EPSON
EXCEED YOUR VISION

Epson ePOS SDK for iOS マイグレーションガイド

M00095001
Rev.B

ご注意

- 本書の内容の一部または全部を無断で転載、複写、複製、改ざんすることは固くお断りします。
- 本書の内容については、予告なしに変更することがあります。最新の情報はお問い合わせください。
- 本書の内容については、万全を期して作成いたしましたが、万一ご不審な点や誤り、記載もれなど、お気づきの点がありましたらご連絡ください。
- 運用した結果の影響については、上項に関わらず責任を負いかねますのでご了承ください。
- 本製品がお客様により不適切に使用されたり、本書の内容に従わずに取り扱われたり、またはエプソンおよびエプソン指定の者以外の第三者により修理・変更されたことなどに起因して生じた損害などにつきましては、責任を負いかねますのでご了承ください。
- エプソン純正品およびエプソン品質認定品以外のオプションまたは消耗品を装着してトラブルが発生した場合には、責任を負いかねますのでご了承ください。

商標について

EPSON および EXCEED YOUR VISION はセイコーエプソン株式会社の登録商標です。
IOS® は、Cisco の米国およびその他の国における商標または登録商標です。
Bluetooth® のワードマークおよびロゴは、Bluetooth SIG, Inc. が所有する登録商標であり、セイコーエプソン株式会社はこれらのマークをライセンスに基づいて使用しています。その他の商標およびトレードネームは、それぞれの所有者に帰属します。
その他の製品名および会社名は、各社の商標または登録商標です。

使用制限

本製品を航空機・列車・船舶・自動車などの運行に直接関わる装置・防災防犯装置・各種安全装置など機能・精度などにおいて高い信頼性・安全性が必要とされる用途に使用される場合は、これらのシステム全体の信頼性および安全維持のためにフェールセーフ設計や冗長設計の措置を講じるなど、システム全体の安全設計にご配慮いただいた上で弊社製品をご使用いただくようお願いいたします。
本製品は、航空宇宙機器、幹線通信機器、原子力制御機器、医療機器など、きわめて高い信頼性・安全性が必要とされる用途への使用を意図しておりませんので、これらの用途には本製品の適合性をお客様において十分ご確認の上、ご判断ください。

もくじ

マイグレーションの概要

本書は、以下の開発ツールを用いて開発したアプリケーションを Epson ePOS SDK for iOS (以降、Epson ePOS SDK) で動作するように修正する方法を説明したマニュアルです。

- ePOS-Print SDK for iOS (以降、ePOS-Print SDK)
- ePOS-Device SDK for iOS (以降、ePOS-Device SDK)

ePOS-Print SDK、ePOS-Device SDK は、今後、新製品対応、新機能対応は行われません。本書を参考にして Epson ePOS SDK に移行 (マイグレーション) してください。

マイグレーションの種類

Epson ePOS SDK へのマイグレーションには 2 種類の方法があります。

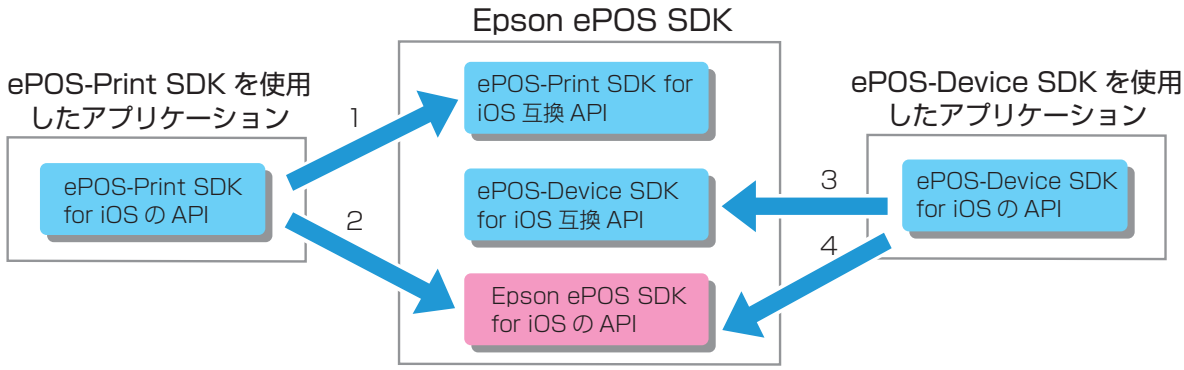
❏ Epson ePOS SDK の互換 API を使用したマイグレーション

Epson ePOS SDK には、ePOS-Print SDK 互換 API と ePOS-Device SDK 互換 API が含まれています。既存のアプリケーションのプログラムは修正せずに、構成ファイルを変更してビルドすることで、Epson ePOS SDK 対応のアプリケーションに移行できます。また、印刷などの基本的な機能を使用する場合は、TM プリンターの新製品にも対応できます。

❏ Epson ePOS SDK の API を使用したマイグレーション

既存のアプリケーションを、Epson ePOS SDK の API を使ったプログラムに修正することで、Epson ePOS SDK 対応のアプリケーションに移行できます。プログラムの修正量は多くなりますが、TM プリンターや周辺機器の新製品、新機能に対応できます。

マイグレーションの種類



1: ePOS-Print SDK から Epson ePOS SDK の ePOS-Print SDK 互換 API にマイグレーション
2: ePOS-Print SDK から Epson ePOS SDK の API にマイグレーション
3: ePOS-Device SDK から Epson ePOS SDK の ePOS-Device SDK 互換 API にマイグレーション
4: ePOS-Device SDK から Epson ePOS SDK の API にマイグレーション

新製品･新機能への対応方針

新製品・新機能	ePOS-Print SDK ePOS-Device SDK	Epson ePOS SDK	
			ePOS-Print SDK 互換 API ePOS-Device SDK 互換 API
エプソン製 TM プリンター、周辺機器	-	○	○
エプソン製 TM プリンター、周辺機器の新機能	-	○	- *
iOS の新バージョン	-	○	○
タブレット端末、スマートフォンの新製品	-	○	○

○ : 対応します　-: 対応しません

* API の新規追加、変更が必要になる新機能には対応しません。

ePOS-Print SDK からのマイグレーション

ePOS-Print SDK を使用したアプリケーションを、Epson ePOS SDK 対応アプリケーションに移行する方法を説明します。

ePOS-Print SDK 互換 API を使用するマイグレーション

既存のアプリケーションのプログラムは修正せずに、構成ファイルを置き換えることで、Epson ePOS SDK 対応のアプリケーションに移行します。

マイグレーションの手順

変更手順は以下のとおりです。

手順		概要
1	SDK の置き換え	ライブラリーファイルの置き換え 「SDK の置き換え」参照
2	アプリケーションのビルド	SDK のファイルを置き換えたアプリケーションのプロジェクトをビルド

以上で ePOS-Print SDK 互換 API を使用したマイグレーションは完了です。

SDK の置き換え

アプリケーションプロジェクトに組まれている以下のファイルを、Epson ePOS SDK のファイルに置き換えます。

種類	ePOS-Print SDK	ePOS-Print SDK 互換 API
ライブラリー	libeposprint.a	libepos2.a

ePOS-Print SDK 互換 API を使用したアプリケーション開発

ePOS-Print SDK 互換 API を使用したアプリケーションを開発･保守する場合に必要な情報は、以下のマニュアルを参照してください。

❏　ePOS-Print SDK 互換 API の仕様
「ePOS-Print SDK for iOS ユーザーズマニュアル」
ePOS-Print SDK 互換 API の仕様は、ePOS-Print SDK API の仕様と同じです。

❏　エプソン製 TM プリンターの新製品の機種情報・サポート API
本書「Epson ePOS SDK for iOS マイグレーションガイド」の付録

Epson ePOS SDK の API を使用するマイグレーション

既存のアプリケーションのプログラムを修正して、Epson ePOS SDK 対応のアプリケーションに移行します。プログラムの修正量は多くなりますが、TM プリンターの新製品、新機能に対応していくことができます。

マイグレーションの手順

変更手順は以下のとおりです。

手順		概要
1	SDK の置き換え	ヘッダーファイルとライブラリーファイルの置き換え 「SDK の置き換え」参照
2	ダイナミックライブラリーを追加	アプリケーションのプロジェクトにダイナミックライブラリーを追加 「ダイナミックライブラリーの追加」参照
3	インポート定義の変更	Objective-C ヘッダーのインポート定義を変更 「インポート定義の変更」参照
4	クラスの変更	ePOS-Print SDK のクラスを Epson ePOS SDK のクラスに変更 「クラスの変更」参照
5	API の変更	Epson ePOS SDK と ePOS-Print SDK で仕様の異なる API の変更やプログラムを修正 変更する内容は、以下のとおりです。 ❏ 特定の機能を実現するためにプログラムを修正 以下の機能を修正してください。 • プリンターとの通信の接続と切断 「プリンターとの通信の接続と切断」参照 • 印刷 「印刷」参照 • コールバックの取得 「コールバックの取得」参照 • ステータスの取得 「ステータスの取得」参照 • プリンターの検索 「プリンターの検索」参照 • ステータスの監視 「ステータスの監視」参照 ❏ API の名称変更 APIの名称を変更するもの（パラメーターの変更が必要な場合もあります） 「API の名称変更」参照 ❏ API のパラメーター変更 API の名称は変わらないが、パラメーターの変更が必要なもの 「API のパラメーター変更」参照
6	アプリケーションのビルド	修正したアプリケーションのプロジェクトをビルド

以上で Epson ePOS SDK の API を使用したマイグレーションは完了です。

SDK の置き換え

アプリケーションプロジェクトに組まれている以下のファイルを、Epson ePOS SDK のファイルに置き換えます。

種類	ePOS-Print SDK	Epson ePOS SDK
ヘッダーファイル	ePOS-Print.h	ePOS2.h
	ePOSEasySelect.h	ePOSEasySelect.h *
ライブラリー	libeposprint.a	libepos2.a
	libeposeasyselect.a	libeposeasyselect.a *

* ファイル名は変わりません。Epson ePOS SDK のパッケージに含まれるファイルを使用してください。

ダイナミックライブラリーの追加

アプリケーションプロジェクトに、以下のダイナミックライブラリーファイルを組み込みます。

- libxml2.2.*

インポート定義の変更

アプリケーションの *.m ソースファイルに含められている、Objective-C ヘッダーのインポート定義を変更します。

ePOS-Print SDK	Epson ePOS SDK
`#import "ePOS-Print.h"`	`#import "ePOS2.h"`

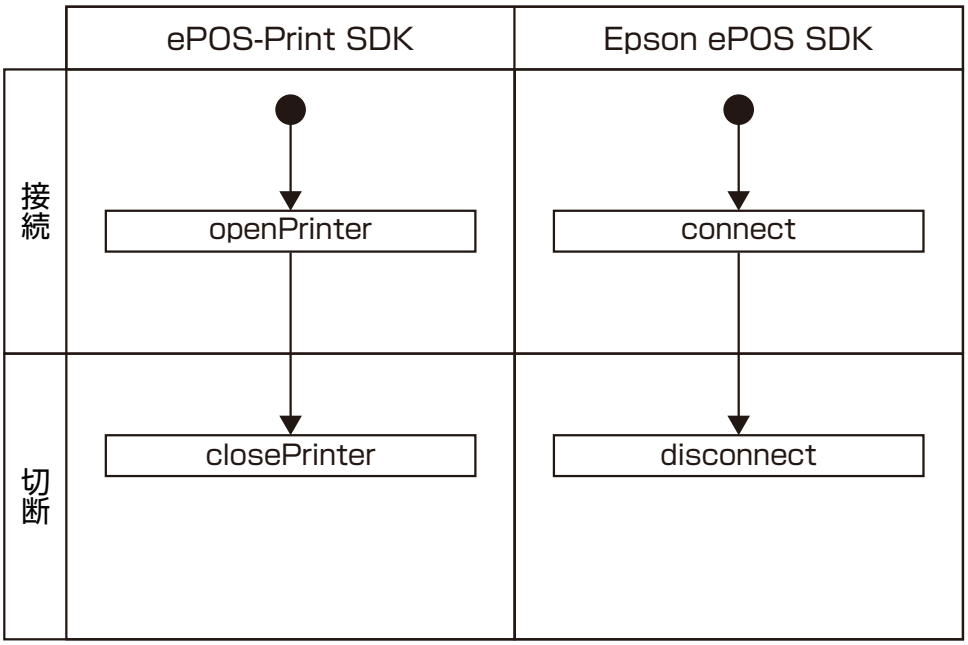
クラスの変更

アプリケーションのプロジェクトで使用している ePOS-Print SDK のクラスを、Epson ePOS SDK のクラスに変更します。

種類	ePOS-Print SDK	Epson ePOS SDK
印刷機能	EposBuilder クラス	Epos2Printer クラス
	EposPrint クラス	
プリンター検索	EpsonIoFinder クラス	Epos2Discovery クラス
ログの出力機能	EposLog クラス	Epos2Log クラス
Bluetooth® 接続	EposBluetoothConnection クラス	Epos2BluetoothConnection クラス

