



iOS Software Development Kit

POS プリンター(Star Line)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 以上
- StarIO iOS SDK

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 対応デバイスリスト**

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 プリンター対応リスト](#)
- ❖ [Star POS プリンターを iOS デバイスに接続するには](#)
- ❖ [はじめに](#)
- ❖ [Star POS プリンターで SDK を使用する](#)
- ❖ [iOS SDK 概要](#)
- ❖ [StarIO framework](#)
 - [プロジェクトに StarIO.framework を追加する](#)
- ❖ [メソッド概要](#)
 - [SMPort クラス](#)
 - [SMBluetoothManager クラス](#)
- ❖ [StarIO iOS SDK 機能](#)
 - [Port Discovery](#)
 - [Help](#)
 - [Get Firmware Information](#)
 - [Get Status](#)
 - [Sample Receipt Printing](#)
 - [Open Cash Drawer](#)
 - [1D Barcodes](#)
 - [2D Barcodes](#)
 - [Cut](#)
 - [Text Formatting](#)
 - [Raster Graphics Text Printing](#)
 - [Image File Printing](#)
 - [AllReceipts](#)
 - [Bluetooth Pairing + Connect](#)
 - [Bluetooth Disconnect](#)
 - [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 POS プリンターが通信を行う、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写・複製・転載することは、固くお断りします。

Star POS プリンター 対応リスト

iOS における Star POS プリンターのモデル別対応リストを以下に記します。

◆Line Mode / Raster Mode

✓ : Line Mode のみ対応 ✓ : Raster Mode のみ対応

Star プリンター		Port Discovery	Get Firmware Information	Get Status	Sample Receipts	Sample Receipts (UTF-8)	Open Drawer	1D Barcodes	2D Barcodes	Cut Patterns	Text Formatting	Text Formatting (UTF-8)	Raster Graphics	Image File Printing	AllReceipts	Bluetooth Pairing + Connect	Bluetooth Disconnect
モデル	インターフェイス																
TSP100IIIW	Ethernet	✓	✓ ***	✓	✓		✓						✓	✓	✓		
TSP100IIILAN	Ethernet	✓	✓ ***	✓	✓		✓						✓	✓	✓		
TSP100LAN	Ethernet	✓	✓ ***	✓	✓		✓						✓	✓	✓		
TSP650	Ethernet	✓	✓	✓	✓		✓	✓		✓	✓		✓	✓	✓		
TSP650II	Ethernet	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		
	Bluetooth	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FVP10	Ethernet	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		
	Bluetooth	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
TSP700II	Ethernet	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		
	Bluetooth	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
TSP800II	Ethernet	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		
	Bluetooth	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓

TUP500	Ethernet	✓	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓	✓		
TSP100U*	USB		✓ ** ***	✓	✓		✓						✓	✓	✓		
TSP100GT*	USB		✓ ** ***	✓	✓		✓						✓	✓	✓		
TSP100ECO*	USB		✓ ** ***	✓	✓		✓						✓	✓	✓		
SP700	Ethernet	✓	✓	✓	✓		✓			✓	✓						
	Bluetooth	✓	✓	✓	✓		✓			✓	✓					✓	✓

* USB インターフェイスプリンターを使用するには、Apple AirMac シリーズが必要です。

** Apple AirMac シリーズを使用する場合、USB モデルは空文字を返します。

*** TSP100U,TSP100GT,TSP100LAN,TSP100ECO,TSP100IIIW,TSP100IIILAN はファームウェアバージョンが取得できません。

Note 1: TSP100 シリーズは Raster Mode で動作しています。この SDK のサンプルプログラムは Line Mode を使用して作成しているので、TSP100 シリーズでは機能しないものがあります。詳細は STAR Line Mode コマンド仕様書の「3-4. グラフィックスコマンド」の項をご参照ください。

Note 2: この SDK は一般的な機能について書いてあり、全ての機能を搭載しておりません(例えば TUP500 のプレゼンタ機能等)。アプリケーションに含まれていないコマンドについては、コマンド仕様書をご参照ください。

Star POS プリンターを iOS デバイスに接続するには

Ethernet インターフェイス

Star POS プリンターは、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 を利用できる環境の場合、POS プリンターが自動的に 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。

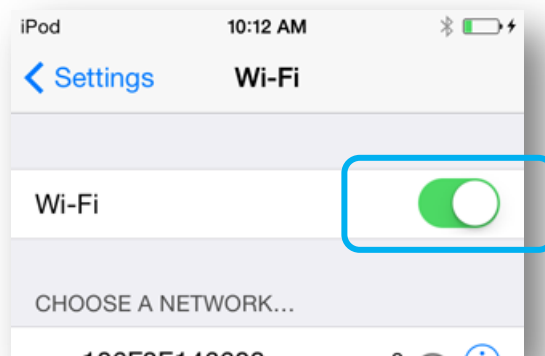
また、プリンター本体の#9100 Multi Session を無効に設定して使用してください。設定は、FEED ボタンを押しながら電源スイッチを ON することで実行される自己印字で確認できます。

#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。

POS プリンターの Ethernet 設定は、TSP100 シリーズ（TSP100III 含む）では TSP100III Utility*、それ以外のモデルでは Star Setting Utility*を使用して設定することができます。TSP100III Utility、Star Setting Utility は App Store よりダウンロードしてください。

* これらの Utility は、IP アドレスの割り当てられていない（IP アドレス：0.0.0.0）プリンターでは使用できません。

1. Star POS デバイスに 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 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

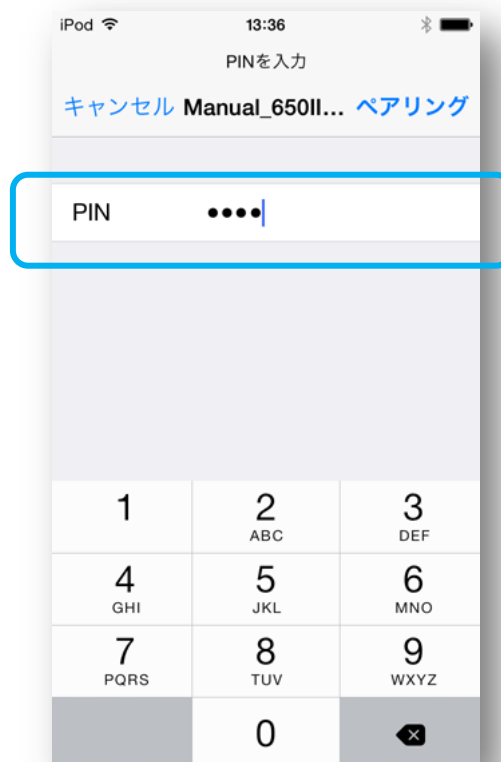
1. Star POS プリンターを、ペアリングを行う iOS デバイスと接続が可能な範囲に設置して電源を投入します。
2. iOS の[設定]より、[Bluetooth]をタップします。



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 プリンターをタップします。



4. PIN を入力します。（Star POS プリンターの 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、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。

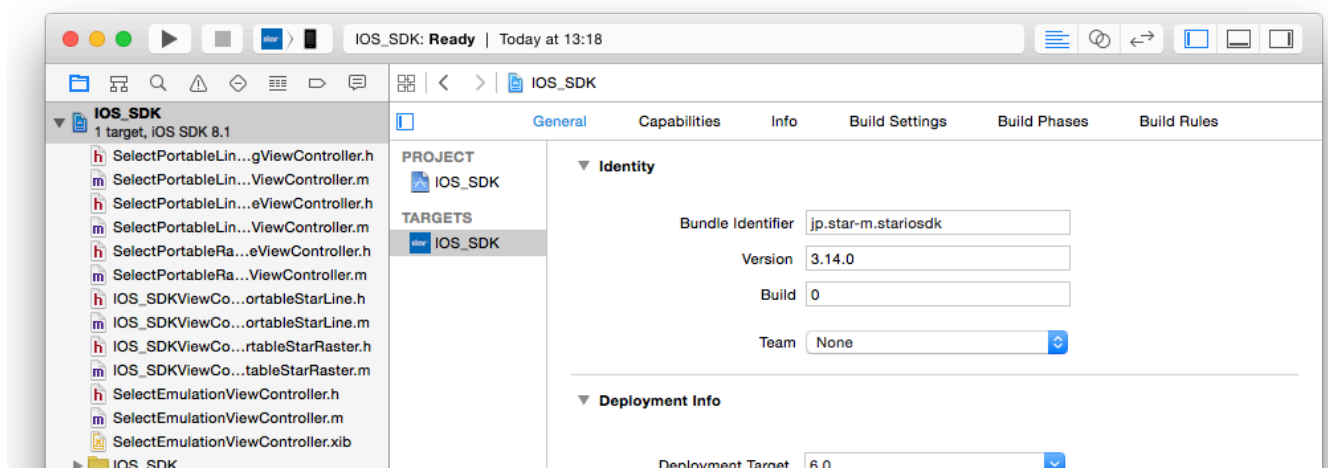
事前に、開発を行う Mac に Xcode のインストールを行ってください。万一、サポートまたは追加情報が必要な場合は、Apple Developer サイトの [Resources](#) セクションを参照してください。

