

Epson ePOS SDK for iOS

Migration Guide

Migration Overview

Migration from ePOS-Print SDK

Migration from ePOS-Device SDK

Appendix

Cautions

- No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Seiko Epson Corporation.
- The contents of this document are subject to change without notice. Please contact us for the latest information.
- While every precaution has been taken in the preparation of this document, Seiko Epson Corporation assumes no responsibility for errors or omissions.
- Neither is any liability assumed for damages resulting from the use of the information contained herein.
- Neither Seiko Epson Corporation nor its affiliates shall be liable to the purchaser of this product or third parties for damages, losses, costs, or expenses incurred by the purchaser or third parties as a result of: accident, misuse, or abuse of this product or unauthorized modifications, repairs, or alterations to this product, or (excluding the U.S.) failure to strictly comply with Seiko Epson Corporation's operating and maintenance instructions.
- Seiko Epson Corporation shall not be liable against any damages or problems arising from the use of any options or any consumable products other than those specified as Original Epson Products or Epson Approved Products by Seiko Epson Corporation.

Trademarks

EPSON is a registered trademark of Seiko Epson Corporation.

Exceed Your Vision is registered trademark or trademark of Seiko Epson Corporation.

IOS® is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

The Bluetooth® word mark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by Seiko Epson Corporation is under license.

All other trademarks are the property of their respective owners and used for identification purpose only.

© Seiko Epson Corporation 2015 - 2017. All rights reserved.

Restriction of Use

When this product is used for applications requiring high reliability/safety, such as transportation devices related to aviation, rail, marine, automotive, etc.; disaster prevention devices; various safety devices, etc.; or functional/precision devices, etc., you should use this product only after giving consideration to including fail-safes and redundancies into your design to maintain safety and total system reliability. Because this product was not intended for use in applications requiring extremely high reliability/safety, such as aerospace equipment, main communication equipment, nuclear power control equipment, or medical equipment related to direct medical care, etc., please make your own judgment on this product's suitability after a full evaluation.

Contents

■ Restriction of Use	3
■ Contents.....	4

Migration Overview.....6

■ Migration Types.....	6
------------------------	---

Migration from ePOS-Print SDK.....8

■ Migration with ePOS-Print SDK-compatible APIs	8
Migration procedure.....	8
SDK file replacement.....	8
Application development with ePOS-Print SDK-compatible APIs.....	8
■ Migration with the Epson ePOS SDK APIs	9
Migration procedure.....	9
SDK file replacement.....	10
Adding dynamic libraries	10
Changing import definitions	10
Changing classes.....	10
Establishing and cutting the communication with the printer.....	11
Printing	12
Obtaining callback.....	14
Obtaining status	16
Printer search.....	18
Monitoring of the status.....	20
Changing API name.....	22
Changing API parameters	24

Migration from ePOS-Device SDK.....25

■ Migration with ePOS-Device SDK-compatible APIs	25
Migration procedure.....	25
SDK file replacement.....	25
Application development with ePOS-Device SDK-compatible APIs.....	25
■ Migration with the Epson ePOS SDK APIs	26
Migration procedure.....	26
SDK file replacement.....	27
Adding frameworks.....	27
Changing import definitions	27
Changing classes.....	28
Establishing and cutting the communication with the device	29
Reconnection notifications	31
Printing	33

Forced sending	35
Obtaining callback.....	37
Obtaining status	39
Monitoring of the status.....	42
Changing API names.....	44
Changing API parameters	48

Appendix.....49

■ ePOS-Print SDK-compatible API..... 49

A list of support APIs for each printer model.....	49
TM-m10.....	51
TM-m30.....	53
TM-T60	55
TM-T88VI.....	56
TM-T88VI-iHUB	58

Migration Overview

This manual explains how to modify applications created with following development tools in order to run these applications in Epson ePOS SDK for iOS ("Epson ePOS SDK").

- ePOS-Print SDK for iOS ("ePOS-Print SDK")
- ePOS-Device SDK for iOS ("ePOS-Device SDK")

ePOS-Print SDK and ePOS-Device SDK will no longer be optimized for new products and new functions. Refer to this document to perform migration to ePOS SDK-compatible application.

Migration Types

The following 2 methods are available for migration to Epson ePOS SDK-compatible application.