プリンターとの通信の接続と切断

実行手順の違い



プログラムの違い

❏　ePOS-Print SDK

```
id printer = [[EposPrint alloc] init];
if ( printer != nil) {
   int errorStatus = EPOS_OC_SUCCESS;
   errorStatus = [printer openPrinter:EPOS_OC_DEVTYPE_TCP
      Name:@”192.168.192.168” Enabled: EPOS_OC_TRUE
      Interval:EPOS_OC_PARAM_DEFAULT Timeout:EPOS_OC_PARAM_DEFAULT];
   ・・・処理・・・

   errorStatus = [printer closePrinter];
}
```

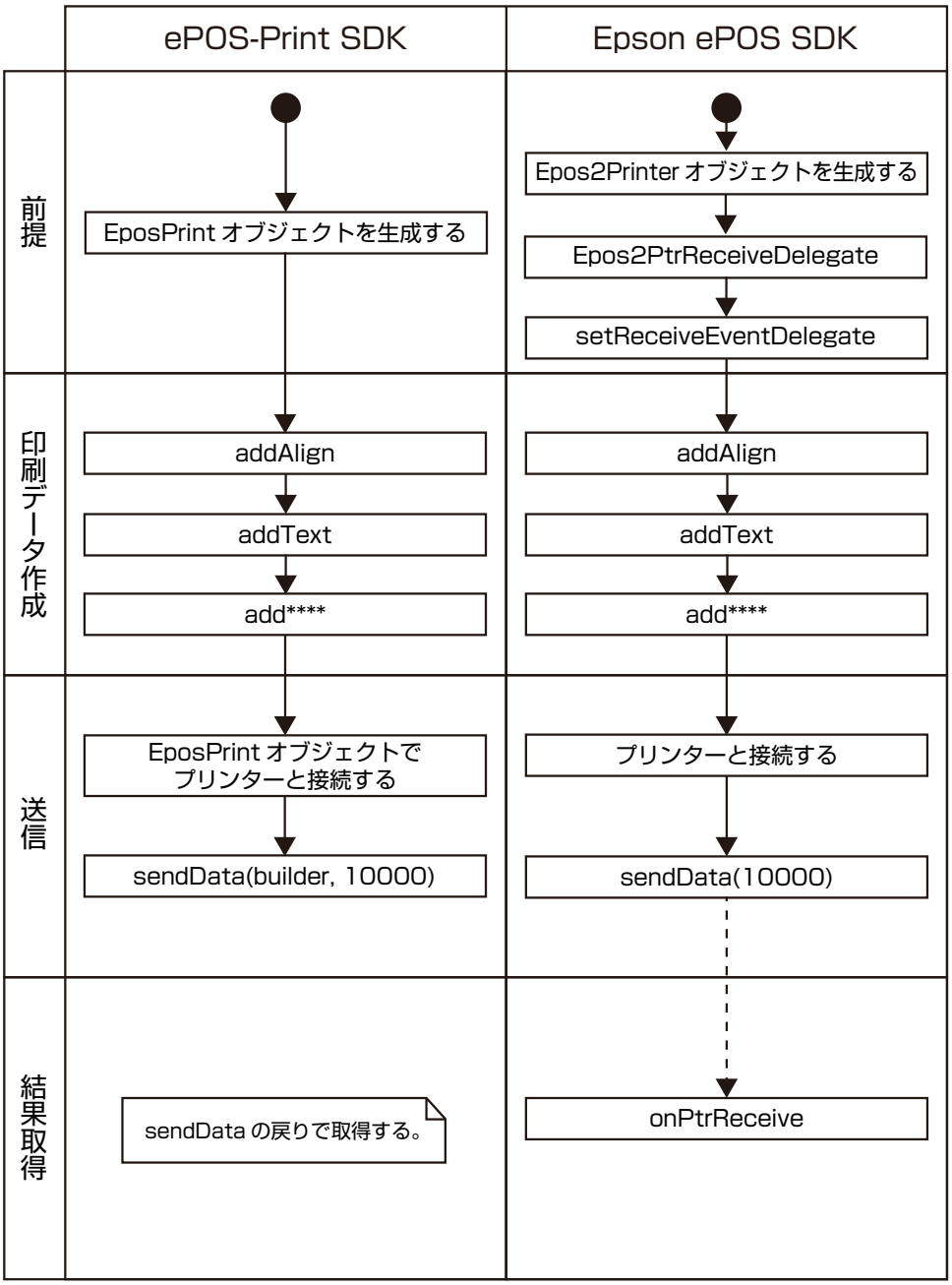
❏　Epson ePOS SDK

```
Epos2Printer printer = [[Epos2Printer alloc] initWithPrinterSeries:EPOS2_TM_T88
                        lang:EPOS2_MODEL_ANK];
if ( printer != nil) {
    int errorStatus = EPOS2_SUCCESS;
    errorStatus = [printer connect:@"TCP:192.168.192.168" timeout:EPOS2_PARAM_DEFAULT];
    ・・・処理・・・
    errorStatus = [printer disconnect];
}
```

印刷

ePOS-Print SDK では、印刷データの送信処理の戻り値で印刷結果を取得していましたが、Epson ePOS SDK では、コールバックで印刷結果を取得します。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Print SDK

```
id builder = [[EposBuilder alloc] initWithPrinterModel: @"TM-T88V" Lang:
EPOS_OC_MODEL_ANK];
if ( builder != nil ) {
   int errorStatus = EPOS_OC_SUCCESS;
   unsigned long status = 0;
   unsigned long battery = 0;
   errorStatus = [builder addText:@"ABCDE"];
   id printer = [[EposPrint alloc] init];
   if ( printer != nil ) {
      errorStatus = [printer openPrinter:EPOS_OC_DEVTYPE_TCP
         DeviceName:@"192.168.192.168"];
      errorStatus = [printer sendData:builder Timeout:10000
         Status:&status Battery:&battery];
      errorStatus = [printer closePrinter];
   }
}
```

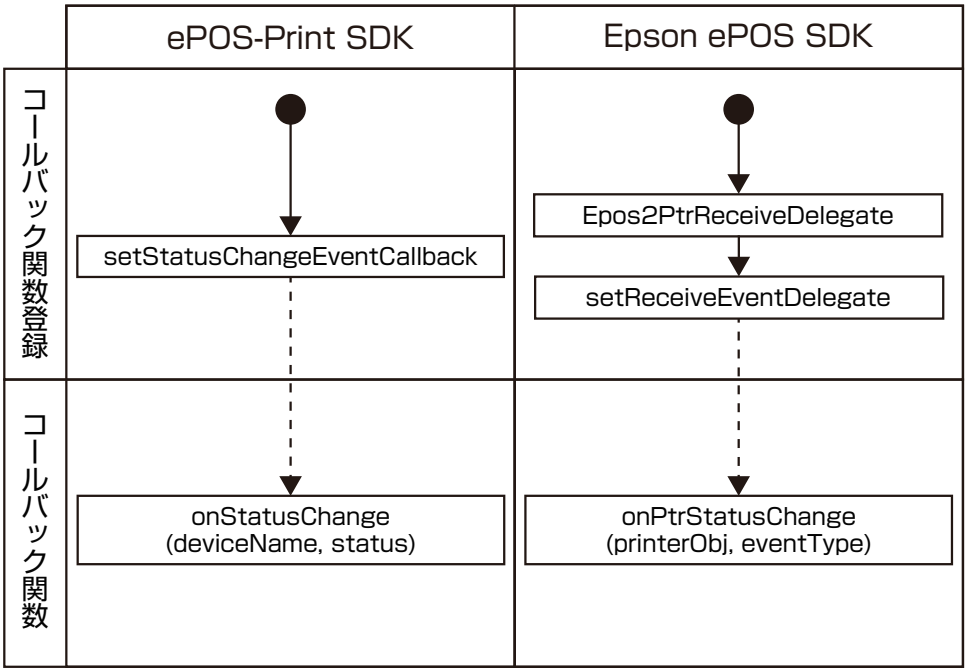
❏ Epson ePOS SDK

```
@interface Sample() <Epos2PtrReceiveDelegate>
{
   Epos2Printer *printer_;
}

- (void) openPrinter
{
    [printer_ setReceiveEventDelegate:self];
    int errorStatus = [printer_ addText:@"ABCDE"];
    ・・・接続・・・
    errorStatus = [printer_ sendData:EPOS2_PARAM_DEFAULT];
}


- (void) onPtrReceive:(Epos2Printer *)printerObj code:(int)code
status:(Epos2PrinterStatusInfo *)status printJobId:(NSString *)printJobId
{
    ・・・処理・・・
}
```

コールバックの取得

ePOS-Print SDK ではセレクターを使ったコールバック処理でしたが、Epson ePOS SDK ではプロトコルを使ったコールバック処理を行います。

実行手順の違い



コールバック : - - - - ▶

プログラムの違い

❏ ePOS-Print SDK

```
- (void)onStatusChange:(NSString *)deviceName Status:(NSNumber *)status
{
・・・接続・・・
}
- (void)openPrinter
{
   id printer = [[EposPrint alloc] init];
   if ( printer != nil) {
      int errorStatus = EPOS_OC_SUCCESS;
      [printer setStatusChangeEventCallback @selector(onStatusChange:Status:)
Target:self];
      errorStatus = [printer openPrinter:EPOS_OC_DEVTYPE_TCP
         Name:@"192.168.192.168" Enabled: EPOS_OC_TRUE
         Interval:EPOS_OC_PARAM_DEFAULT Timeout:EPOS_OC_PARAM_DEFAULT];
   }
}
```