Xcode で Star iOS SDK プロジェクトを開く方法：

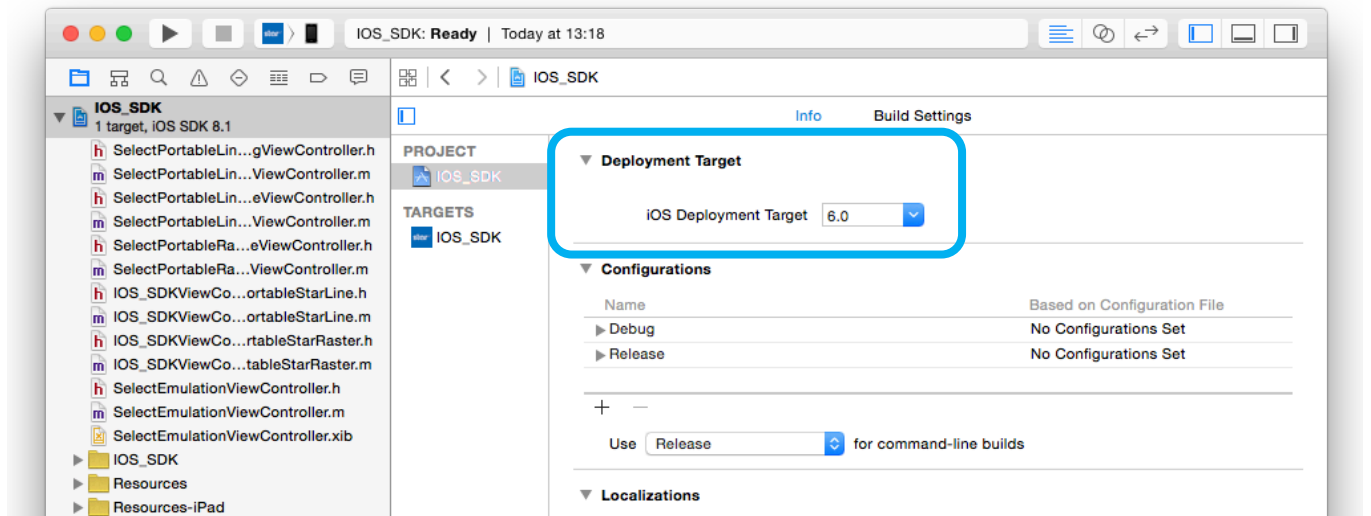
- 1) Star iOS SDK フォルダを解凍し、開きます。



- 2) IOS_SDK.xcodeproj を開きます。

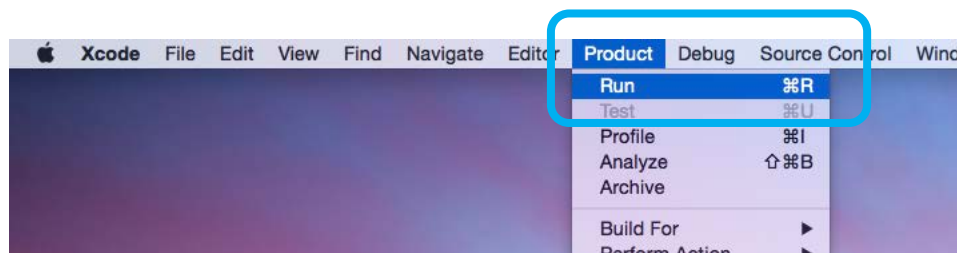


3) iOS Deployment Target の欄にて 6.0 以降を選択してください。



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 POS Printer で SDK を使用する

[iOS に対応する Star POS プリンター](#)が手元にあることを確認してください。

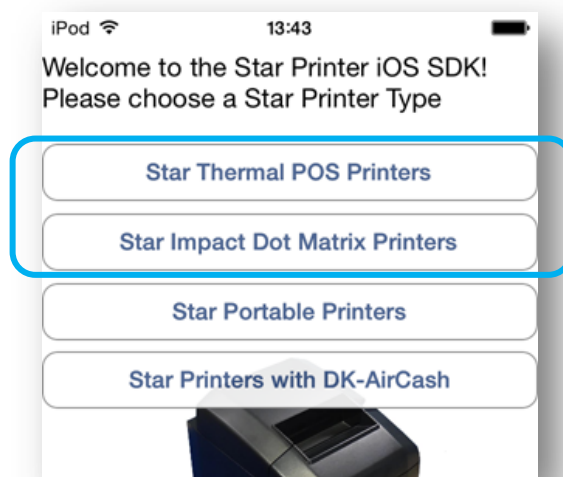
ポート名の設定:

StarIO は、プリンター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プリンターとの通信を行えません。また、用途に合わせて PortSettings 等の設定も必要です。詳しくはこちらをご参照ください。

Interface	Port Name
Ethernet (TCP/IP)	TCP:"IP アドレス"
Bluetooth	BT:"iOS ポート名"

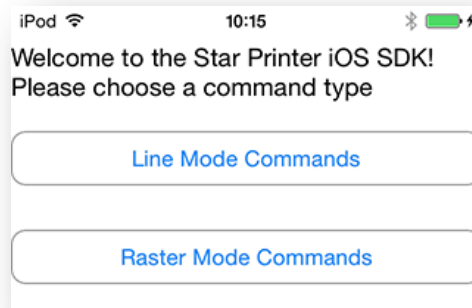
POS プリンターの設定

- 1) サーマルプリンターを使用する場合は"Star Thermal POS Printers"を、ドットプリンターを使用する場合は"Star Impact Dot Matrix Printers"をタップします。



- 2) 使用するコマンドの種類をタップしてください。コマンドの種類によってプリンターへのデータ送信方法が変わります。

ドットプリンターの場合は対応する Line Mode Command の画面に移ります。



Line Mode Commands:

プリンターへのデータ送信を1行ごとに行います。データは小さな単位でプリンターに送信されるため、開発者が任意の場所のコマンドでカスタマイズを行うのに適しています。

このモードではプリンターにインストールされているデバイスフォントのみ利用できます。

ラインモード対応モデル:

TSP650, TSP650II, FVP10, TSP700II, TSP800II, TUP500, SP700

Raster Mode Commands:

プリンターはすべての情報をグラフィックイメージで受信します。それにより、見栄えの良い TrueType フォントを利用して、高い印字速度での印刷を実現できます。

ラスターコマンドでは、印字データをプリンターに送信する前にレシート全体のグラフィックデータを生成する必要があり、ラインモードコマンドよりも複雑なコーディングが必要になります。

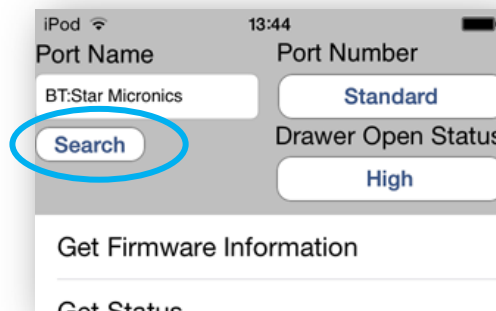
ラスターモード対応モデル:

TSP100LAN, TSP650, TSP650II, FVP10, TSP700II, TSP800II, TUP500
TSP100U, TSP100GT, TSP100ECO, TSP100IIIW, TSP100IIILAN

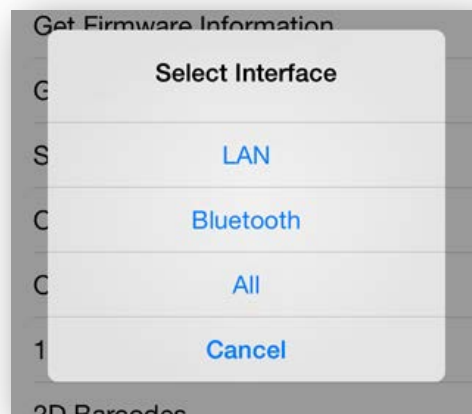
Ethernet／Bluetooth プリンターの設定

・接続するプリンターを検索して設定する

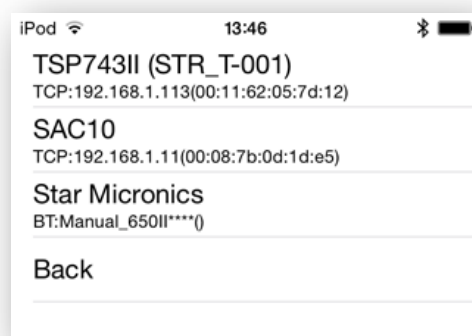
- 1) ネットワークに接続されているすべての Star の Ethernet／Bluetooth プリンターを検索するために、“Search”をタップします。



- 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。
LAN の場合、接続可能な 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。

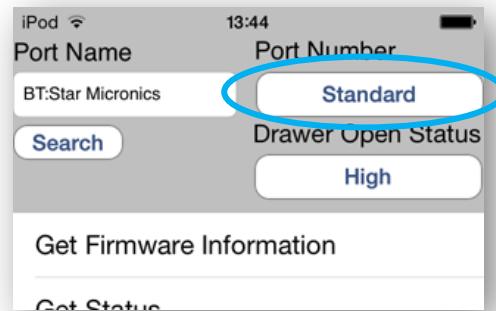


- 3) 検索されたプリンター一覧から、接続するプリンターの名前をタップしてください。



- 4) Ethernet 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 シリーズ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。

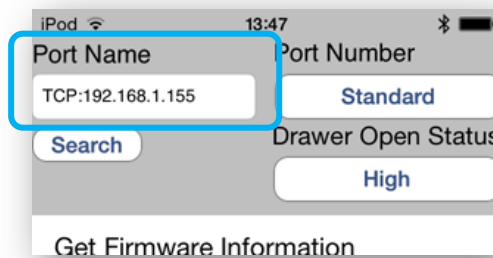


・接続するプリンターを手動で設定する

1) 「PortName」に以下のように手動で入力します。

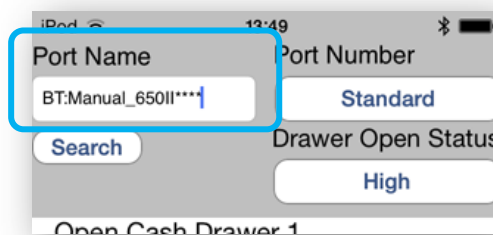
TCP:<IP アドレス>

"TCP:"の後にプリンターの IP アドレスを追加します。



BT: <Bluetooth デバイス名>

"BT:"の後に Bluetooth デバイス名を入力します。



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_SDKViewController.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 POS プリンターの起点となります。

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 中の 1D barcode Code128 に相当します。

全ての機能が両方のプリンター・タイプ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。(以下に再リストします。)

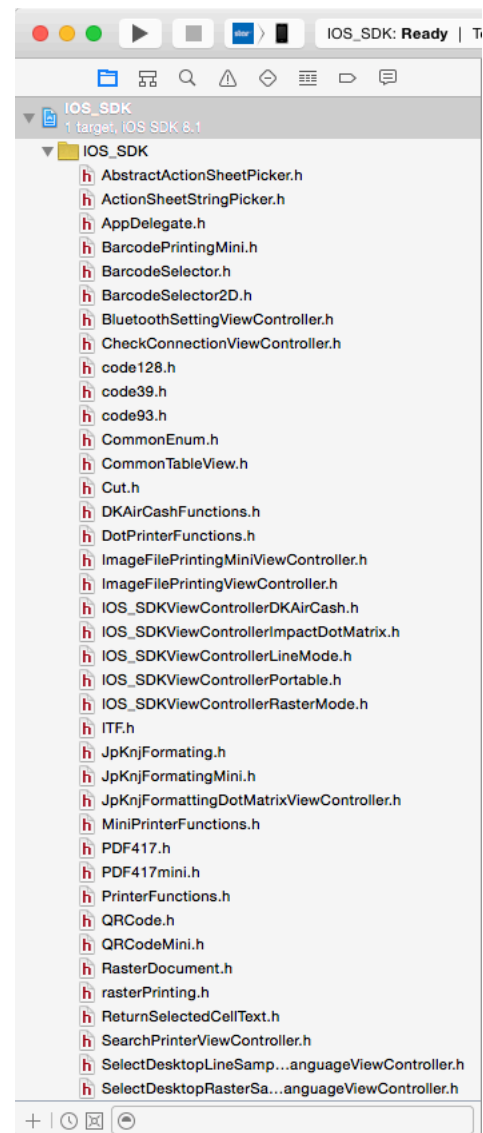
POS プリンター：

- ・磁気カードリーダー非対応

モバイル プリンター：

- ・キャッシュドロフ非対応
- ・カッター動作非対応

また、“Mini”を含むソース・ファイルは、モバイルプリンターを ESC/POS モードで使用する場合のサンプルコードです。StarBitmap.m は、両方のプリンター・タイプに適用されます。



