#)

FAQ を参照してください。

グローバルサポートサイトより、以下の情報を入手できます。

- 最新バージョンの SDK マニュアル／ソース・コード
- アドイス／業界情報
- スター精密デバイスドライバ
- 技術的な質問／サポート

[Apple Developer Site](#)

Apple の公式開発リソース

[Apple Developer Site Resources](#)

Apple ライブラリ - 開発者のためのドキュメントに関する情報の入手

[Unicode.org](#)

ユニコードコンソーシアム - Unicode の詳細について

[1D Barcodes](#)

Barcode Island - バーコードの詳細について

[2D Barcodes](#)

[QR Codes](#)、[PDF417](#) に関する情報について

[Code Pages](#)

コード・ページに関する情報について

ASCII コード表

10進	16進	文字	10進	16進	文字	10進	16進	文字	10進	16進	文字
0	0	NUL	16	10	DLE	32	20	(space)	48	30	0
1	1	SOH	17	11	DC1	33	21	!	49	31	1
2	2	STX	18	12	DC2	34	22	"	50	32	2
3	3	ETX	19	13	DC3	35	23	#	51	33	3
4	4	EOT	20	14	DC4	36	24	\$	52	34	4
5	5	ENQ	21	15	NAK	37	25	%	53	35	5
6	6	ACK	22	16	SYN	38	26	&	54	36	6
7	7	BEL	23	17	ETB	39	27	'	55	37	7
8	8	BS	24	18	CAN	40	28	(56	38	8
9	9	TAB	25	19	EM	41	29)	57	39	9
10	A	LF	26	1A	SUB	42	2A	*	58	3A	:
11	B	VT	27	1B	ESC	43	2B	+	59	3B	;
12	C	FF	28	1C	FS	44	2C	,	60	3C	<
13	D	CR	29	1D	GS	45	2D	-	61	3D	=
14	E	SO	30	1E	RS	46	2E	.	62	3E	>
15	F	SI	31	1F	US	47	2F	/	63	3F	?

10 進 16 進 文字			10 進 16 進 文字			10 進 16 進 文字			10 進 16 進 文字		
64	40	@	80	50	P	96	60	`	112	70	p
65	41	A	81	51	Q	97	61	a	113	71	q
66	42	B	82	52	R	98	62	b	114	72	r
67	43	C	83	53	S	99	63	c	115	73	s
68	44	D	84	54	T	100	64	d	116	74	t
69	45	E	85	55	U	101	65	e	117	75	u
70	46	F	86	56	V	102	66	f	118	76	v
71	47	G	87	57	W	103	67	g	119	77	w
72	48	H	88	58	X	104	68	h	120	78	x
73	49	I	89	59	Y	105	69	i	121	79	y
74	4A	J	90	5A	Z	106	6A	j	122	7A	z
75	4B	K	91	5B	[107	6B	k	123	7B	{
76	4C	L	92	5C	¥	108	6C	l	124	7C	
77	4D	M	93	5D]	109	6D	m	125	7D	}
78	4E	N	94	5E	^	110	6E	n	126	7E	~
79	4F	O	95	5F	_	111	6F	o	127	7F	□

SDK パッケージ改訂履歴

[illegible]



Star Micronics is a global leader in the manufacturing of small printers. We apply over 50 years of knowhow and innovation to provide elite printing solutions that are rich in stellar reliability and industry-respected features. Offering a diverse line of Thermal, Hybrid, Mobile, Kiosk and Impact Dot Matrix printers, we are obsessed with exceeding the demands of our valued customers every day.

We have a long history of implementations into Retail, Point of Sale, Hospitality, Restaurants and Kitchens, Kiosks and Digital Signage, Gaming and Lottery, ATMs, Ticketing, Labeling, Salons and Spas, Banking and Credit Unions, Medical, Law Enforcement, Payment Processing, and more!

High Quality POS Receipts, Interactive Coupons with Triggers, Logo Printing for Branding, Advanced Drivers for Windows, Mac and Linux, Complete SDK Packages, Android, iOS, Blackberry Printing Support, OPOS, JavaPOS, POS for .NET, Eco-Friendly Paper and Power Savings with Reporting Utility, ENERGY STAR, MSR Reading, *future*PRNT, StarPRNT... How can Star help you fulfill the needs of your application?

Don't just settle on hardware that won't work as hard as you do. Demand everything from your printer. Demand a Star!

Star Micronics Worldwide

Star Micronics Co., Ltd.
536 Nanatsushinya
Shimizu-ku, Shizuoka 424-0066
Japan
+81-54-347-2163
<http://www.star-m.jp/eng/index.htm>

Star Micronics America, Inc.
65 Clyde Road. Suite G
Somerset, NJ 08873
USA
1-848-216-3300
<http://www.starmicronics.com>

Star Micronics EMEA
Star House
Peregrine Business Park, Gomm Road
High Wycombe, Buckinghamshire HP13
7DL
UK
+44-(0)-1494-471111
<http://www.star-emea.com>

Star Micronics Southeast Asia Co., Ltd.
Room 2902C. 29th Fl. United Center
Bldg.
323 Silom Road, Silom Bangrak,
Bangkok 10500
Thailand
+66-2-631-1161 x 2
<http://www.starmicronics.co.th/>