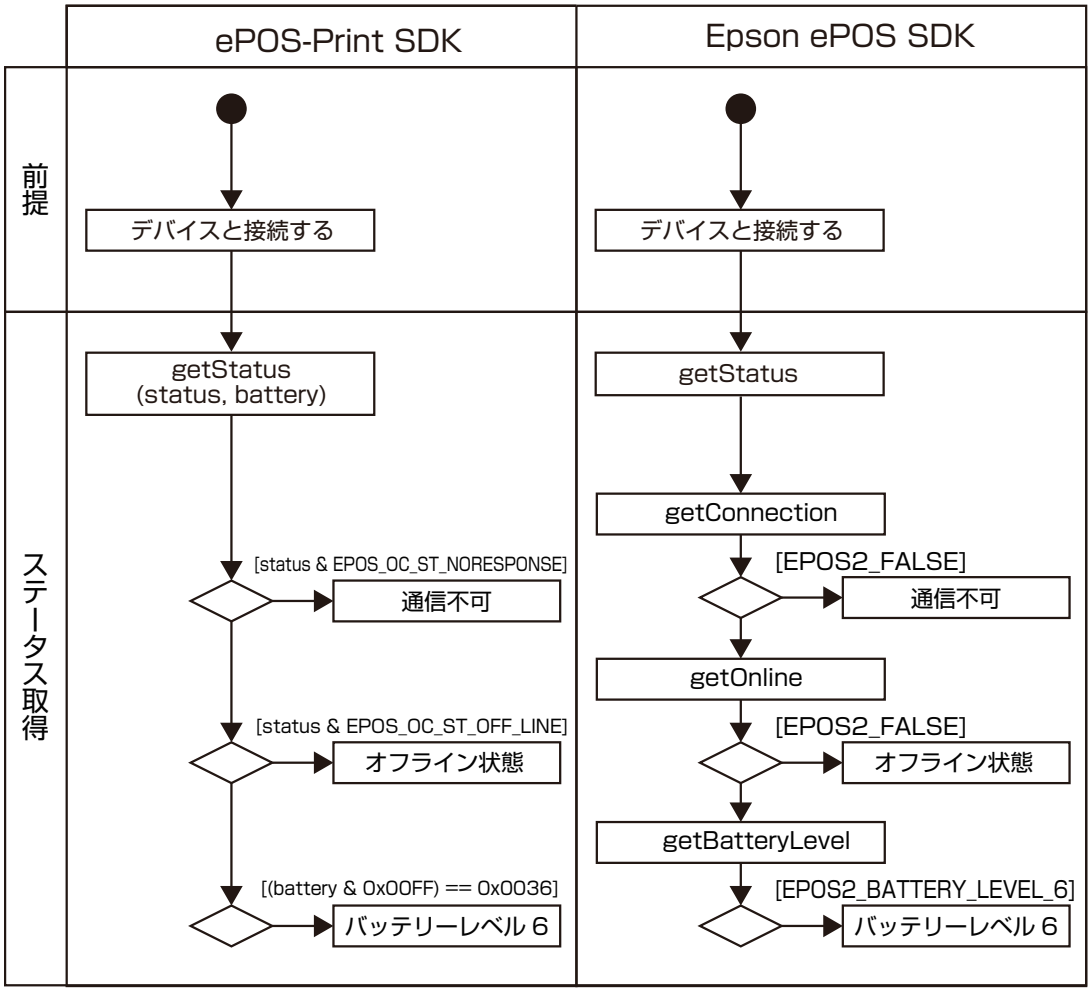
❏ Epson ePOS SDK

```
@interface Sample() <Epos2PtrStatusChangeDelegate>
{
    Epos2Printer *printer_;
}

- (void) openPrinter
{
    ・・・接続・・・
    [printer_ setStatusChangeEventDelegate:self];
    [printer_ startMonitor];
}
- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ・・・処理・・・
}
```

ステータスの取得

ePOS-Print SDK では、複数のプリンターステータスの組み合わせを戻り値で取得していましたが、Epson ePOS SDK では、PrinterStatusInfo 型のプロパティーで各ステータスを取得します。

実行手順の違い



プログラムの違い

❏ ePOS-Print SDK

```
id printer = [[EposPrint alloc] init:];
unsigned long status = 0;
unsigned long battery = 0;
if ( printer != nil) {
   int errorStatus = EPOS_OC_SUCCESS;
   errorStatus = [printer openPrinter:EPOS_OC_DEVTYPE_TCP
   Name:@"192.168.192.168"];
   errorStatus = [printer getStatus:&status Battery:&battery];
   ///Process///
   if(status & EPOS_OC_ST_NO_RESPONSE) {
      // no response
   }
   if(status & EPOS_OC_ST_OFF_LINE){
      // status offline
   }
   if((battery & 0x00FF) == 0x0036){
      // battery level 6
   }
   errorStatus = [printer closePrinter];
}
```

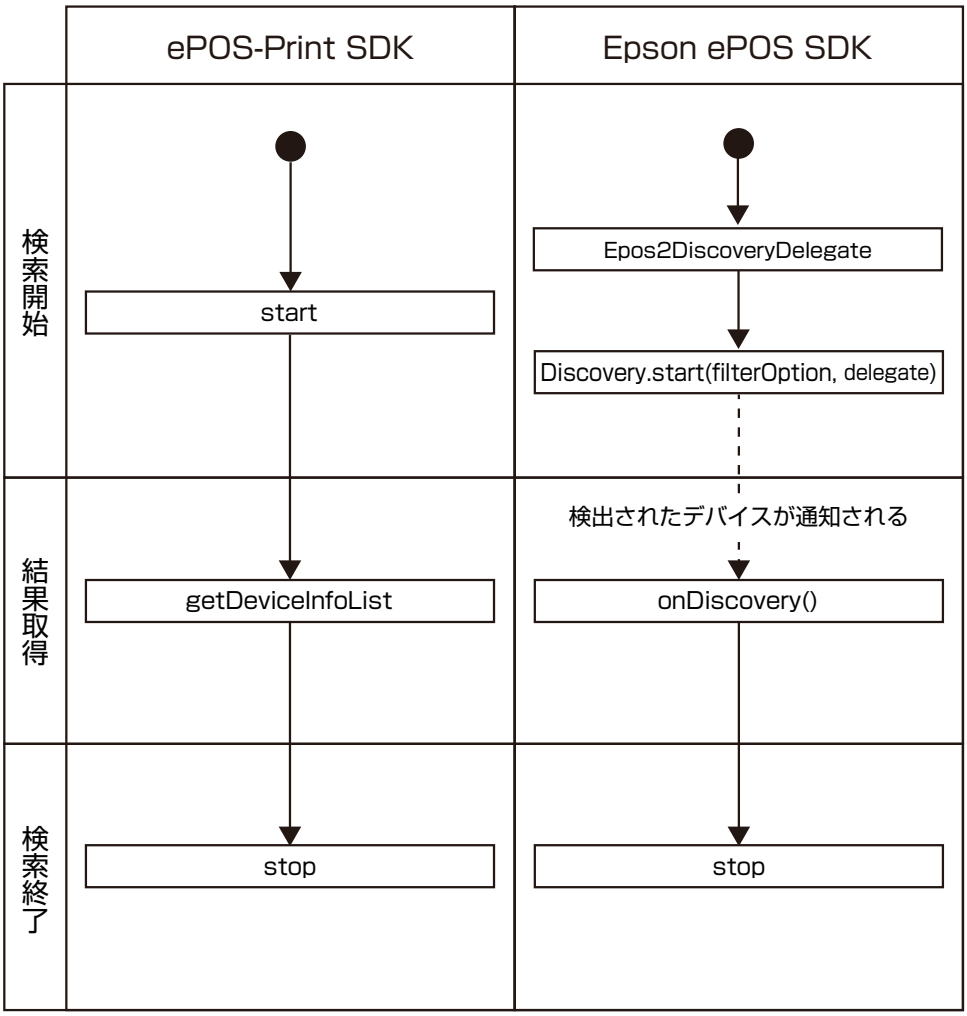
❏ Epson ePOS SDK

```
@interface Sample()
{
   Epos2Printer *printer_;
}

- (void) getStatus
{
     ・・・接続・・・
    Epos2PrinterStatusInfo *status = [printer_ getStatus];

    if([status getConnection] == EPOS2_TRUE) {
        // no response
    }
    if([status getOnline] != EPOS2_TRUE) {
        // status offline
    }
    if([status getBatteryLevel] == EPOS2_BATTERY_LEVEL_6) {
        // offline
    }

}
```

プリンターの検索

ePOS-Print SDK では、プリンターの検索結果を API で取得していましたが、Epson ePOS SDK では、フィルタリング設定しコールバックメソッドで取得します。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Print SDK

```
[EpsonIoFinder start:EPOS_OC_DEVTYPE_TCP FindOption:@"255.255.255.0"];

NSArray *array = [EpsonIoFinder getDeviceInfoList:&errorStatus
                                          FilderOption:EPSONIO_OC_PARAM_DEFAULT];
NSString* deviceName = NULL;
NSString* ipAddress = NULL;
for(int index=0; index < [array count]; index++){
   EpsonIoDeviceInfo deviceInfo = [array objectAtIndex:index];
   deviceName = [deviceInfo deviceName];
   ipAddress = [deviceInfo ipAddress];
}

[EpsonIoFinder stop];
```

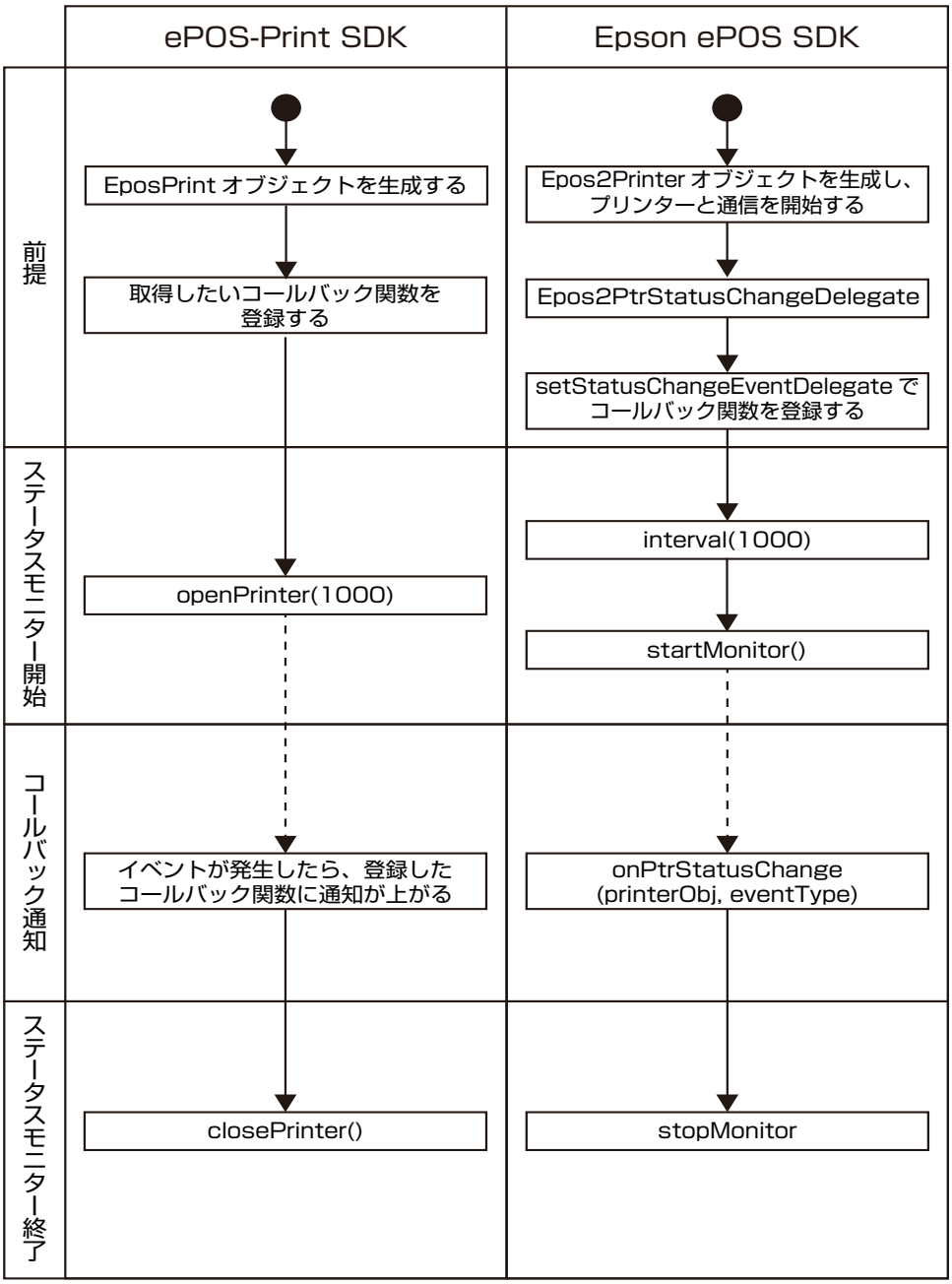
❏ Epson ePOS SDK

```
@interface Sample () <Epos2DiscoveryDelegate>

- (void) discovery
{
    Epos2FilterOption *option = [Epos2FilterOption alloc]init];
    [Epos2Discovery start:option delegate:self];

    ・・・検索中・・・

    [Epos2Discovery stop];
}

- (void) onDiscovery:(Epos2DeviceInfo *)deviceInfo
{
    NSString* target = [deviceInfo getTarget];
}
```