StarIO Framework

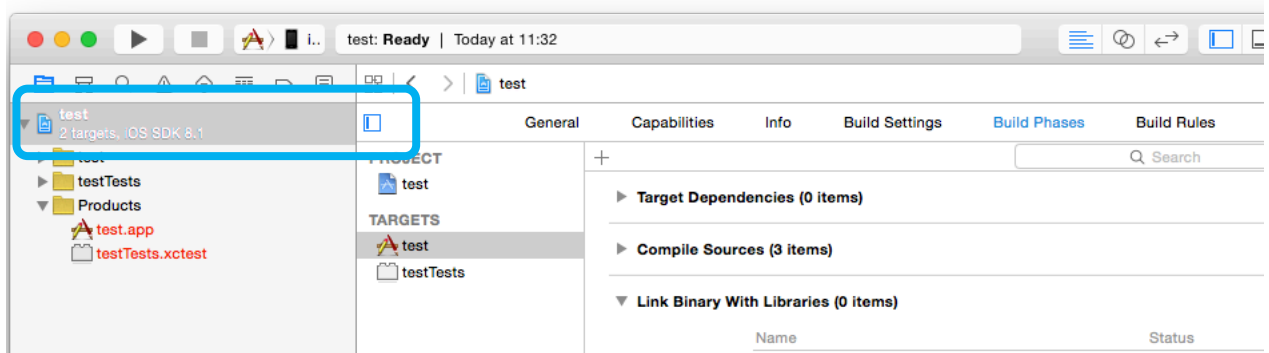
Star iOS SDK プロジェクトには、既に StarIO framework は含まれています。(この SDK をテストする場合、そのまま使用できます)

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。

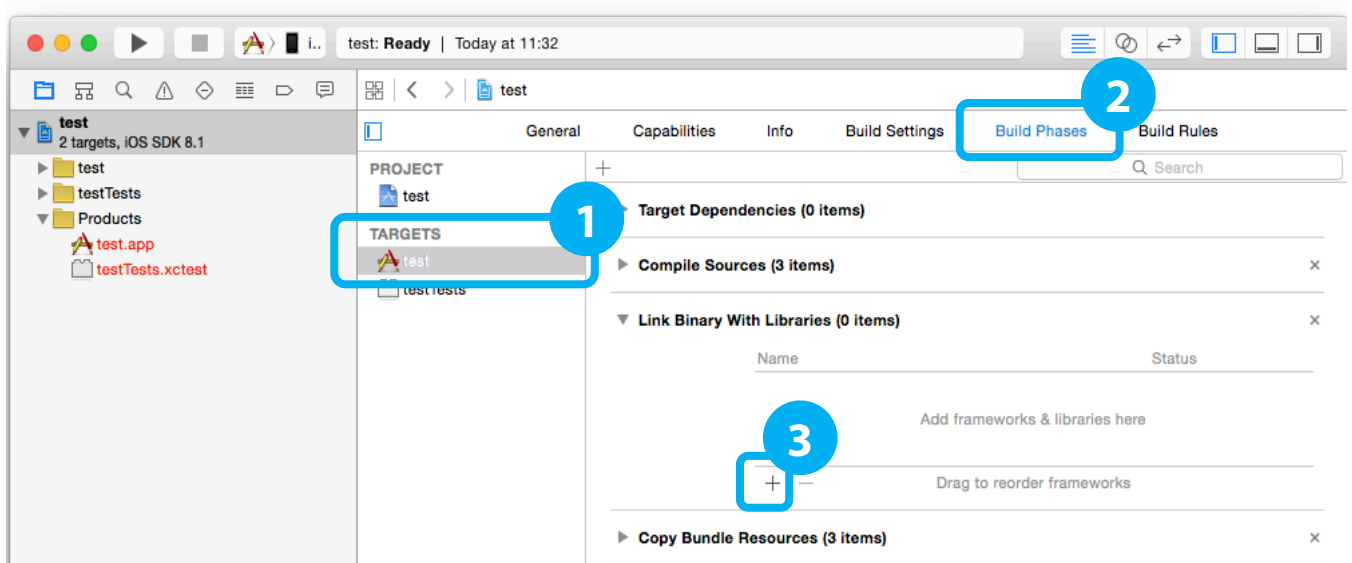
●新規のアプリケーションを作成するには

1. プロジェクトに StarIO.framework を追加する

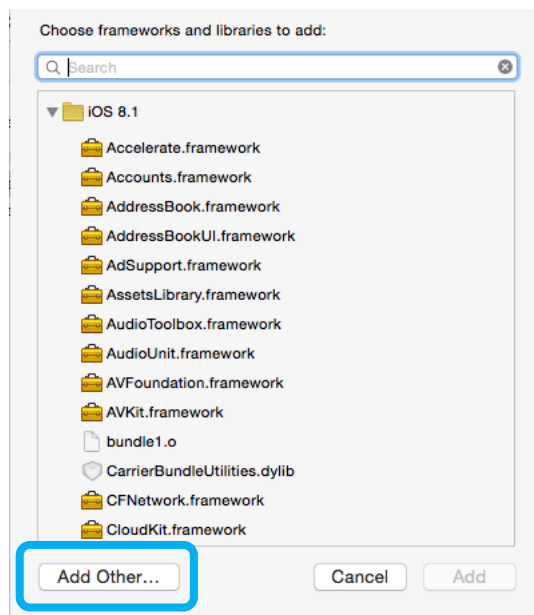
1. 新規に作成したプロジェクトをクリックします。



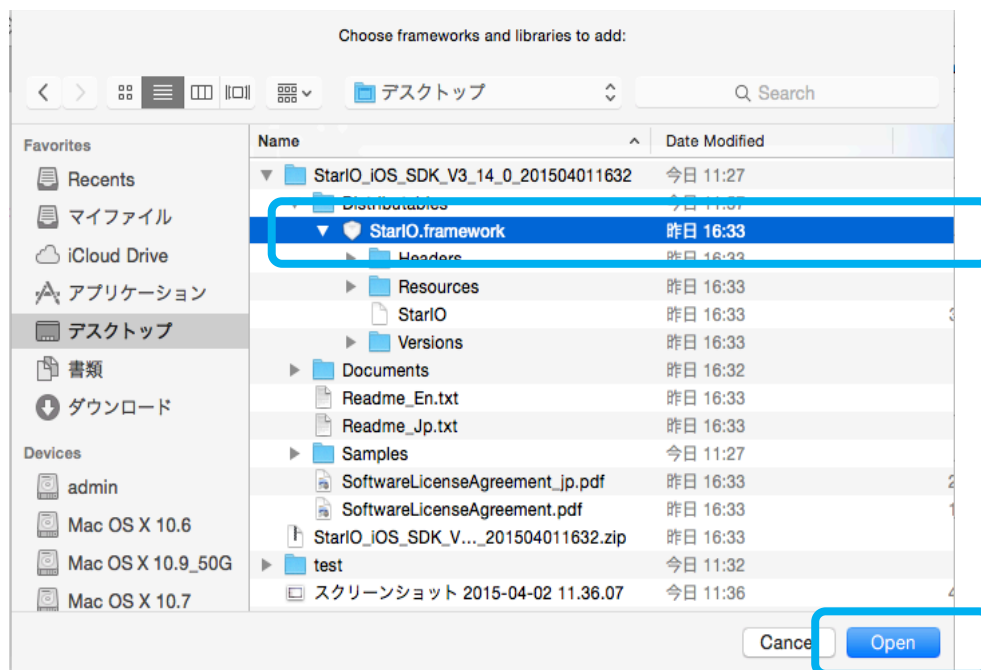
2. ターゲット → “Build Phases” → Link Binary With Libraries の“+” をクリックします。



3. “Add Other…”ボタンをクリックします。



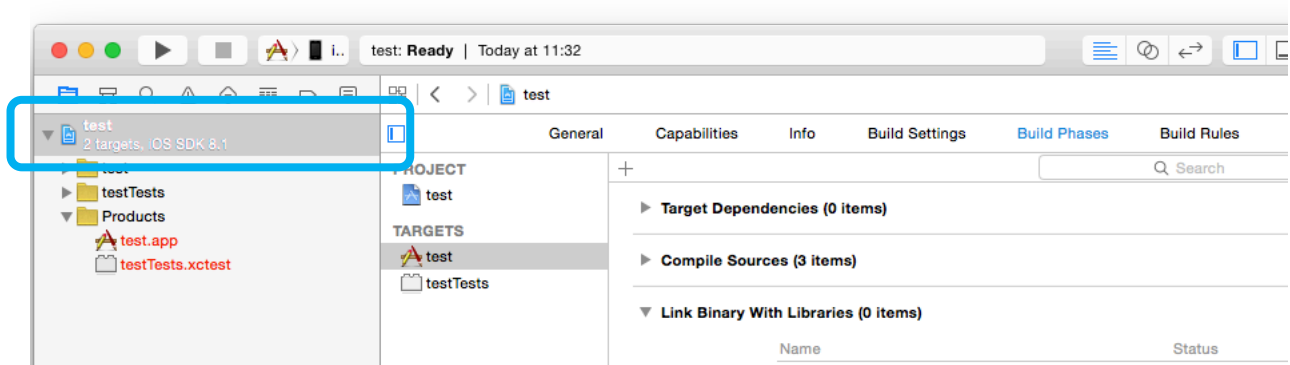
4. Star iOS SDK を解凍した場所の StarIO.framework フォルダを参照し、“Open”をクリックします。



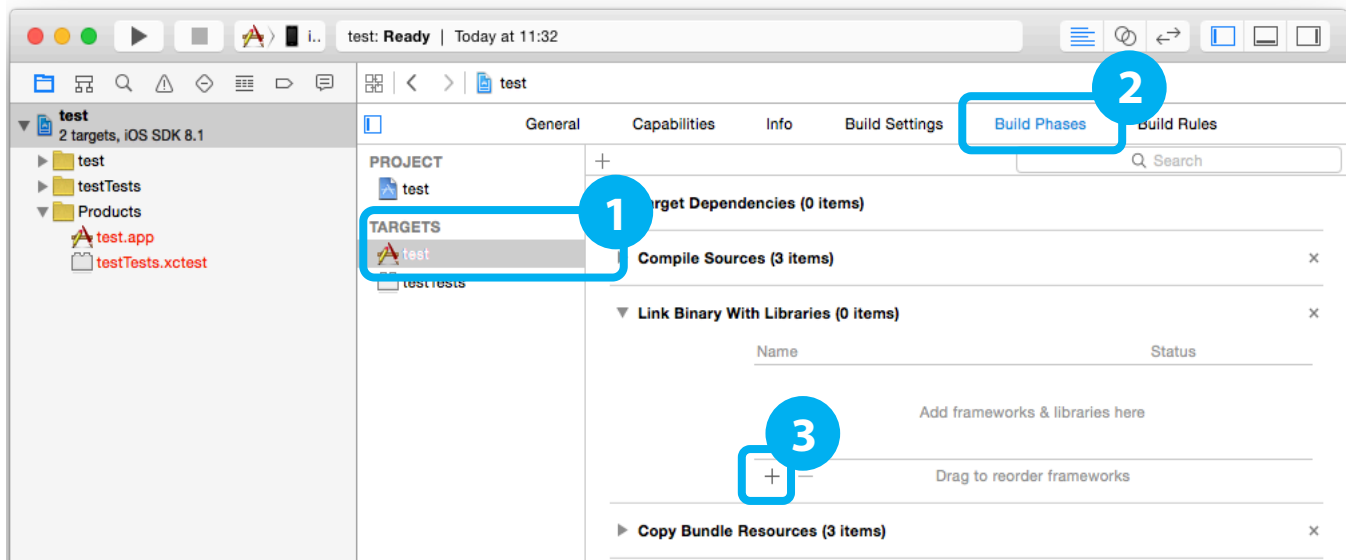
5. StarIO framework はプロジェクトに追加されます。

2. その他の framework を追加する

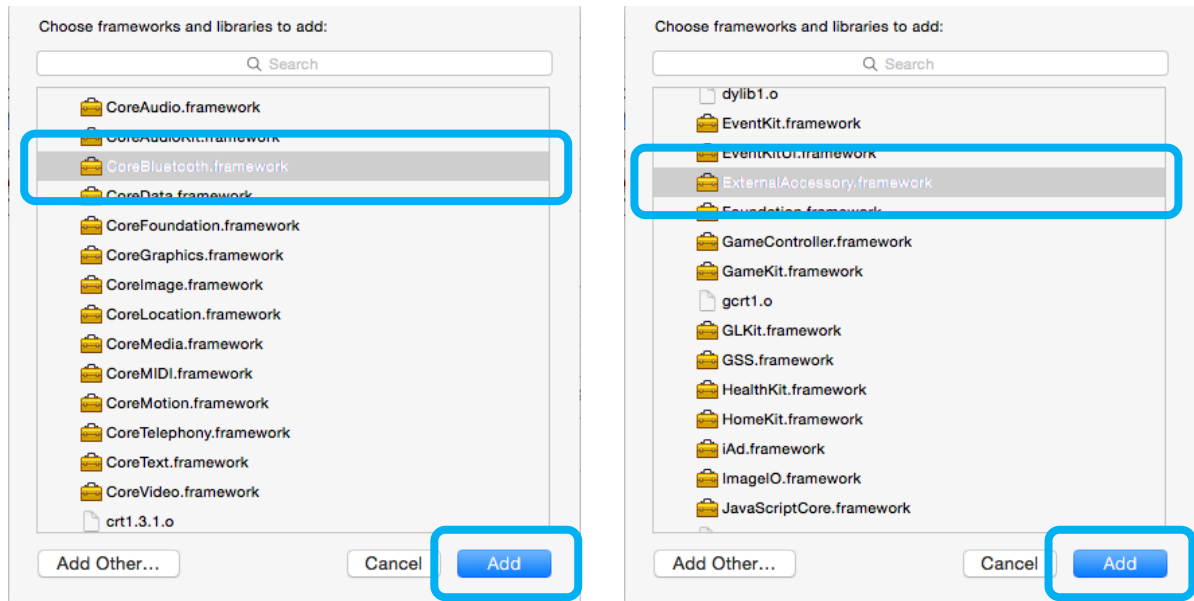
1. 新規に作成したプロジェクトをクリックします。



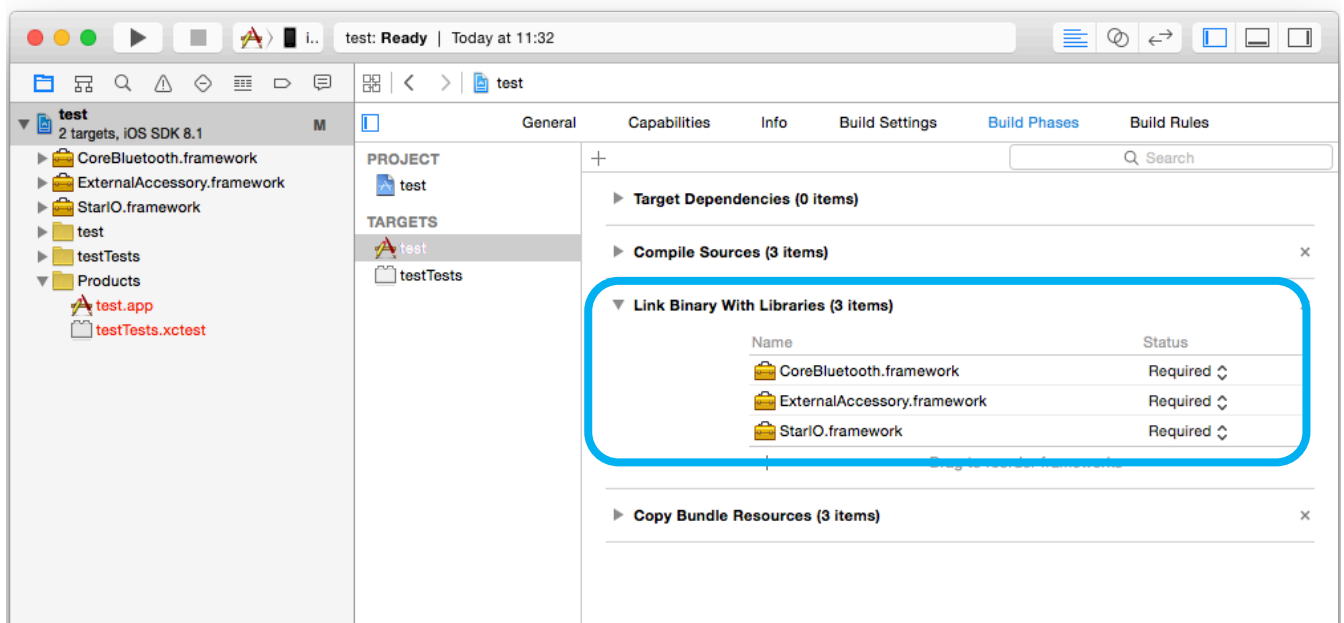
2. ターゲット → “Build Phases” → Link Binary With Libraries の“+” をクリックします。



- 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。



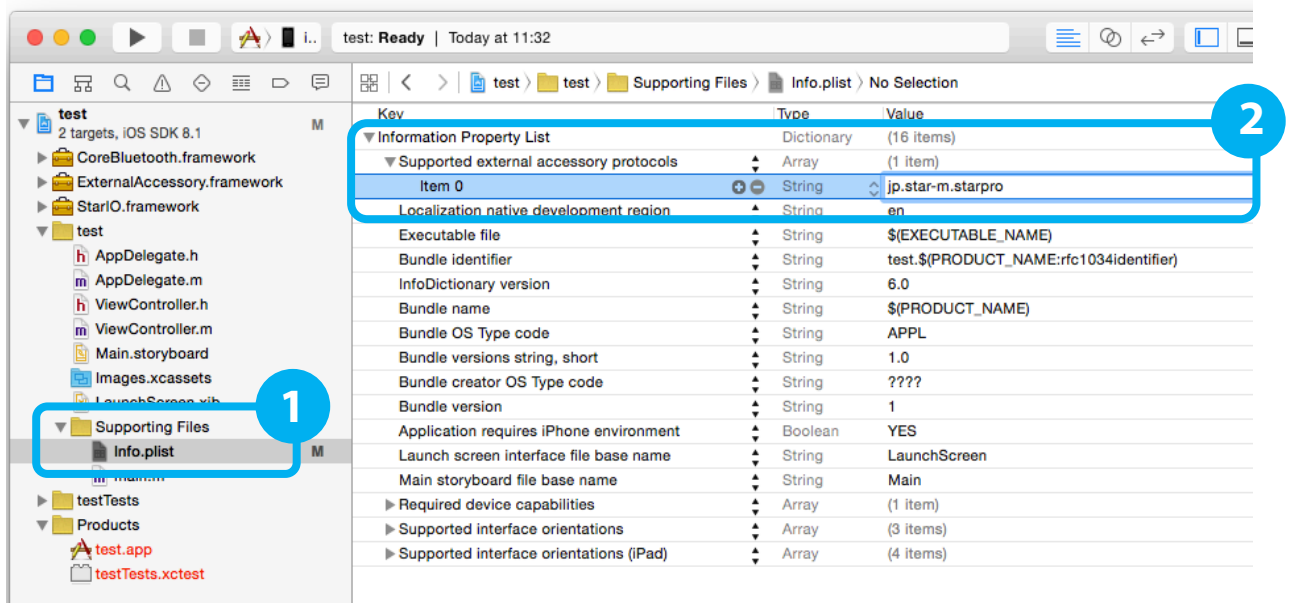
- 必要な framework が追加されているか確認してください。



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")を選択します。
2. Information Property List に項目を追加して、[Key]に"Supported external accessory protocols"を設定（入力）、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、設定します。
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、設定します。（単位：ミリ秒）

@property(assign, readwrite) u_int32_t endCheckedBlockTimeoutMillis

endCheckedBlock メソッドのタイムアウト時間を取得、設定します。[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。

指定した時間が 10 秒未満の場合、タイムアウトは 10 秒になります。



また、getPort メソッドの portSettings パラメータで[データタイムアウト機能の設定]を使用する場合、このメソッドで設定するタイムアウト時間には、データタイムアウト機能の指定時間より 3 秒以上長くなるように設定してください。3 秒未満に設定した場合、内部制御により自動的に 3 秒長くなるように設定されます。

●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 シリーズのプリンター共有機能を使用する場合

portName には、AirMac シリーズの IP アドレスを指定してください。

例) @"TCP:192.168.1.2"

portSettings - 空文字列(@"")を指定

 - データタイムアウト機能*を有効とする場合 @"d[数値]"を指定

*データタイムアウト機能 (Bluetooth のみ対応)