ステータスの監視

ePOS-Print SDKでは、プリンターとの通信とステータスの監視を同時に開始していましたが、Epson ePOS SDK では、プリンターとの通信を開始した後にステータスの監視を開始します。

実行手順の違い



コールバック：- - - -▶

プログラムの違い

❏　ePOS-Print SDK

```
- (void)onStatusChange:(NSString *)deviceName Status:(NSNumber *)status
{
・・・接続・・・
}
- (void)openPrinter
{
   id printer = [[EposPrint alloc] init];
   if ( printer != nil) {
      int errorStatus = EPOS_OC_SUCCESS;
      [printer setStatusChangeEventCallback @selector(onStatusChange:Status:)
Target:self];
      errorStatus = [printer openPrinter:EPOS_OC_DEVTYPE_TCP
         Name:@"192.168.192.168" Enabled: EPOS_OC_TRUE
         Interval:EPOS_OC_PARAM_DEFAULT Timeout:EPOS_OC_PARAM_DEFAULT];
      ・・・接続・・・

      errorStatus = [printer closePrinter];
   }
}
```

❏　Epson ePOS SDK

```
@interface Sample() <Epos2PtrStatusChangeDelegate>
{
   Epos2Printer *printer_;
}

- (void) openPrinter
{
    [printer_ setStatusChangeEventDelegate:self];
    ・・・接続・・・
    [printer_ startMonitor];

    [printer_ stopMonitor];
}

- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ・・・処理・・・
}
```

API の名称変更

ePOS-Print SDK から Epson ePOS SDK にマイグレーションする際に、API の名称に以下の変更を行います。

- 引数のキーワードの頭文字を、大文字から小文字に変更します。以下は、addImage を例にしています。

ePOS-Print SDK

```
errorStatus = [printer addImage: imageData X: 0 Y: 0 Width: 256 Height: 256
               Color: EPOS_OC_PARAM_DEFAULT Mode: EPOS_OC_MODE_MONO
               Halftone: EPOS_OC_HALFTONE_DITHER Brightness: 1.0
               Compress: EPOS_OC_COMPRESS_NONE];
```

Epson ePOS SDK

```
errorStatus = [printer addImage: imageData x: 0 y: 0 width: 256 height: 256
               color: EPOS2_PARAM_DEFAULT mode: EPOS2_MODE_MONO
               halftone: EPOS2_HALFTONE_DITHER brightness: 1.0
               compress: EPOS2_COMPRESS_NONE];
```

- APIの名称を変更します。名称を変更する必要のあるAPIは下表のとおりです。APIによっては複数のAPIが 1 つにまとめられたり、1 つの API が複数の API に分けられたりしたものがあります。下表の API には、名称以外に仕様が変更になっている API もあります。

変更内容は「ePOS-Print SDK for iOS ユーザーズマニュアル」と「Epson ePOS SDK for iOS ユーザーズマニュアル」で API を比較してください。

名称変更する API の一覧表

機能	ePOS-Print SDK	Epson ePOS SDK
クラスの初期化	initWithPrinterModel	initWithPrinterSeries
	init	
改行量設定を命令バッファーに追加	addTextLineSpace	addLineSpace
文字倍角設定を命令バッファーに追加	addTextDouble	addTextSize
文字印字位置設定を命令バッファーに追加	addTextPosition	addHPosition
通信を開始	openPrinter	connect
ステータスの監視を開始		startMonitor
通信を終了	closePrinter	disconnect
ステータスの監視を終了		stopMonitor
プリンターステータスの通知先を登録	setStatusChangeEventCallback	setStatusChangeEventDelegate
オンラインイベントの通知先を登録	setOnlineEventCallback	
オフラインイベントの通知先を登録	setOfflineEventCallback	
無応答イベントの通知先を登録	setPowerOffEventCallback	

機能	ePOS-Print SDK	Epson ePOS SDK
カバークローズイベントの通知先を登録	setCoverOkEventCallback	setStatusChangeEventDelegate
カバーオープンイベントの通知先を登録	setCoverOpenEventCallback	
用紙ありイベントの通知先を登録	setPaperOkEventCallback	
用紙残量少イベントの通知先を登録	setPaperNearEndEventCallback	
用紙なしイベントの通知先を登録	setPaperEndEventCallback	
ドロアークローズイベントの通知先を登録	setDrawerClosedEventCallback	
ドロアーオープンイベントの通知先を登録	setDrawerOpenEventCallback	
バッテリー残量なしイベントの通知先を登録	setBatteryLowEventCallback	
バッテリー残量ありイベントの通知先を登録	setBatteryOkEventCallback	
バッテリーステータスの通知先を登録	setBatteryStatusChangeEventCallback	
プリンターの検索結果を取得	start	start
	getDeviceInfoList	

API のパラメーター変更

ePOS-Print SDK から Epson ePOS SDK にマイグレーションする際に、パラメーターに以下の変更を行います。

- プリンターの簡単選択で使用するパラメーターを、以下の Epson ePOS SDK 用に変更します。

ePOS-Print SDK	Epson ePOS SDK
EPOS_OC_DEVTYPE_TCP	EPOS_EASY_SELECT_DEVTYPE_TCP
EPOS_OC_DEVTYPE_BLUETOOTH	EPOS_EASY_SELECT_DEVTYPE_BLUETOOTH

- パラメーターの命名規則を ePOS-Print SDK 用から Epson ePOS SDK 用に変更します。

ePOS-Print SDK	Epson ePOS SDK
EPOS_OC_****	EPOS2_****

- パラメーターの追加・統合、設定値の追加・削減を行います。パラメーターの変更が必要な API は下表のとおりです。

変更内容は「ePOS-Print SDK for iOS ユーザーズマニュアル」と「Epson ePOS SDK for iOS ユーザーズマニュアル」で API を比較してください。

パラメーターを変更する API の一覧表

API	パラメーターの変更内容
addTextAlign	align の設定値が追加
addTextRotate	rotate の設定値が追加
addTextLang	lang の設定値が追加
addTextFont	font の設定値が追加
addTextSmooth	smooth の設定値が追加
addTextSize	width/ height の設定値が追加
addTextStyle	reverse/ ul/ em/ color の設定値が追加
addImage	compress の設定値が追加
addBarcode	hri/ font の設定値が追加
addPageDirection	direction の設定値が追加
addPagePosition	x/ y の設定値が追加
addSound	pattern/ repeat の設定値が追加 pattern の設定値の名前が変更
sendData	timeout のみに変更
getStatus	パラメーターがなくなり、戻り値で値の取得
createQR	deviceType の設定値の名前が変更

ePOS-Device SDK からのマイグレーション

ePOS-Device SDK を使用したアプリケーションを、Epson ePOS SDK 対応アプリケーションに移行する方法を説明します。

ePOS-Device SDK 互換 API を使用するマイグレーション

既存のアプリケーションのプログラムは修正せずに、構成ファイルを置き換えることで、Epson ePOS SDK 対応のアプリケーションに移行します。

マイグレーションの手順

変更手順は以下のとおりです。

手順		概要
1	SDK の置き換え	ライブラリーファイルの置き換え 「SDK の置き換え」参照
2	アプリケーションのビルド	SDK のファイルを置き換えたアプリケーションのプロジェクトをビルド

以上で ePOS-Device SDK 互換 API を使用したマイグレーションは完了です。

SDK の置き換え

アプリケーションプロジェクトに組まれている以下のファイルを、Epson ePOS SDK のファイルに置き換えます。

種類	ePOS-Device SDK	ePOS-Device SDK 互換 API
ライブラリー	libeposdevice.a	libepos2.a

ePOS-Device SDK 互換 API を使用したアプリケーション開発

ePOS-Device SDK 互換 API を使用したアプリケーションを開発・保守する場合に必要な情報は、以下のマニュアルを参照してください。
ePOS-Device SDK 互換 API の仕様 : 「ePOS-Device SDK for iOS ユーザーズマニュアル」

Epson ePOS SDK の API を使用するマイグレーション

既存のアプリケーションのプログラムを修正して、Epson ePOS SDK 対応のアプリケーションに移行します。プログラムの修正量は多くなりますが、TM プリンターや周辺機器の新製品、新機能に対応していくことができます。

マイグレーションの手順

変更手順は以下のとおりです。

手順		概要
1	SDK の置き換え	ヘッダーファイルとライブラリーファイルの置き換え 「SDK の置き換え」参照
2	フレームワークを追加	アプリケーションのプロジェクトにフレームワークを追加 「フレームワークの追加」参照
3	インポート定義の変更	Objective-C ヘッダーのインポート定義を変更 「インポート定義の変更」参照
4	クラスの変更	ePOS-Device SDK のクラスを Epson ePOS SDK のクラスに変更 「クラスの変更」参照
5	API の変更	Epson ePOS SDK と ePOS-Device SDK で仕様の異なる API の変更やプログラムを修正 変更する内容は、以下のとおりです。 ❏ 特定の機能を実現するためにプログラムを修正 以下の機能を修正してください。 • デバイスとの通信の接続と切断 「デバイスとの通信の接続と切断」参照 • 再接続の通知 「再接続通知」参照 • 印刷 「印刷」参照 • 強制送信 「強制送信」参照 • コールバックの取得 「コールバックの取得」参照 • ステータスの取得 「ステータスの取得」参照 • ステータスの監視 「ステータスの監視」参照 ❏ API の名称変更 API の名称を変更するもの（パラメーターの変更が必要な場合もあります） 「API の名称変更」参照 ❏ API のパラメーター変更 API の名称は変わらないが、パラメーターの変更が必要なもの 「API のパラメーター変更」参照

手順		概要
6	アプリケーションのビルド	修正したアプリケーションのプロジェクトをビルド

以上で Epson ePOS SDK の API を使用したマイグレーションは完了です。

SDK の置き換え

アプリケーションプロジェクトに組まれている以下のファイルを、Epson ePOS SDK のファイルに置き換えます。

種類	ePOS-Device SDK	Epson ePOS SDK
ヘッダーファイル	ePOS-Device.h	ePOS2.h
ライブラリー	libeposdevice.a	libepos2.a

フレームワークの追加

アプリケーションプロジェクトに、以下のフレームワークファイルを組み込みます。

- ExternalAccessory.framework

インポート定義の変更

アプリケーションの *.m ソースファイルに含められている、Objective-C ヘッダーのインポート定義を変更します。

ePOS-Device SDK	Epson ePOS SDK
`#import "ePOS-Device.h"`	`#import "ePOS2.h"`

クラスの変更

アプリケーションのプロジェクトで使用している ePOS-Device SDK のクラスを、Epson ePOS SDK のクラスに変更します。

削除する ePOS-Device SDK のクラス名

クラス名	説明
EposDevice クラス	EposDevice クラスが持っている以下の機能は、各クラスの API に割り当てています。 • 通信経路の確立 • 通信経路の切り離し • 通信経路の確立状態を取得 • 管理者情報の取得 • 設置場所情報の取得 • 再接続処理開始イベントのコールバックメソッドを登録 • 再接続終了イベントのコールバックメソッドを登録 • ネットワーク切断イベントのコールバックメソッドを登録
EposCommBoxManager クラス	CommBox クラスに統合します。