プリンターが印刷中にエラーになった場合のほか、指定時間、プリンターに対してデータが送られなかった場合にもデータキャンセル機能を動作させます。

指定時間は[数値]に 0～255（単位：秒）を設定します。デフォルトでは 3 秒となっています。範囲外の値を設定した場合、その値は無視されます。

この機能により、データ送信中に Bluetooth 接続が切断された場合に、次回の印刷が不正な内容になることを防ぐことができます。



データタイムアウト機能は、TSP650II、TSP800II で F/W Ver 2.0 以降、TSP700II では F/W Ver. 5.0 以降に対応しています。

例) @"d127" (印刷中に 127 秒間データが送られなかった場合に、データキャンセル機能を動作する場合)

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 シリーズ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”



portSettings パラメータで[データタイムアウト機能の設定]を有効とする場合、データタイムアウト機能の指定時間より 3 秒以上長くなるように設定してください。3 秒未満に設定した場合、内部制御により endCheckedBlock メソッドのタイムアウト時間は自動的に 3 秒長くなるように設定されます。

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。



Bluetooth インターフェイスの場合、プリンターとの通信を 30 秒以上行わない場合は一度ポートをクローズしてください。

1 トランザクションごとにポートオープン・クローズすることを推奨します。

searchPrinter

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

searchPrinter は、LAN 上のプリンターとペアリングされた Bluetooth プリンターを検索し、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、プリンターの MAC アドレス、プリンターモデル名を持ち、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、Ethernet もしくは Bluetooth プリンター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 :	Ethernet プリンターを検索
	@”BT:”を指定した場合 :	Bluetooth プリンターを検索



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、ネットワーク上のプリンターと Bluetooth プリンターの両方を検索して一覧を取得し、その内容をログに出力します。

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。プリンターから 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。コマンドや印刷データの送信に使用します。印字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、プリンター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プリンターから取得します。

戻り値:

[StarPrinterStatus](#) 構造体は、現在のデバイス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
blackMarkError	ブラックマークエラー	SM_BOOLEAN	ブラックマークエラー(ブラックマーク設定時に非ブラックマーク用紙を使って印刷した場合等に発生)の時に SM_TRUE となる。通常時は SM_FALSE。
compulsionSwitch	コンパルジョン SW	SM_BOOLEAN	ドロワのコンパルジョン 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	カバーが開いている場合に SM_TRUE となる。閉じている場合は SM_FALSE。
cutterError	オートカッターエラー	SM_BOOLEAN	カッターエラー発生時に SM_TRUE となる。通常は SM_FALSE。
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	ヘッドサーミスタ異常値検出時に SM_TRUE となる。通常は SM_FALSE。
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	ヘッドが高温になり印刷停止している状態で SM_TRUE となる。通常は SM_FALSE。
presenterPaperJamError	プレゼンター紙ジャムエラー	SM_BOOLEAN	プレゼンター装着時、プレゼンターで用紙ジャムが発生すると SM_TRUE となる。通常は SM_FALSE。
presenterState	プレゼンタ用紙位置	UCHAR	以下のいずれかの数値。 0: プレゼンタ内部に用紙がない 1: 用紙を給紙した状態(ループ状態) 3: 用紙を排出した状態 6: 用紙回収状態 7: 用紙が引き抜かれた状態
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 00)
rawLength	raw の長さ	CHAR	raw の長さ
receiptPaperEmpty	用紙エンド	SM_BOOLEAN	用紙切れの場合は SM_TRUE となる。通常は SM_FALSE。
receiptPaperNearEmptyInner	用紙ニアエンド(内側)	SM_BOOLEAN	用紙ニアエンド状態の時に SM_TRUE となる。通常は SM_FALSE。
receiveBufferOverflow	受信バッファオーバーフロー	SM_BOOLEAN	受信バッファフルの時に SM_TRUE となる。通常は SM_FALSE。
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	電源電圧異常値検出時に SM_TRUE となる。通常は SM_FALSE。

StarPrinterStatus 構造体 機種別対応リスト

メンバ名	TSP 100 LAN	TSP 100 U	TSP 100 GT	TSP 100 IIU	FVP 10	TSP 650	TSP 650 II	TSP 700 II	TSP 800 II	TUP 500	SP 700
blackMarkError					✓			✓	✓	✓	✓
compulsionSwitch	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
coverOpen	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
cutterError	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
etbAvailable	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
etbCounter	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
headThermistorError										✓	
offline	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
overTemp	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
presenterPaperJamError										✓	
presenterState										✓	
raw	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
rawLength	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
receiptPaperEmpty	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
receiptPaperNearEmptyInner					✓	✓	✓	✓	✓	✓	✓
receiveBufferOverflow					✓	✓	✓	✓	✓	✓	✓
unrecoverableError	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
voltageError										✓	

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印字終了の確認を行います。
印刷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。



TSP650、TUP500 では F/W 3.0 以降が必要です。

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います。

プリンタの状態を監視し、送信した印刷データの印刷が完了すると制御を返します。印刷データ以外を送信した場合は、そのコマンドがプリンタに処理されると制御を返します。

タイムアウト時間(*1) 内に印刷が完了しなかった場合や、印刷中にプリンタ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プリンタからの終了の応答がない



TSP650、TUP500 では F/W 3.0 以降が必要です。

```

unsigned char command[] = {0x41, 0x42, 0x43, 0x44, 0x1B, 0x7A, 0x00, 0x1B, 0x64, 0x02};
uint bytesWritten = 0;

StarPrinterStatus_2 starPrinterStatus;
SMPort *port = nil;

@try
{
    port = [SMPort getPort:@"BT:" :@"":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
- プリンターが切断機能に対応していない場合(モバイルプリンター等)

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。

Ethernet 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。

Key に @modelName を設定することで戻り値からモデル名を取得します。

Key に @firmwareVersion を設定することで戻り値からファームウェアバージョンを取得します。

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

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。

設定可能文字数 : 1~16

使用可能文字列 : 0-9, a-z, A-Z,

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

@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。

設定可能文字数 : 1~16

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。

設定可能文字数 : 4~16

使用可能文字列 : 0-9, a-z, A-Z

●メソッド

initWithPortName : deviceType

```
-(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。

open メソッドの実行後は、必ず loadSetting メソッドで現在の設定を取得してください。

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。

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。

apply

- (BOOL) apply

deviceName, iOSSPortName, 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。

StarIO iOS SDK 機能

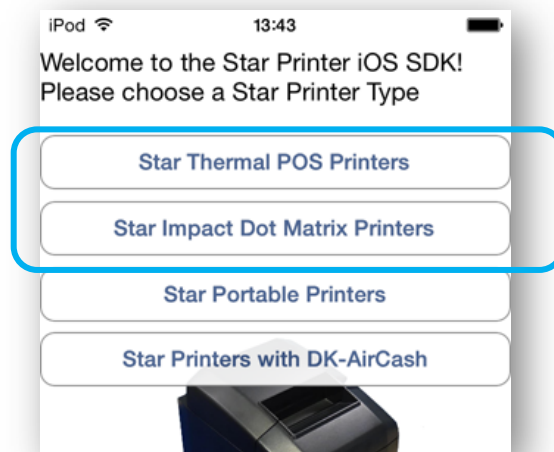
SDK 機能と StarIO のプリンターコマンドについての概要

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。

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。

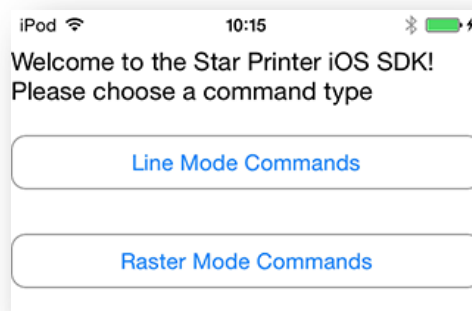
プリンターとコマンド種類の選択

- 1) サーマルプリンターを使用する場合は“Star Thermal POS Printers”を、ドットプリンターを使用する場合は“Star Impact Dot Matrix Printers”をタップします。



2. “Line Mode Commands”、“Raster Mode Commands”を選択してください。ドットプリンターを使用する場合は対応する Line Mode Commands の画面に移ります。

“Line Mode”、“Raster Mode”の詳細については[こちら](#)を参照してください。



コマンド別サンプル機能対応リスト

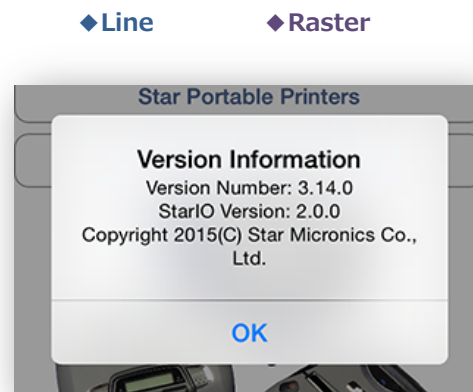
Line Mode Command Samples Include:

- Port Discovery
- Get Firmware Information
- Get Status
- Sample Receipt
- Sample Receipt (UTF-8)
- Open Cash Drawer
- 1D Barcodes
- 2D Barcodes
- Cut
- Text Formatting
- Text Formatting (UTF-8)
- Bluetooth Pairing + Connect
- Bluetooth Disconnect
- Bluetooth Setting

Raster Mode Command Samples Include:

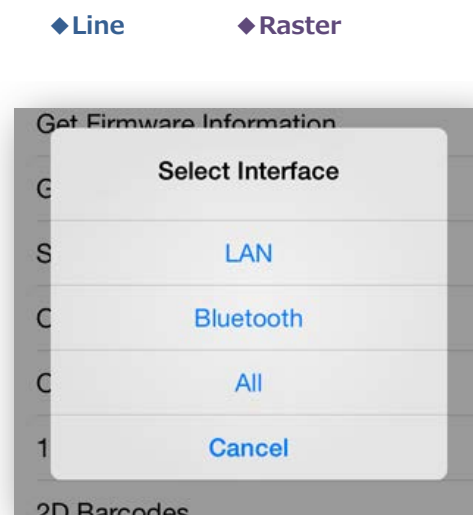
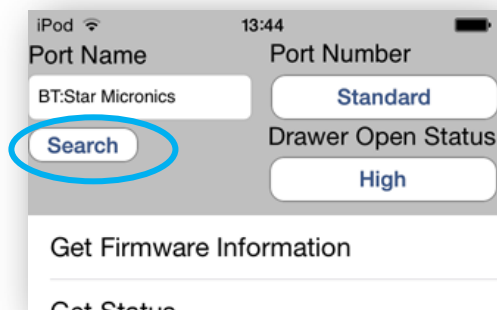
- Port Discovery
- Get Firmware Information
- Get Status
- Sample Receipt
- Open Cash Drawer
- Raster Graphics Text Printing
- ImageFile Printing
- AllReceipts
- Bluetooth Pairing + Connect
- Bluetooth Disconnect
- Bluetooth Setting

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 Star のプリンターを自動的に検索します。検索結果より、接続したいプリンターを選択してください。[詳細はこちらをご参照ください。](#)
※USB プリンターはサポートしておりません。

Help

◆Line

◆Raster

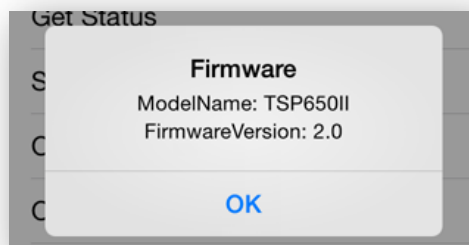


Help は、StarIO ポートの設定に関する情報を表示します。

Get Firmware Information

◆Line

◆Raster



Port Name に設定されたプリンターのファームウェア情報を表示します。

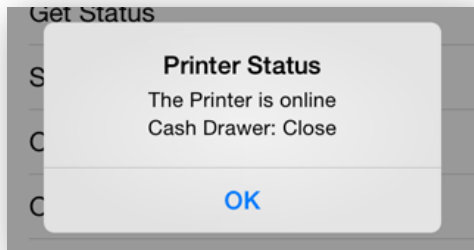
TSP100LAN, TSP100IIIW, TSP100IIILAN, TSP100U, TSP100GT, TSP100ECO では FirmwareVersion を取得できません。

TSP100LAN, TSP100IIIW, TSP100IIILAN では ModelName に“TSP100LAN”と表示します。

TSP100U, TSP100GT, TSP100ECO では ModelName に“TSP100”と表示します。

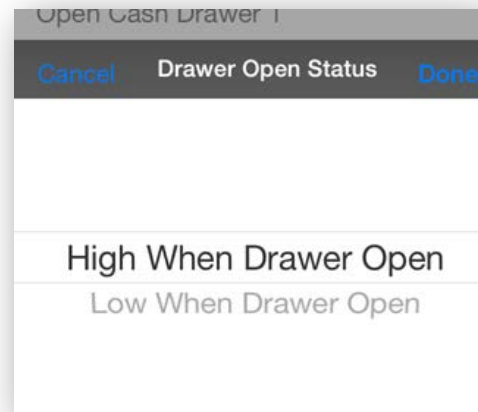
Apple Air Mac シリーズを使用する環境では、ModelName、FirmwareVersion とともに表示されません。

Get Status



◆Line

◆Raster



Drawer Open Status

使用する外部機器の仕様に合わせて設定してください。

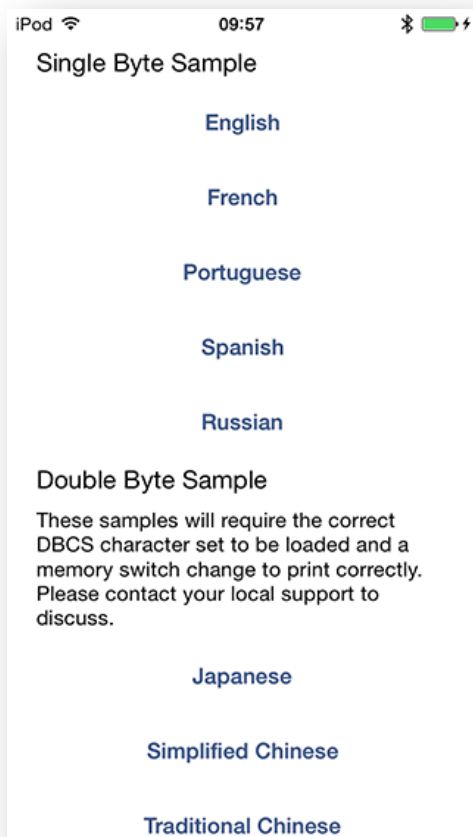
StarPrinterStatus

SM_BOOLEAN getParsedStatus::	ステータスの戻り値はここを参照してください
offline	SM_FALSE = プリンターオンライン
	SM_TRUE = プリンターオフライン
other	ステータスの戻り値はここを参照してください

Sample Receipt

◆Line

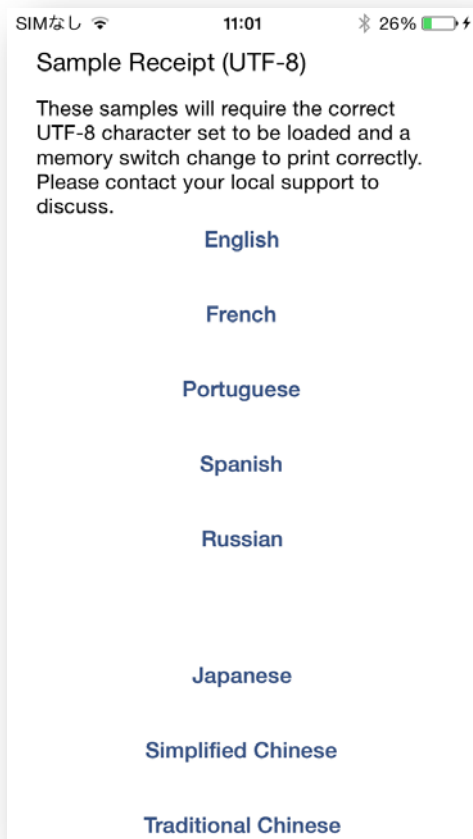
◆Raster



選択したコマンド種類で、選択した言語でのサンプルレシートの印刷をします。
レシート幅をタップして印刷を開始してください。
ソースコードには、レシートをカスタマイズする方法が詳しく説明されています。

Sample Receipt(UTF-8)

◆Line



選択したコマンド種類で、選択した言語でのサンプルレシートの印刷をします。
レシート幅をタップして印刷を開始してください。
ソースコードには、レシートをカスタマイズする方法が詳しく説明されています。



Sample Receipt (UTF-8) は、TSP650II の F/W Ver. 3.0 以降で使用できます。
また、上記以外の機種では、以下の言語のみ、以下のモデル、ファームウェアバージョンで対応しています。

Japanese ^{*1} (JP モデルのみ利用可能) Simplified Chinese ^{*1} (CH/欧米モデル利用可能) Traditional Chinese ^{*1} (TW モデルのみ利用可能)	TSP650II F/W Ver. 1.2 以降 TSP700II F/W Ver. 4.1 以降
Japanese ^{*1} (JP モデルのみ利用可能) Simplified Chinese ^{*1} (CH/欧米モデル利用可能)	TSP800II F/W Ver. 1.6 以降 FVP10 F/W Ver. 1.5 以降

^{*1}TSP650II F/W 3.0 以降以外で利用するには、プリンターのメモリスイッチを変更する必要があります。
Bank0 の bit4 を有効に設定してから利用してください。
メモリスイッチの変更は、StarPRNT Intelligence ソフトウェアから行えます。[こちら](#)からダウンロードしてください。

Open Cash Drawer1

◆Line

◆Raster

キャッシュドロフが接続されていれば、オープンします。

BEL

キャッシュドロフをオープンします。(値の指定は不要)

Open Cash Drawer2

◆Line

◆Raster

キャッシュドロフが接続されていれば、オープンします。

SUB

キャッシュドロフをオープンします。(値の指定は不要)



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

1D Barcodes

◆Line

<<CODE39>>



ESC b n1 n2 n3 n4 d1 ... dk RS

n1 = バーコード種選択

0 = UPC-E *1 1 = UPC-A *1 2 = JAN/EAN8 *1 3 = JAN/EAN13 *1
4 = Code39 5 = ITF 6 = Code128 7 = Code93 8 = NW-7 *1

n2 = バー下文字選択、及び、改行付加選択

1 = バー下文字を付加しない。バーコード印字後、改行を実行する。
2 = バー下文字を付加する。バーコード印字後、改行を実行する。
3 = バー下文字を付加しない。バーコード印字後、改行を実行しない。
4 = バー下文字を付加する。バーコード印字後、改行を実行しない。

n3 = バーコードモード選択

n4 = バーコード高さ(ドット数)

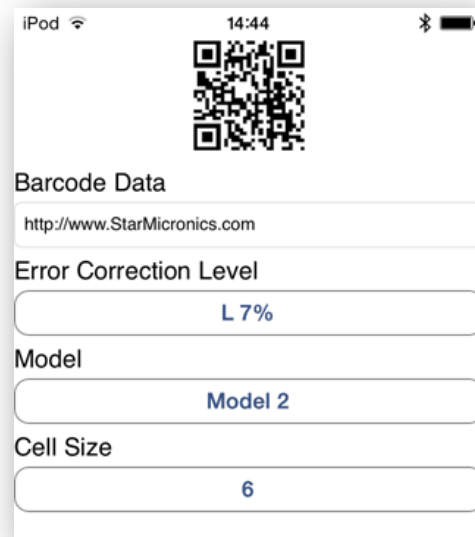
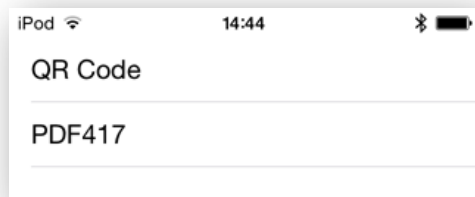
*1: これらのバーコードは、Star POS プリンターではサポートされていますが、SDK では実装されていません。

注記：このサンプルアプリケーションは Raster Mode での 1D Barcodes 印刷に対応していません。Raster Mode ではバーコードを画像データとしてプリンターに送信する必要があります。

2D Barcodes

◆Line

<<QR Code>>



QR Code に関するコマンドは、その機能により以下の 5 つに分類される。

- | | |
|--------------------|------------------------------|
| (1) モデルの指定 | ESC GS y S 0 n |
| (2) エラー訂正レベルの指定 | ESC GS y S 1 n |
| (3) セルサイズの指定 | ESC GS y S 2 n |
| (4) QR Code データの指定 | ESC GS y D 1 NUL nL nH d1…dk |
| (5) QR Code の印刷 | ESC GS y P |