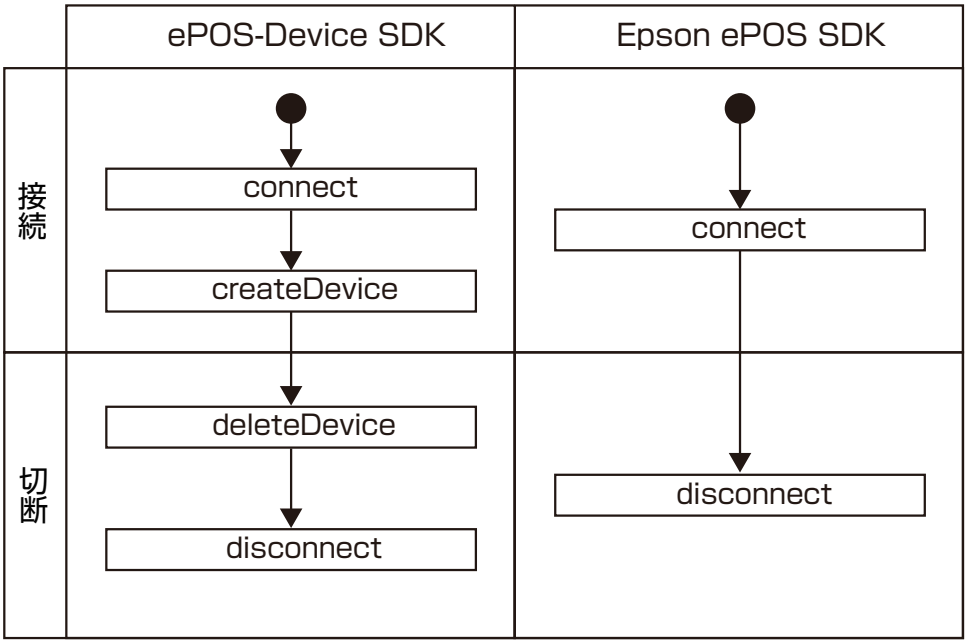
変更するクラス名

種類	ePOS-Device SDK	Epson ePOS SDK
印刷機能	EposPrinter クラス	Epos2Printer クラス
デバイス制御	EposDisplay クラス	Epos2LineDisplay クラス
	EposKeyboard クラス	Epos2Keyboard クラス
	EposScanner クラス	Epos2BarcodeScanner クラス
コマンド送信	EposSimpleSerial クラス	Epos2SimpleSerial クラス
コミュニケーションボックス	EposCommBoxManager クラス	Epos2CommBox クラス
	EposCommBox クラス	
ログ機能	EposDeviceLog クラス	Epos2Log クラス

デバイスとの通信の接続と切断

ePOS-Device SDK では、ePOS-Device Service と接続した後に各デバイスとの通信を開始していましたが、Epson ePOS SDK では、デバイスごと通信を開始します。また ePOS-Device SDK では、デバイスと通信を終了した後に ePOS-Device Service との接続を終了していましたが、Epson ePOS SDK では、デバイスごとに通信を終了します。

実行手順の違い



プログラムの違い

❏ ePOS-Device SDK

```
@interface Sample()
{
    EposDevice *device_;
    EposPrinter *printer_;
}

- (void) openPrinter
{
    device_ = [[EposDevice alloc] init];
    if ( device_ != nil) {
        int errorStatus = EDEV_OC_SUCCESS;
       errorStatus = [device connect:@"192.168.192.168" Collback:@selector(onConnect:Code)
Target:self];
    }
}

- (void) onConnect:(NSString *)ipAddress Code:(int)code
{
    if(code == EDEV_OC_SUCCESS) {
        int errorStatus = [device_ createDevice:@"local_printer"
DeviceType:EDEV_OC_TYPE_PRINTER Crypto:EDEV_OC_FALSE Buffer:EDEV_OC_FALSE
Callback:@selector(onCreateDevice:DeviceId:DeviceType:DeviceObject:Code) Target:self];

    }

}
- (void) onCreateDevice:(NSString *)ipAddress DeviceId:(NSString *)deviceId
DeviceType:(int)deviceType
           DeviceObject:(id)deviceObject Code:(int)code
{
    if(code == EDEV_OC_SUCCESS) {
        if(deviceType == EDEV_OC_TYPE_PRINTER) {
            printer_ = deviceObject;
        }
    }
}

- (void) closePrinter
{
    if(printer_ != nil) {
        int errorStatus = [device_ deleteDevice:printer_
Callback:@selector(onDeleteDevice:DeviceId:Code) Target:self];
    }
}

- (void) onDeleteDevice:(NSString *)ipAddress DeviceId:(NSString *)deviceId Code:(int)code
{
    if(code == EDEV_OC_SUCCESS) {
        [device_ disconnect];
    }
}
```

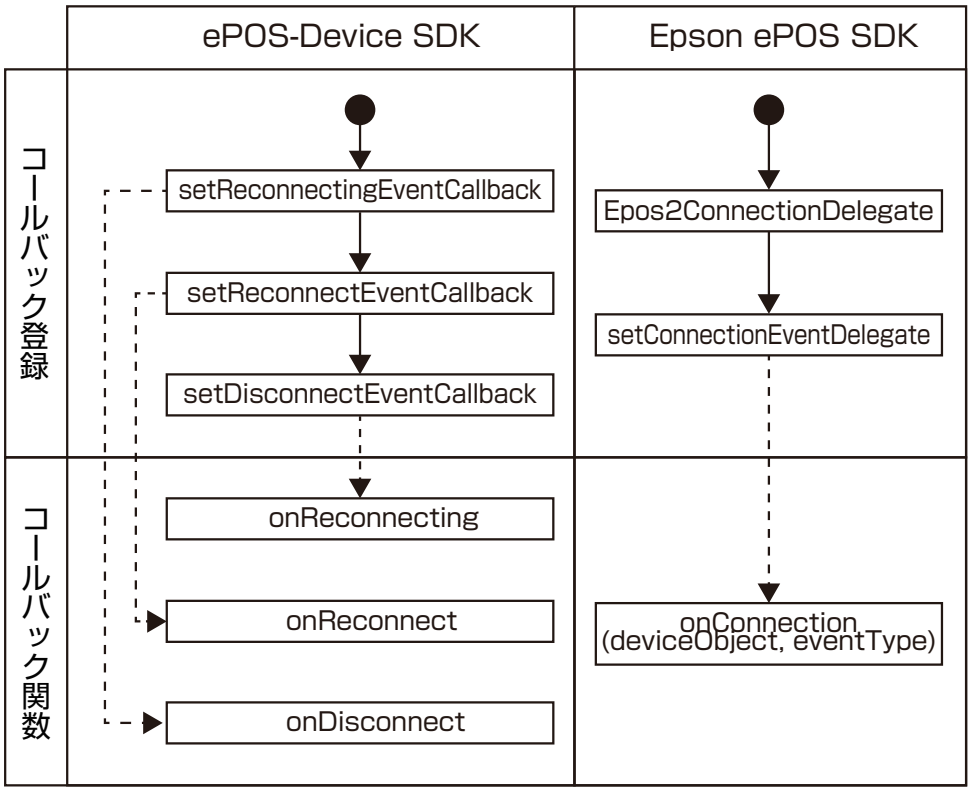
❏ Epson ePOS SDK

```
Epos2Printer printer = [[Epos2Printer alloc] initWithPrinterSeries:EPOS2_TM_T88
                        lang:EPOS2_MODEL_ANK];
if ( printer != nil) {
    int errorStatus = EPOS2_SUCCESS;
    errorStatus = [printer connect:@"TCP:192.168.192.168" timeout:EPOS2_PARAM_DEFAULT];
    ・・・処理・・・
    errorStatus = [printer disconnect];
}
```

再接続通知

ePOS-Device SDK では、通知の種類ごと API で登録していましたが、Epson ePOS SDK では、通知先登録 API が一つに統合され、通知先メソッドで通知の種類ごと処理します。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Device SDK

```
@interface Sample()
{
    EposDevice *device_;
}

- (void) openPrinter
{
    device_ = [EposDevice alloc]init];
    if(device_ != nil) {
        [device_ setReconnectingEventCallback:@selector(onReconnect:) Target:self];
        [device_ setReconnectEventCallback:@selector(onReconnect:) Target:self];
        [device_ setDisconnectEventCallback:@selector(onReconnect:) Target:self];
        ・・・接続・・・
    }
}
- (void) onReconnecting:(NSString *)ipAddress
{
    ・・・再接続開始・・・
}
- (void) onReconnect:(NSString *)ipAddress
{
    ・・・再接続完了・・・
}
- (void) onDisconnect:(NSString *)ipAddress
{
    ・・・再接続失敗・・・
}
```

❏ Epson ePOS SDK

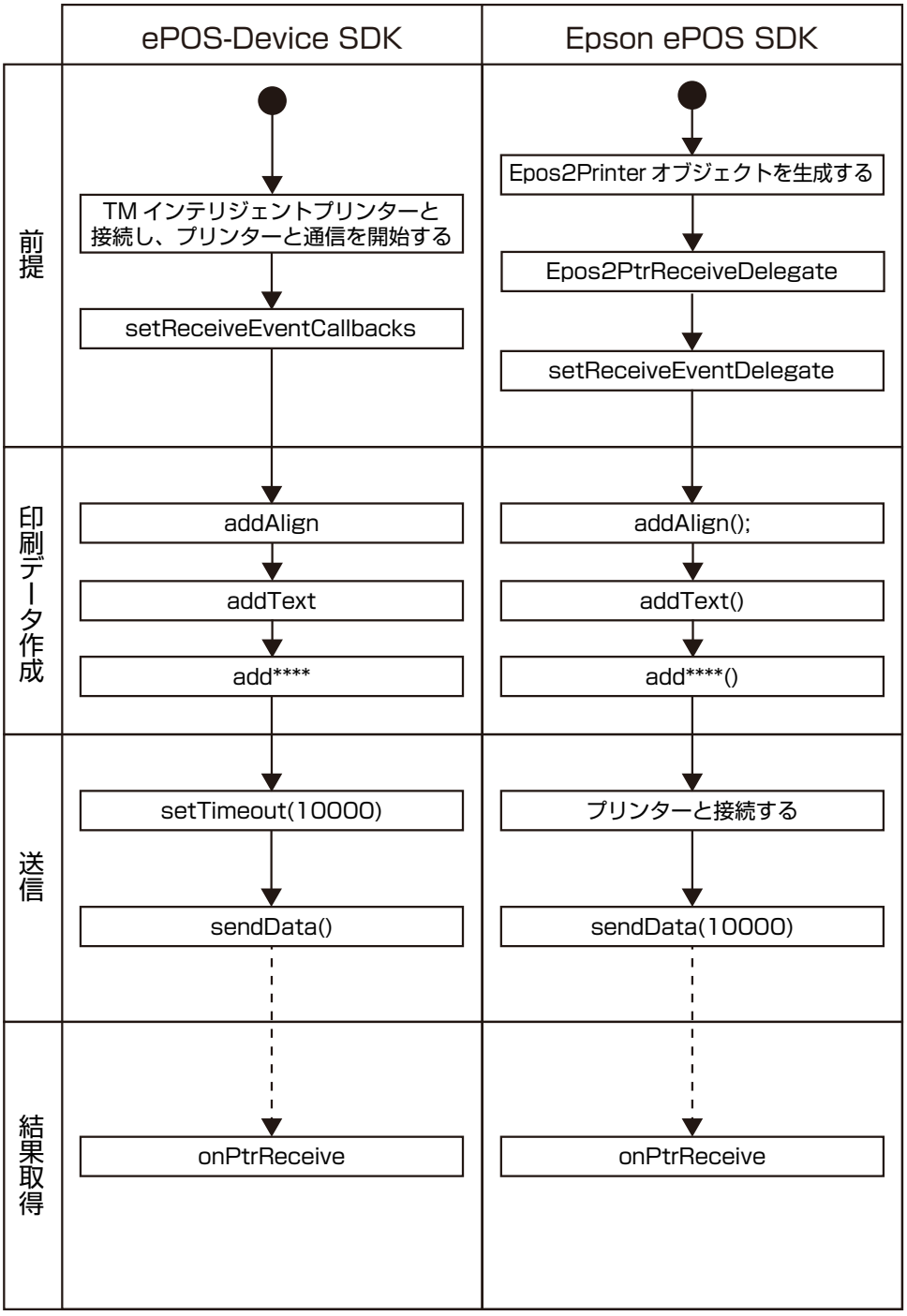
```
@interface Sample() <Epos2ConnectionDelegate>
{
   Epos2Printer *printer_;
}

- (void) openPrinter
{
    printer_ = [[Epos2Printer alloc] initWithPrinterSeries:EPOS2_TM_T88
                lang:EPOS2_MODEL_ANK];
    if(printer_ != nil) {
        [printer_ setConnectionEventDelegate:self];
        ・・・接続・・・
    }
}
- (void) onConnection:(id)deviceObj eventType:(int)eventType
{
    if(eventType == EPOS2_EVENT_RECONNECTING) {
        ・・・再接続開始・・・
    }
    if(eventType == EPOS2_EVENT_RECONNECT) {
        ・・・再接続完了・・・
    }
    if(eventType == EPOS2_EVENT_DISCONNECT) {
        ・・・再接続失敗・・・
    }
}
```

印刷

ePOS-Device SDK では、プリンターと接続した後に印刷データ作成を行っていましたが、Epson ePOS SDK では、プリンターの接続前、後のどちらでも作成できます。
既存のプログラムの実行手順を修正しなくても、印刷できます。

実行手順の違い

Epson ePOS SDK の実行手順は、プリンターと接続する前に印刷データを作成する手順です。



コールバック : - - - -▶

プログラムの違い

❏　ePOS-Device SDK

```
@interface Sample()
{
    EposDevice *device_;
    EposPrinter *printer_;
}

- (void) openPrinter
{
    ・・・接続・・・

    [printer_ setReceiveEventCallback:@selector(onReceive:DeviceId:Success:Code:Status:
                                                Battery) Target:self];
    int errorStatus = [printer_ addText:@"ABCDE"];
    errorStatus = [printer_ sendData];
}

- (void)onReceive:(NSString *)ipAddress DeviceId:(NSString *)deviceId
          Success:(int)success
             Code:(int)code
           Status:(NSNumber *)status
          Battery:(NSNumber *)battery
{
    ・・・処理・・・
}
```

❏　Epson ePOS SDK

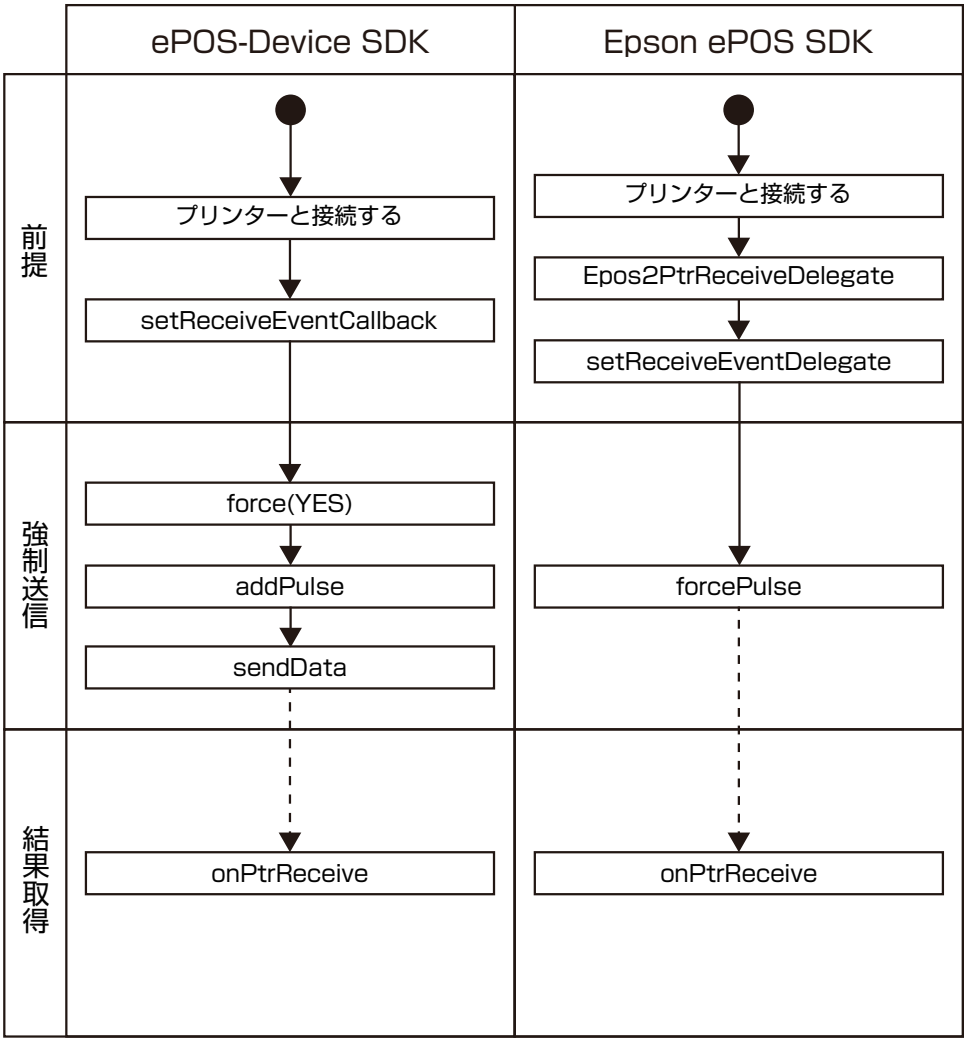
以下のプログラムは、プリンターと接続する前に印刷データを作成しています。

```
@interface Sample() <Epos2PtrReceiveDelegate>
{
   Epos2Printer *printer_;
}

- (void) openPrinter
{
    [printer_ setReceiveEventDelegate:self];
    int errorStatus = [printer_ addText:@"ABCDE"];
    ・・・接続・・・
    errorStatus = [printer_ sendData:EPOS2_PARAM_DEFAULT];
}


- (void) onPtrReceive:(Epos2Printer *)printerObj code:(int)code
status:(Epos2PrinterStatusInfo *)status printJobId:(NSString *)printJobId
{
    ・・・処理・・・
}
```

強制送信

ePOS-Device SDK では、強制送信を 3 つの API を使って実行していましたが、Epson ePOS SDK では、1 つの API で実行します。
また ePOS-Device SDK では、強制送信はオフライン時のみ有効でしたが、Epson ePOS SDK では、オンライン、オフライン時の両方で使用できます。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Device SDK

```
@interface Sample()
{
    EposPrinter *printer_;
}

- (void) openPrinter
{
    ・・・接続・・・

    [printer_ setReceiveEventCallback:@selector(onReceive:DeviceId:Success:Code:Status:
                                                Battery) Target:self];
    [printer_ setForce:YES];
    [printer_ addPulse:EDEV_OC_DRAWER_1 Time:EDEV_OC_PULSE_100];
    [printer_ sendData];
}

- (void)onReceive:(NSString *)ipAddress DeviceId:(NSString *)deviceId
            Success:(int)success
               Code:(int)code
             Status:(NSNumber *)status
            Battery:(NSNumber *)battery
{
    ・・・処理・・・
}
```

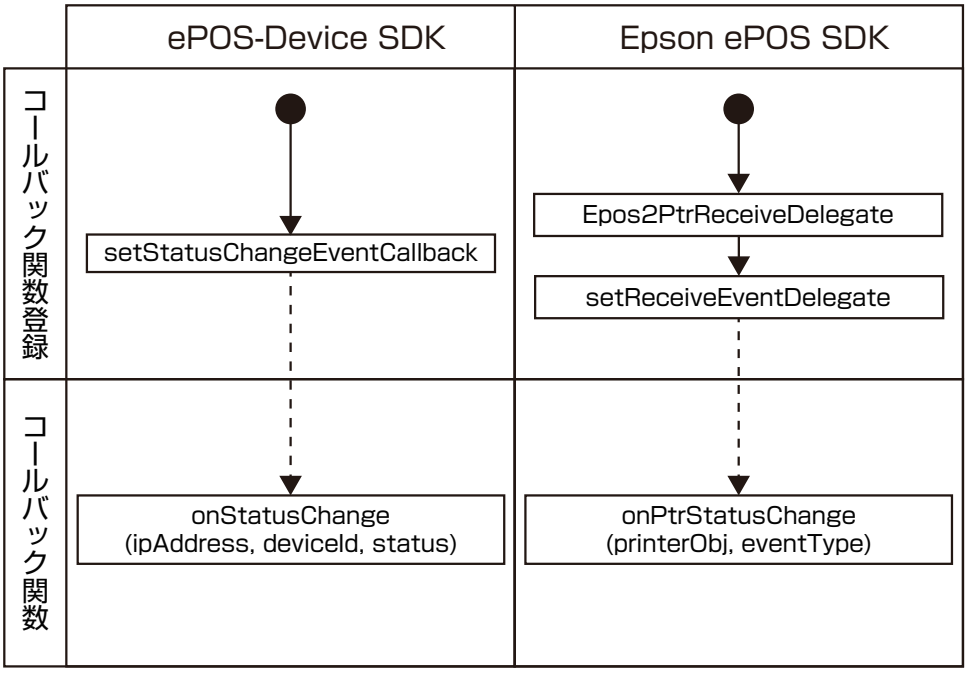
❏ Epson ePOS SDK

```
@interface Sample() <Epos2PtrReceiveDelegate>
{
   Epos2Printer *printer_;
}

- (void) openPrinter
{
    ・・・接続・・・
    [printer_ setReceiveEventDelegate:self];
    int errorStatus = [printer_ forcePulse:EPOS2_PARAM_DEFAULT pulseTime:EPOS2_PULSE_100
                       timeout:EPOS2_PARAM_DEFAULT];
}

- (void) onPtrReceive:(Epos2Printer *)printerObj code:(int)code
status:(Epos2PrinterStatusInfo *)status printJobId:(NSString *)printJobId
{
    ・・・処理・・・
}
```

コールバックの取得

ePOS-Device SDK ではセレクターを使ったコールバック処理でしたが、Epson ePOS SDK ではプロトコルを使ったコールバック処理を行います。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Device SDK

```
@interface Sample() {
    EposDevice *device_;
    EposPrinter *printer_;
}
@end

- (void) openPrinter
{
    ・・・接続・・・

    [printer_ setStatusChangeEventCallback:@selector(onStatusChange:DeviceId:Status:)
                                    Target:self];
    [printer_ startMonitor];
}

- (void)onStatusChange:(NSString *)ipAddress DeviceId:(NSString *)deviceId
Status:(NSNumber *)status
{
    ・・・処理・・・
}
@end
```