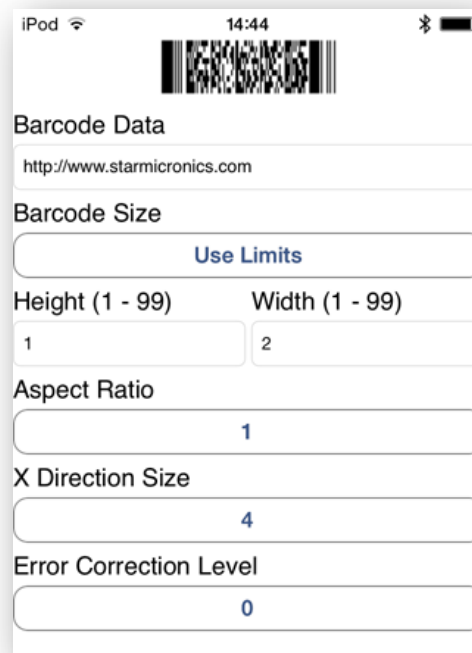
以下の順番でコマンドを送信することにより、QR Code が印刷できます。

モデル指定 + エラー訂正レベル + セルサイズ + QR Code データ + QR Code 印刷

※詳細は、Star ラインモードコマンド仕様書を参照してください。

注記：このサンプルアプリケーションは Raster Mode での 2D Barcodes 印刷に対応していません。Raster Mode ではバーコードを画像データとしてプリンターに送信する必要があります。

<<PDF417>>



iPod 14:44

Barcode Data
http://www.starmicronics.com

Barcode Size
Use Limits

Height (1 - 99) Width (1 - 99)
1 2

Aspect Ratio
1

X Direction Size
4

Error Correction Level
0

PDF417 に関するコマンドは、その機能により以下の 7 つに分類される。

- | | |
|-----------------------|---------------------------|
| (1) サイズの指定 | ESC GS x S 0 n p1 p2 |
| (2) エラー訂正レベルの指定 | ESC GS x S 1 n |
| (3) モジュールサイズ(X 方向)の指定 | ESC GS x S 2 n |
| (4) モジュールサイズ(Y 方向)の指定 | ESC GS x S 3 n |
| (5) PDF417 データの指定 | ESC GS x D nL nH d1 d2…dk |
| (6) PDF417 の印刷 | ESC GS x P |
| (7) PDF417 展開情報取得 | ESC GS x I |

以下の順番でコマンドを送信することにより、PDF417 が印刷できます。

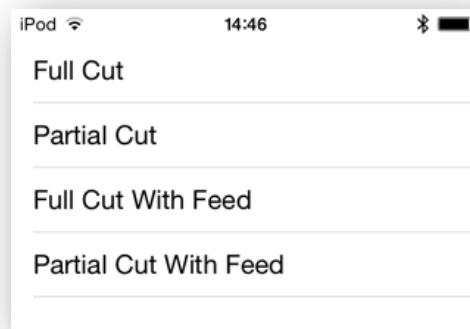
サイズ + エラー訂正レベル + モジュール(X) + モジュール(Y) + PDF417 データ + PDF417 印刷

※詳細は、Star ラインモードコマンド仕様書を参照してください。

注記：このサンプルアプリケーションは Raster Mode での 2D Barcodes 印刷に対応していません。
Raster Mode ではバーコードを画像データとしてプリンターに送信する必要があります。

Cut

◆Line



Full Cut

ESC d 0 現在の位置でフルカット。

Partial Cut

ESC d 1 現在の位置でパーシャル(一点残し)カット。

Full Cut With Feed

ESC d 2 カット位置まで用紙を送って、フルカット。

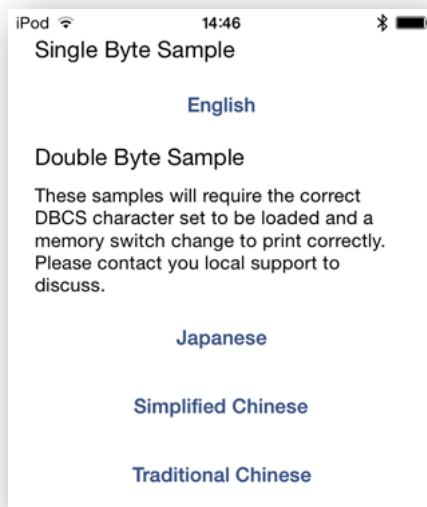
Partial CutWith Feed

ESC d 3 カット位置まで用紙を送って、パーシャル(一点残し)カット。

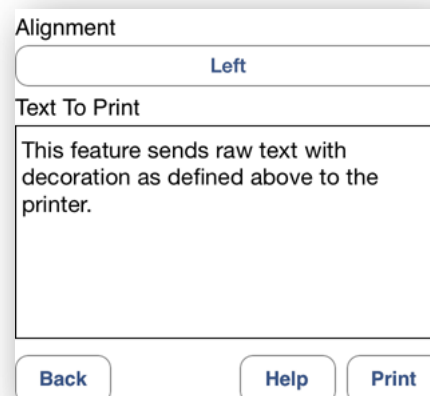
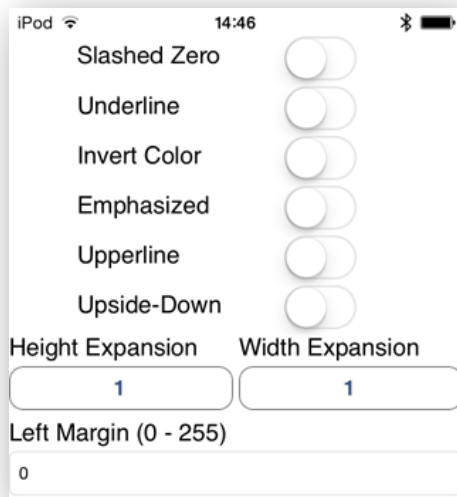
注記：このサンプルアプリケーションは Raster Mode での Cut 機能に対応していません。Raster Mode でのカットコマンドについて詳しくは、Star ラインモードコマンド仕様書を参照してください。

Text Formatting

◆Line



例) サーマルプリンター/English



例) サーマルプリンター/Japanese



文字の書式設定

- ・ **Shift-JIS :** シフト JIS 漢字モードの指定(Japanese のみ)
ESC \$ 1 =シフト JIS 漢字モード設定 ESC \$ 0 =解除[デフォルト]

- ・ **JIS :** JIS 漢字モードの指定 (Japanese のみ)
ESC p =設定 ESC q =解除[デフォルト]

- ・ **Slashed Zero :** スラッシュゼロ (English のみ)
ESC / n 1 = 設定 0=解除[デフォルト]

- ・ **Underline :** アンダーライン
ESC - n 1 = 設定 0=解除[デフォルト]

- ・ **Invert Color :** 白黒反転印字 (サーマルプリンター)
ESC 4 = 設定 ESC 5 = 解除[デフォルト]

- ・ **Two Color :** 2色印字 (ドットプリンター)
ESC 4 = 赤 ESC 5 = 黒[デフォルト]

- ・ **Emphasized :** 強調印字
ESC E = 設定 ESC F = 解除[デフォルト]

- ・ **Upperline :** アッパーライン
ESC _ n 1 = 設定 0=解除[デフォルト]

- ・ **Upside-Down :** 倒立印字
SI = 設定 DC2 = 解除[デフォルト]

- ・ **文字サイズの拡大** (サーマルプリンター)
Height Expansion: 高さ ESC h n 0 ≤ n ≤ 5
Width Expansion: 幅 ESC W n 0 ≤ n ≤ 5

- ・ **文字サイズの拡大** (ドットプリンター)
Height Expansion: 縦拡大 (English のみ) ESC h n 1 = 設定 0 = 解除[デフォルト]
Width Expansion: 横拡大 ESC W n 1 = 設定 0 = 解除[デフォルト]

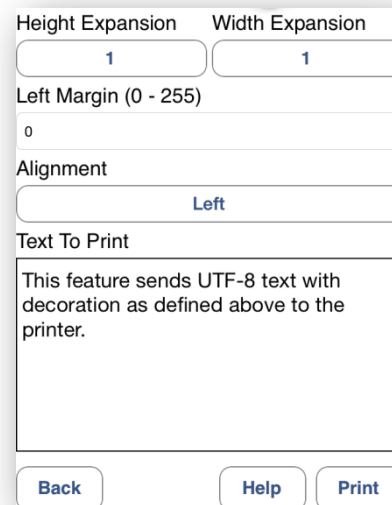
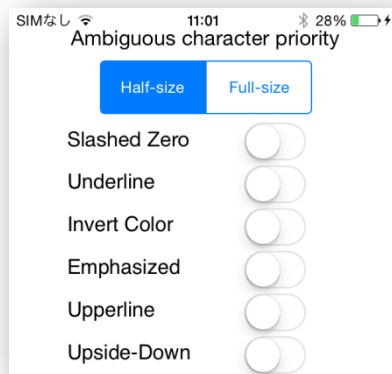
- ・ **Left Margin (0 – 255):** 左マージンの設定
GS l n 0 ≤ n ≤ 255

- ・ **Alignment:** 位置揃え
ESC GS a 0 = 左揃え[デフォルト]
ESC GS a 1 = 中央寄せ
ESC GS a 2 = 右揃え

注記：Raster Mode を使用されている場合は、[Raster Graphical Text Printing](#) をご参照ください。

Text Formatting(UTF-8)

◆Line



文字の書式設定

• Ambiguous character priority: Unicode あいまい文字の設定

ESC GS) U pL pH fn 0 = Half-size: 半角優先 [デフォルト]

ESC GS) U pL pH fn 1 = Full-size: 全角優先

• Slashed Zero : スラッシュゼロ (English のみ)

ESC / n

1 = 設定

0 = 解除 [デフォルト]

• Underline : アンダーライン

ESC _ n

1 = 設定

0 = 解除 [デフォルト]

• Invert Color : 白黒反転印字 (サーマルプリンター)

ESC 4 = 設定

ESC 5 = 解除 [デフォルト]

• Emphasized : 強調印字

ESC E = 設定

ESC F = 解除 [デフォルト]

• Upperline : アッパーライン

ESC _ n

1 = 設定

0 = 解除 [デフォルト]

• Upside-Down : 倒立印字

SI = 設定

DC2 = 解除 [デフォルト]

• 文字サイズの拡大

Height Expansion: 高さ

ESC h n 0 ≤ n ≤ 5

Width Expansion: 幅

ESC W n 0 ≤ n ≤ 5

• Left Margin (0 - 255): 左マージンの設定

GS l n

0 ≤ n ≤ 255

- **Alignment:** **位置揃え**
ESC GS a 0 = 左揃え[デフォルト]
ESC GS a 1 = 中央寄せ
ESC GS a 2 = 右揃え