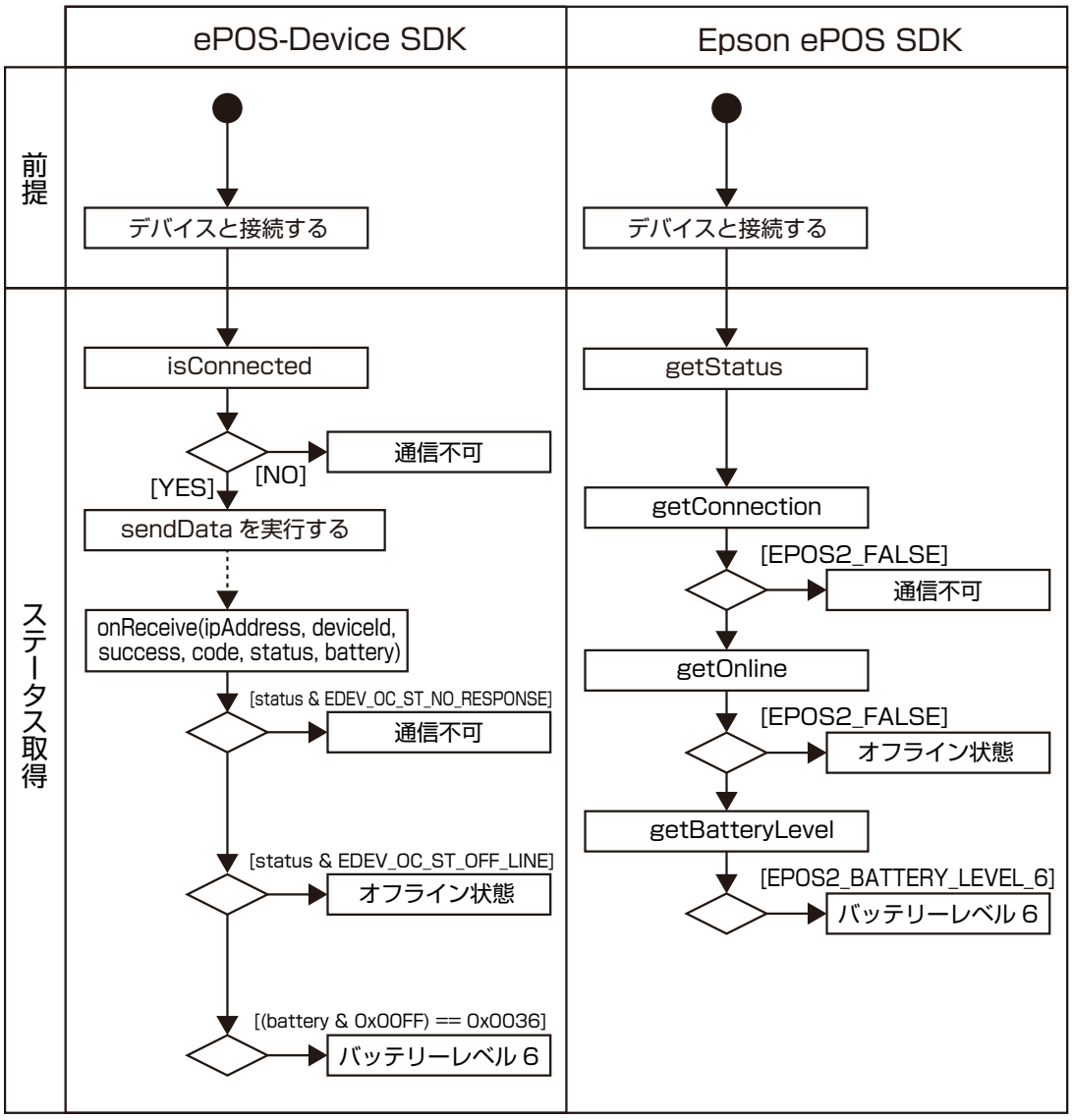
❏ Epson ePOS SDK

```
@interface Sample() <Epos2PtrStatusChangeDelegate>
{
    Epos2Printer *printer_;
}

- (void) openPrinter
{
    ・・・接続・・・
    [printer_ setStatusChangeEventDelegate:self];
    [printer_ startMonitor];
}
- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ・・・処理・・・
}
```

ステータスの取得

ePOS-Device SDK では、複数のプリンターステータスの組み合わせを戻り値で取得していましたが、Epson ePOS SDK では、PrinterStatusInfo 型のプロパティーで各ステータスを取得します。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Device SDK

```
@interface Sample()
{
    EposDevice *device_;
    EposPrinter *printer_;
}

- (void) getStatus
{
    ・・・接続・・・

    if([device_ isConnected] != YES) {
        ・・・切断中・・・
    }
    [printer_ setReceiveEventCallback:@selector(onReceive:DeviceId:Success:Code:Status
                                        :Battery) Target:self];
    int errorStatus = [printer_ sendData];
}

- (void) onReceive:(NSString *)ipAddress
          DeviceId:(NSString *)deviceId
           Success:(int)success
              Code:(int)code
            Status:(NSNumber *)status
           Battery:(NSNumber *)battery
{
    if((status & EDEV_OC_ST_NO_RESPONSE) == EDEV_OC_ST_NO_RESPONSE) {
        // no response
    }
    if((status & EDEV_OC_ST_OFF_LINE) == EDEV_OC_ST_OFF_LINE){
        // status offline
    }
    if((battery & 0x00FF) == 0x0036){
        // battery level 6
    }
}
```

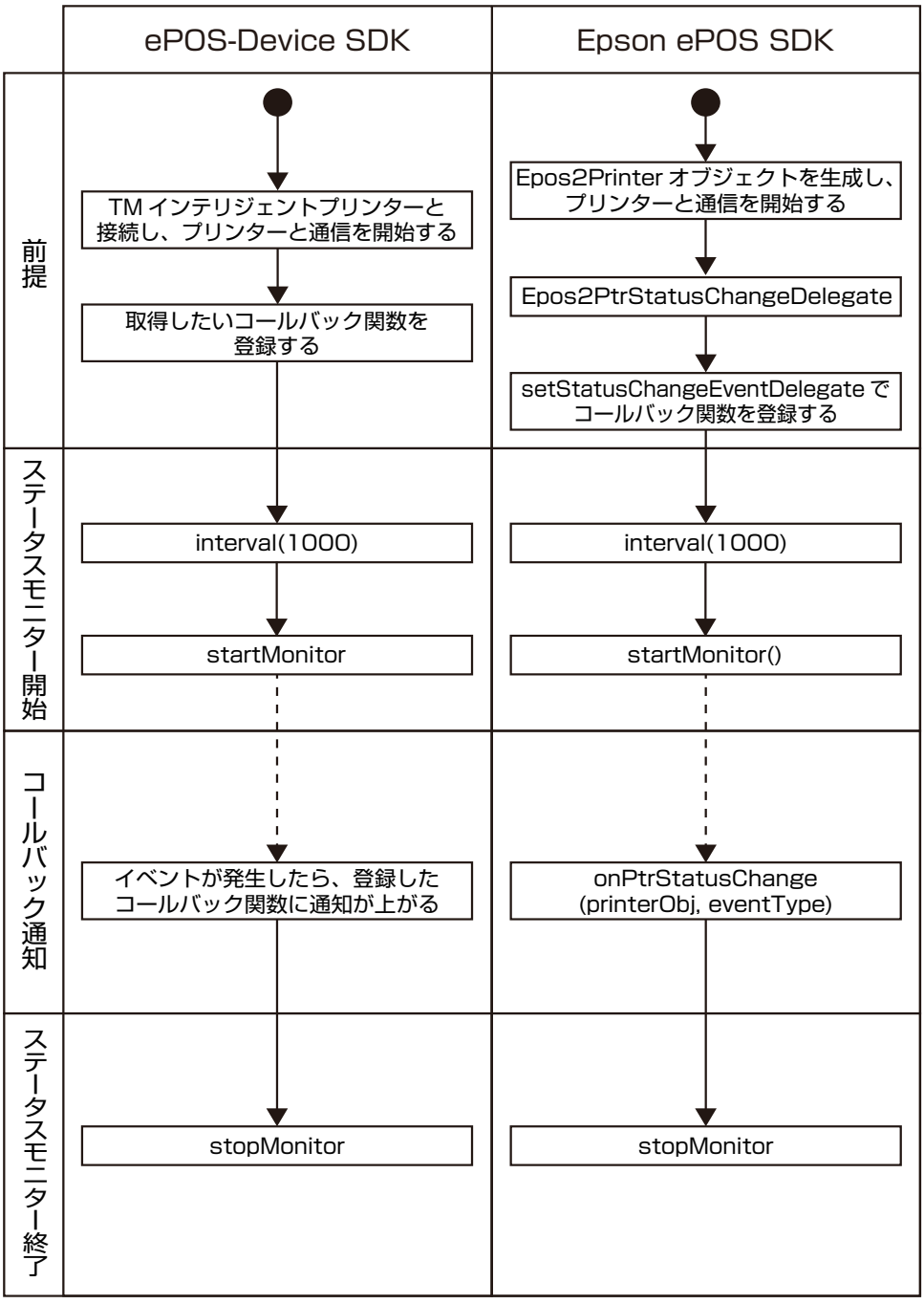
❏ Epson ePOS SDK

```
@interface Sample()
{
   Epos2Printer *printer_;
}

- (void) getStatus
{
    ・・・接続・・・
    Epos2PrinterStatusInfo *status = [printer_ getStatus];

    if([status getConnection] == EPOS2_TRUE) {
        // no response
    }
    if([status getOnline] != EPOS2_TRUE) {
        // status offline
    }
    if([status getBatteryLevel] == EPOS2_BATTERY_LEVEL_6) {
        // offline
    }

}
```

ステータスの監視

ePOS-Device SDK では、ステータスの種類ごとに API で登録していましたが、Epson ePOS SDK では、通知先登録 API が一つにまとめられ、通知先メソッドで処理を選択します。

実行手順の違い



コールバック : - - - -▶

プログラムの違い

❏ ePOS-Device SDK

```
@interface Sample()
{
    EposPrinter *printer_;
}

- (void) openPrinter
{
    ・・・接続・・・

    [printer_ setStatusChangeEventCallback @selector(onStatusChange:DeviceId:Status:)
              Target:self];
    [printer_ startMonitor];

    [printer_ stopMonitor];
}

- (void)onStatusChange:(NSString *)ipAddress DeviceId:(NSString *)deviceId
        Status:(NSNumber *)status
{
    ・・・処理・・・
}
```

❏ Epson ePOS SDK

```
@interface Sample() <Epos2PtrStatusChangeDelegate>
{
   Epos2Printer *printer_;
}

- (void) openPrinter
{
    [printer_ setStatusChangeEventDelegate:self];
    ・・・接続・・・
    [printer_ startMonitor];

    [printer_ stopMonitor];
}

- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ・・・処理・・・
}
```

API の名称変更

ePOS-Device SDK から Epson ePOS SDK にマイグレーションする際に、API の名称に以下の変更を行います。

- 引数のキーワードの頭文字を、大文字から小文字に変更します。以下は、addImage を例にしています。

ePOS-Device SDK

```
errorStatus = [printer addImage: imageData X: 0 Y: 0 Width: 256 Height: 256
               Color: EDEV_OC_PARAM_DEFAULT Mode: EDEV_OC_MODE_MONO];
```

Epson ePOS SDK

```
errorStatus = [printer addImage: imageData x: 0 y: 0 width: 256 height: 256
               color: EPOS2_PARAM_DEFAULT mode: EPOS2_MODE_MONO
               halftone: EPOS2_HALFTONE_DITHER brightness: 1.0
               compress: EPOS2_COMPRESS_NONE];
```

- APIの名称を変更します。名称を変更する必要のあるAPIは下表のとおりです。APIによっては複数のAPIが 1 つにまとめられたり、1 つの API が複数の API に分けられたり、削除したものがあります。下表の API には、名称以外に仕様が変更になっている API もあります。

変更内容は「ePOS-Device SDK for iOS ユーザーズマニュアル」と「Epson ePOS SDK for iOS ユーザーズマニュアル」で API を比較してください。

名称変更する API の一覧表

クラス / 機能	ePOS-Device SDK	Epson ePOS SDK
クラス共通		
クラスの初期化	init	initWithPrinterSeries
		initWithDisplaySeries
		init
デバイスのオブジェクトを取得	connect	connect
	createDevice	
デバイスのオブジェクトを破棄	disconnect	disconnect
	deleteDevice	
現在のステータス情報を取得	isConnected	getStatus
再接続処理開始イベントの通知先を登録	setReconnectingEventCallback	setConnectionEventDelegate
再接続終了イベントの通知先を登録	setReconnectEventCallback	
ネットワーク切断イベントの通知先を登録	setDisconnectEventCallback	

クラス / 機能	ePOS-Device SDK	Epson ePOS SDK
EposDisplay クラス		
表示領域の定義を命令バッファーに追加	createWindow	addCreateWindow
表示領域の設定破棄を命令バッファーに追加	destroyWindow	addDestroyWindow
表示領域の切り替えを命令バッファーに追加	setCurrentWindow	addSetCurrentWindow
現在の表示領域の消去を命令バッファーに追加	clearWindow	addClearCurrentWindow
カーソル位置を命令バッファーに追加	setCursorPosition	addSetCursorPosition
表示領域内でカーソル位置を命令バッファーに追加	moveCursorPosition	addMoveCursorPosition
カーソルの種類の変更を命令バッファーに追加	setCursorType	addSetCursorType
マーキー表示を命令バッファーに追加	addMarquee	addMarqueeText
表示の点滅情報を命令バッファーに追加	setBlink	addSetBlink
表示輝度の情報を命令バッファーに追加	setBrightness	addSetBrightness
時計の時刻を命令バッファーに追加	ShowClock	addShowClock
カスタマーディスプレイを初期化	reset	addInitiailze
制御結果受信イベントの通知先を登録	SetReceiveEventCallback	setReceiveEventDelegate
EposKeyboard クラス		
キー押下検出イベントの通知先を登録	setKeyPressEventCallback	setKeyPressEventDelegate
文字列検出イベントの通知先を登録	setStringEventCallback	setReadStringEventDelegate
EposPrinter クラス		
改行量設定を命令バッファーに追加	addTextLineSpace	addLineSpace
文字倍角設定を命令バッファーに追加	addTextSize	addTextSize
	addTextDouble	
文字印字位置設定を命令バッファーに追加	addTextPosition	addHPosition
縦方向の印字開始位置設定を命令バッファーに追加	addTextVPosition	addPagePosition