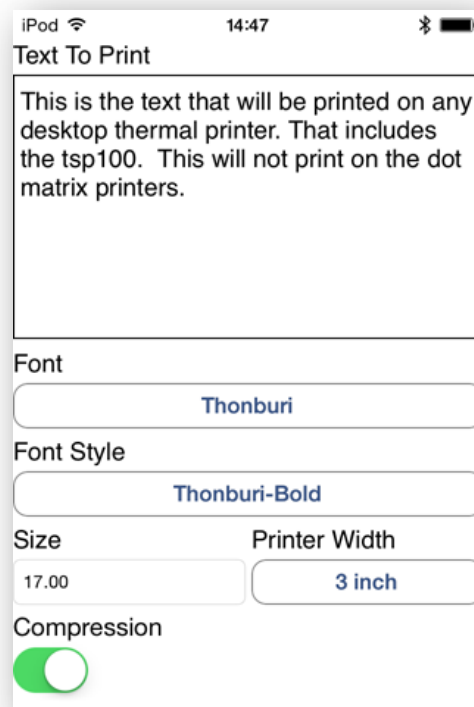


Text Formatting (UTF-8) は、TSP650II の F/W Ver. 3.0 以降で使用
できます。

Raster Graphical Text Printing

◆Raster

◆Raster



Raster Mode では、すべての印刷データをイメージデータに変換してプリンターへ送信します。これにより、文字データおよびロゴ/クーポンの付加された印刷を高速で行うことができます。

Raster コマンドの詳細については、「Star ラインモードコマンド仕様書」をご参照ください。また、画面右下に表示される[Help]をタップすることにより、Raster コマンドについて確認できます。

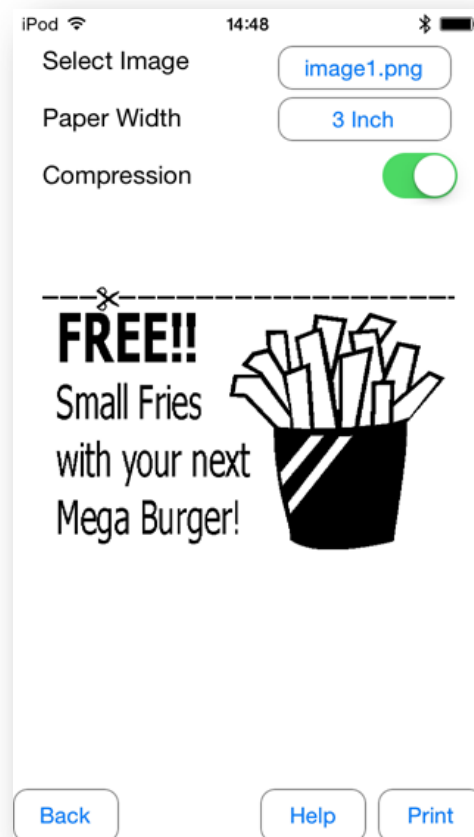
注記：

- ・ Line Mode を使用されている場合は、[Text Formatting](#) をご参照ください。

Image File Printing

◆Raster

◆Raster



イメージデータの Raster 印刷に使用する画像を Select Image から選択してください。

Raster Mode では、すべての印刷データをイメージデータに変換してプリンターへ送信します。これにより、文字データおよびロゴ/クーポンの付加された印刷を高速で行うことができます。

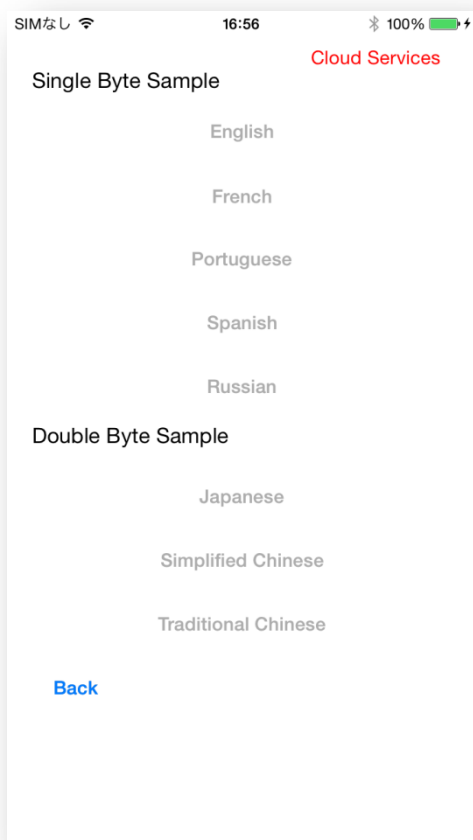
“Compression”を使用することにより、スループットの向上が望めます。

Raster コマンドの詳細については、「Star Line Mode コマンド仕様書」をご参照ください。
また、画面右下に表示される[Help]をタップすることにより、Raster コマンドについて確認できます。

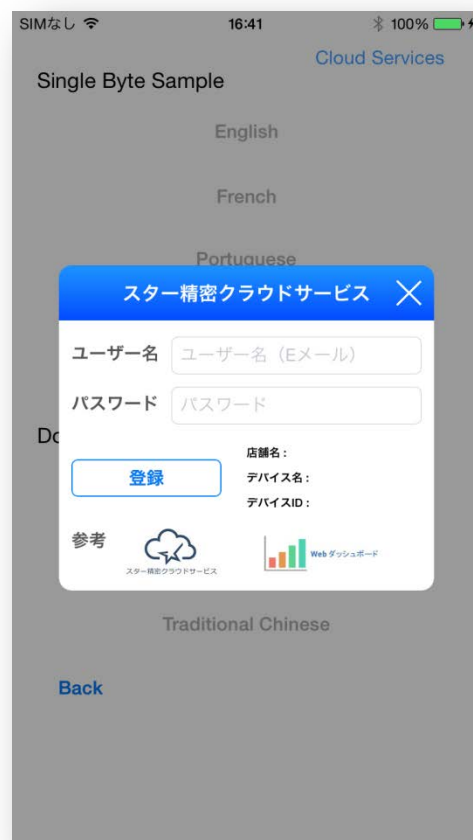
注記： このイメージデータは 80mm幅のレシートに合わせて作成されています。
80mm幅以外の用紙を使用されている場合、自動的にリサイズされません。

AllReceipts

◆Raster



◆Raster



Cloud Services をタップすると、スター精密クラウドサービスへのデバイス登録を行うための Registration View を表示します。

AllReceipts、スター精密クラウドサービスについて詳しくは、Star_AllReceipts_iOS_SDK マニュアルをご参照ください。

Bluetooth Pairing + Connect

◆Line

◆Raster

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。

Bluetooth Disconnect

◆Line

◆Raster

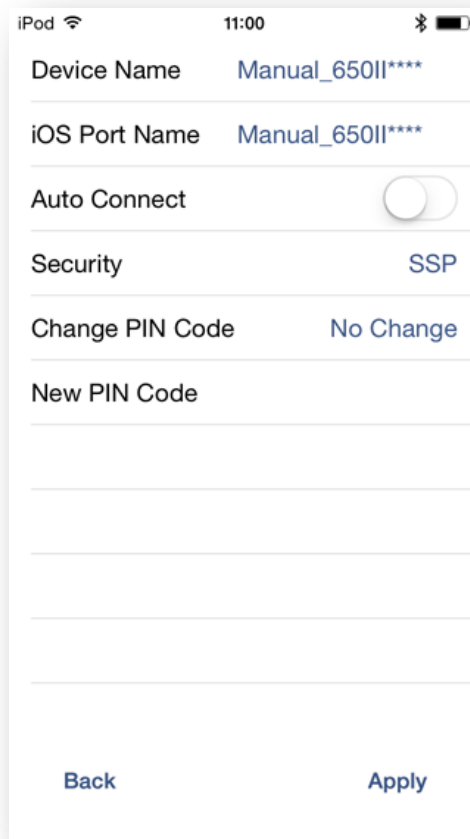
タップするとペアリング+コネクションが行われた Star 製プリンターの一覧が表示されます。選択すると、そのデバイスとのコネクションが切断され、他のホストからそのデバイスに対して接続できるようになります。

この機能は“disconnect”メソッドを使用しています。

Bluetooth Setting

◆Line

◆Raster



iPod 11:00

Device Name Manual_650II****

iOS Port Name Manual_650II****

Auto Connect ☐

Security SSP

Change PIN Code No Change

New PIN Code

Back Apply

PortName に指定された Bluetooth デバイスに接続して、Bluetooth インターフェイス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、プリンターの 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プリンター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、各プリンター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
- スター精密プリンタードライバ
- 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

改訂年月	SDK パッケージ バージョン	更新内容
Mar. 2016	3.16.0	・ AllReceipts 対応追加
Jan. 2016	3.15.a	・ 対応モデル追加 (TSP100IIIW、TSP100IIILAN)
Jul. 2015	3.14.1	・ Sample Receipt/Text Formatting UTF-8 対応
Apr. 2015	3.14.0	・ iOS 5. x サポート終了
Mar. 2015	3.13.1	・ 32 ビット 個別のビルド設定を削除
Sep. 2014	3.12.0	・ ドットプリンター対応追加 (SP700) ・ StarIO Version メソッド追加 ・ Image File Printing 追加
Jul. 2014	3.10.3	・ データタイムアウト機能追加 ・ Bluetooth 接続における多重 getPort 実行に関する問題を修正
Mar. 2014	3.10.0	・ getFirmwareInformation メソッド追加 ・ Open Cash Drawer2 機能追加
Nov. 2013	3.9.0	・ SMBluetoothManager クラス追加 ・ endCheckedBlockTimeoutMillis プロパティ追加 ・ iPad Air,iPad mini (第 2 世代),iPhone 5S,iPhone 5C 対応 ・ iOS 4.3 サポート終了 ・ 64 ビット ビルド対応
Sep. 2013	3.8.0	・ iOS 7 対応
Jul. 2013	3.7.1	・ Star Printer Status 構造体ステータスリスト追加
Jun. 2013	3.6.0	・ SearchPrinter 機能追加
Mar. 2013	3.4.0a	・ サンプルコードのバグを修正
Jan. 2013	3.3.0	・ Bluetooth I/F 対応
Nov. 2012	3.2.0	・ StarIO SDK 対応リスト更新
Oct. 2012	3.1.0	・ iOS 6 対応 ・ iPhone 5 対応
Aug. 2012	3.0.0	・ SDK UI 変更 ・ Port Discovery 対応 (POS Printer) ・ Apple AirPort Express 対応 ・ サンプルレシート印刷対応
May. 2012	2.3.0	・ モバイルプリンターにおけるスリープモード使用時の不具合修正 (モバイルプリンター)

改訂年月	SDK パッケージ バージョン	更新内容
Apr. 2012	2.2.0	・ StarIO ポートデフォルトクラス名変更 ・ Retina Display 対応 ・ ARC 対応
Jan. 2012	2.1.0	・ Japanese Text Formatting Sample 機能追加 ・ 日本語マニュアル追加
Dec. 2011	2.0.0	・ Sample Printing 機能更新
Jun. 2011	1.2.1	・ 初期リリース



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/>