クラス 機能	ePOS-Device SDK	Epson ePOS SDK
EposPrinter クラス		
改行を命令バッファーに追加	addFeedLine	addFeedLine
	addFeed	
復帰可能エラーから復帰	recover	forceRecover
エラーからの復帰タグを追加する	addRecovery	
プリンターにリセットコマンドを強制送信	reset	forceReset
プリンターのリセットタグを追加する	addReset	
印刷結果を取得	getPrintJobStatus	requestPrintJobStatus
現在のステータスを取得	setOnlineEventCallback	getStatus
ラスターイメージのハーフトーン処理方法を設定	halftone プロパティー	addImage
ラスターイメージの明るさ補正値を設定	brightness プロパティー	
強制送信	force プロパティー	forceRecover
		forcePulse
		forceStopSound
		forceCommand
送信タイムアウト時間を設定する	timeout プロパティー	削除
ドロアーの信号線状態	drawerOpenLevel プロパティー	削除
プリンターステータスの通知先を登録	setStatusChangeEventCallback	setStatusChangeEventDelegate
オンラインイベントの通知先を登録	setOnlineEventCallback	
オフラインイベントの通知先を登録	setOfflineEventCallback	
無応答イベントの通知先を登録	setPowerOffEventCallback	
カバークローズイベントの通知先を登録	setCoverOkEventCallback	
カバーオープンイベントの通知先を登録	setCoverOpenEventCallback	
用紙ありイベントの通知先を登録	setPaperOkEventCallback	
用紙残量少イベントの通知先を登録	setPaperNearEndEventCallback	
用紙なしイベントの通知先を登録	setPaperEndEventCallback	

クラス 機能	ePOS-Device SDK	Epson ePOS SDK
EposPrinter クラス		
ドロアークローズイベントの通知先を登録	setDrawerClosedEventCallback	setStatusChangeEventDelegate
ドロアーオープンイベントの通知先を登録	setDrawerOpenEventCallback	
バッテリー残量なしイベントの通知先を登録	setBatteryLowEventCallback	
バッテリー残量ありイベントの通知先を登録	setBatteryOkEventCallback	
バッテリーステータスの通知先を登録	setBatteryStatusChangeEventCallback	
応答ドキュメント受信イベントの通知先を登録	setReceiveEventCallback	setReceiveEventDelegate
EposScanner クラス		
バーコードデータ入力イベントの通知先を登録	setDataEventCallback	setScanEventDelegate
EposSimpleSerial クラス		
デバイスからの受信イベントの通知先を登録	setCommandReplyEventCallback	setReceiveEventDelegate
EposCommBox クラス		
コミュニケーションボックスを作成する	openCommBox	connect
コミュニケーションボックスを破棄する	closeCommBox	disconnect
メッセージをコミュニケーションボックスに送信	sendData	sendMessage
コミュニケーションボックスのメッセージ受信の通知先を登録	setReceiveEventCallback	setReceiveEventDelegate

API のパラメーター変更

ePOS-Device SDK から Epson ePOS SDK にマイグレーションする際に、パラメーターに以下の変更を行います。

- パラメーターの命名規則を ePOS-Device SDK 用から Epson ePOS SDK 用に変更します。

ePOS-Device SDK	Epson ePOS SDK
EDEV_OC_****	EPOS2_****

- パラメーターの追加・統合、設定値の追加・削減を行います。パラメーターの変更が必要な API は下表のとおりです。

変更内容は「ePOS-Print SDK for iOS ユーザーズマニュアル」と「Epson ePOS SDK for iOS ユーザーズマニュアル」で API を比較してください。

パラメーターを変更する API の一覧表

クラス / API	パラメーターの変更内容
EposDisplay クラス	
addText	lang の設定値が追加
addReverseText	lang の設定値が追加
setReceiveEventListener	コールバックメソッドの code の値が追加
EposPrinter クラス	
sendData	timeout のみに変更
addTextAlign	align の設定値が追加
addTextRotate	rotate の設定値が追加
addTextLang	lang の設定値が追加
addTextFont	font の設定値が追加
addTextSmooth	smooth の設定値が追加
addTextSize	width/ height の設定値が追加
addTextStyle	reverse/ ul/ em/ color の設定値が追加
addImage	halftone/ brightness パラメーターが追加 compress の設定値が追加
addBarcode	hri/ font/ width/ height の設定値が追加
addSymbol	level/ width/ height/ size の設定値が追加
addPageDirection	direction の設定値が追加
addPagePosition	x/ y の設定値が追加

クラス / API	パラメーターの変更内容
EposPrinter クラス	
setReceiveEventListener	コールバックメソッドの status/ battery パラメーターが統合 コールバックメソッドの code の値が削減
interval プロパティー	設定値が追加
EposCommBox クラス	
getCommHistory	コールバックメソッドの code の値が削減

付録

ePOS-Print SDK 互換 API

ePOS-Print SDK 互換 API でサポートしている、エプソン製 TM プリンターの新製品の機種情報とサポート API について説明します。

プリンターごとのサポート API 一覧

各プリンターのサポート API を一覧表で掲載します。

- ○：対応している。
- -：対応していない。

API	TM-m10	TM-m30
addTextAlign	○	○
addTextLineSpace	○	○
addTextRotate	○	○
addText	○	○
addTextLang	○	○
addTextFont	○	○
addTextSmooth	○	○
addTextDouble	○	○
addTextSize	○	○
addTextStyle	○	○
addTextPosition	○	○
addFeedUnit	○	○
addFeedLine	○	○
addImage	○	○
addImage(旧フォーマット)	○	○
addImage(旧フォーマット)	○	○
addLogo	○	○
addBarcode	○	○
addSymbol	○	○
addPageBegin	○	○

API	TM-m10	TM-m30
addPageEnd	○	○
addPageArea	○	○
addPageDirection	○	○
addPagePosition	○	○
addPageLine	-	-
addPageRectangle	-	-
addCut	○	○
addPulse	○	○
addSound	○	○
addSound(旧フォーマット)	○	○
addFeedPosition	-	-
addLayout	-	-
addCommand	○	○

TM-m10

TM-m10 の機種情報は以下のとおりです。

		58 mm
解像度		203 x 203 dpi
言語		ANK モデル 日本語モデル 繁体字中国語モデル
印字幅		420 ドット
印字桁数	フォント A	ANK: 35 桁 / 漢字 : 17 桁
	フォント B	ANK: 42/ 漢字 : 21 桁
	フォント C	ANK: 46 桁
文字サイズ	フォント A	ANK: 12 x 24 ドット / 漢字 : 24 x 24 ドット
	フォント B	ANK: 10x 24 ドット / 漢字 : 20 x 24 ドット
	フォント C	ANK: 9 x 17 ドット
文字のベースライン	フォント A	文字の上端から 21 ドット目
	フォント B	文字の上端から 21 ドット目
	フォント C	文字の上端から 16 ドット目
初期改行量		30 ドット
色指定		第 1 色
ページモード初期領域		420 x 2400 ドット
ページモード最大領域		420 x 2400 ドット
バーコード		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded
2 次元シンボル		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, Data Matrix (Composite Symbology not supported)
用紙のカット		カット / フィードカット
ドロアーキック		サポート
ブザー		オプション (パターン A ～パターン E, エラー , 用紙なし , ストップ)
バッテリー		非サポート
Bluetooth 接続		TM-m10 *Bluetooth*® モデルのみサポート

API 情報

API については、「ePOS-Print SDK for iOS ユーザーズマニュアル」を参照してください。
API の差分情報は以下のとおりです。

API	パラメーター	指定可能な設定値	説明
initWithPrinterModel	printerModel	“TM-m10”	• TM-m10 USB モデル • TM-m10 Ethernet モデル • TM-m10 Wi-Fi モデル • TM-m10 *Bluetooth*® モデル
	lang	EPOS_OC_MODEL_ANK	ANK モデル
		EPOS_OC_MODEL_JAPANESE	日本語モデル
		EPOS_OC_MODEL_TAIWAN	繁体字中国語モデル
addTextFont	font	EPOS_OC_FONT_A	フォント A
		EPOS_OC_FONT_B	フォント B
		EPOS_OC_FONT_C	フォント C
addImage	mode	EPOS_OC_MODE_MONO	モノクロ (2 階調)
		EPOS_OC_MODE_GRAY16	多階調 (16 階調)
		EPOS_OC_PARAM_DEFAULT	既定値 (モノクロ (2 階調))
	compress	EPOS_OC_COMPRESS_DEFLATE	画像を圧縮する。
		EPOS_OC_COMPRESS_NONE	画像を非圧縮する。
		EPOS_OC_PARAM_DEFAULT	既定値 (画像を非圧縮する。)

TM-m30

TM-m30 の機種情報は以下のとおりです。

		58 mm	80 mm
解像度		203 x 203 dpi	
言語		ANK モデル 日本語モデル 簡体字中国語モデル 繁体字中国語モデル	
印字幅		420 ドット	576 ドット
印字桁数	フォント A	ANK 35 桁 / 漢字 17 桁	ANK 48 桁 / 漢字 24 桁
	フォント B	ANK 42 桁 / 漢字 21 桁	ANK 57 桁 / 漢字 28 桁
	フォント C	ANK 46 桁	ANK 64 桁
文字サイズ	フォント A	ANK 12 x 24 ドット / 漢字 24 x 24 ドット	
	フォント B	ANK 10 x 24 ドット / 漢字 20 x 24 ドット	
	フォント C	ANK 9 x 17 ドット	
文字のベースライン	フォント A	文字の上端から 21 ドット目	
	フォント B	文字の上端から 21 ドット目	
	フォント C	文字の上端から 16 ドット目	
初期改行量		30 ドット	
色指定		第 1 色	
ページモード初期領域		420 x 2400 ドット	576 x 2400 ドット
ページモード最大領域		420 x 2400 ドット	576 x 2400 ドット
バーコード		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded	
2 次元シンボル		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, Data Matrix (Composite Symbology not supported)	
用紙のカット		カット / フィードカット	
ドロアーキック		サポート	
ブザー		オプション (パターン A ～パターン E, エラー , 用紙なし , ストップ)	
バッテリー		非サポート	
Bluetooth 接続		TM-m30 *Bluetooth*® モデルのみサポート	

API 情報

API については、「ePOS-Print SDK for iOS ユーザーズマニュアル」を参照してください。
API の差分情報は以下のとおりです。

API	パラメーター	指定可能な設定値	説明
initWithPrinterModel	printerModel	“TM-m30”	• TM-m30 標準モデル • TM-m30 *Bluetooth*® モデル
	lang	EPOS_OC_MODEL_ANK	ANK モデル
		EPOS_OC_MODEL_JAPANESE	日本語モデル
		EPOS_OC_MODEL_CHINESE	簡体字中国語モデル
		EPOS_OC_MODEL_TAIWAN	繁体字中国語モデル
addTextFont	font	EPOS_OC_FONT_A	フォント A
		EPOS_OC_FONT_B	フォント B
		EPOS_OC_FONT_C	フォント C
addImage	mode	EPOS_OC_MODE_MONO	モノクロ (2 階調)
		EPOS_OC_MODE_GRAY16	多階調 (16 階調)
		EPOS_OC_PARAM_DEFAULT	既定値 (モノクロ (2 階調))
	compress	EPOS_OC_COMPRESS_DEFLATE	画像を圧縮する。
		EPOS_OC_COMPRESS_NONE	画像を非圧縮する。
		EPOS_OC_PARAM_DEFAULT	既定値 (画像を非圧縮する。)