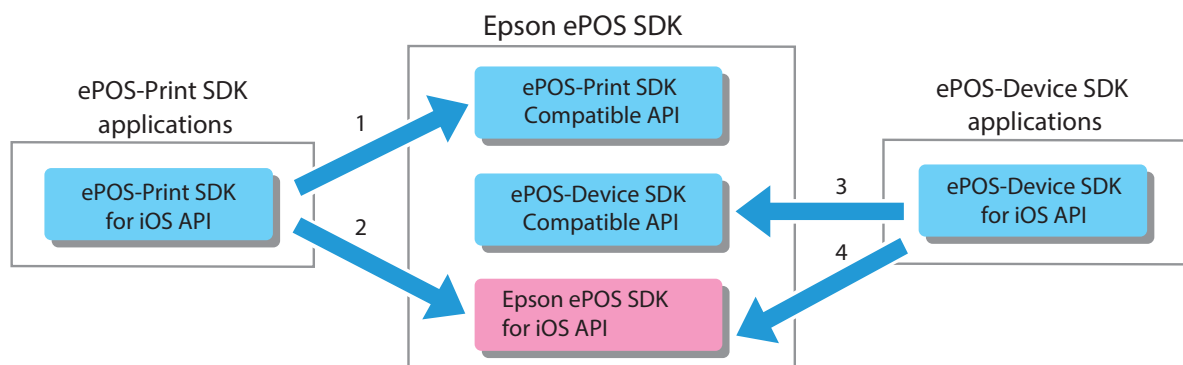
- ❑ Migration with compatible APIs for Epson ePOS SDK

Epson ePOS SDK includes both ePOS-Print SDK-compatible APIs and ePOS-Device SDK-compatible APIs. You can perform migration to an Epson ePOS SDK-compatible application by updating and building the configuration file without modifying the existing application program. If you want to use printing and other basic functions, you can also apply applications to new TM printers.

- ❑ Migration with Epson ePOS SDK APIs

You can perform migration of an existing application to a Epson ePOS SDK-compatible application by turning it into a program that uses Epson ePOS SDK APIs. Many program modifications are necessary, but applications can be applied to new models and functions of TM printers and peripheral devices.

Migration Types



- 1: Migration from ePOS-Print SDK with ePOS-Print SDK-compatible APIs of Epson ePOS SDK
- 2: Migration from ePOS-Print SDK with the APIs of Epson ePOS SDK
- 3: Migration from ePOS-Device SDK with ePOS-Device SDK-compatible APIs of Epson ePOS SDK
- 4: Migration from ePOS-Device SDK with the APIs of Epson ePOS SDK

Compatibility with new products and functions

New products and functions	ePOS-Print SDK ePOS-Device SDK	Epson ePOS SDK	
			ePOS-Print SDK-compatible API ePOS-Device SDK-compatible API
Epson TM printers and peripheral devices	-	✓	✓
New functions of Epson TM printers and peripheral devices	-	✓	- *
New iOS version	-	✓	✓
New tablets and smartphone models	-	✓	✓

✓ : Compatible -: Incompatible

* Not compatible with the new functions that require new APIs or API modifications.

Migration from ePOS-Print SDK

This chapter explains the method to perform migration of the applications that use ePOS-Print SDK to the applications that are compatible with Epson ePOS SDK.

Migration with ePOS-Print SDK-compatible APIs

You can perform migration to an Epson ePOS SDK-compatible application by replacing the configuration file without modifying the existing application program.

Migration procedure

The migration procedure is described below.

Procedure		Description
1	SDK file replacement	Library file replacement Refer to " SDK file replacement ".
2	Application building	Building a project for the application with SDK files being replaced

This completes the migration with the ePOS-Print SDK-compatible API.

SDK file replacement

Replace the following files that are included in the application project with Epson ePOS SDK files.

Type	ePOS-Print SDK	ePOS-Print SDK-compatible API
Library	libeposprint.a	libepos2.a

Application development with ePOS-Print SDK-compatible APIs

Refer to the following manuals for information that is necessary for the development and maintenance of the applications that use ePOS-Print SDK-compatible APIs.

❑ Specifications for ePOS-Print SDK-compatible APIs

"ePOS-Print SDK for iOS User's Manual"

The specifications for ePOS-Print SDK-compatible APIs are the same as the specifications for the ePOS-Print SDK APIs.

❑ Device information and support APIs for the new Epson TM printer models

The [Appendix](#) of "Epson ePOS SDK for iOS Migration Guide" (this document)

Migration with the Epson ePOS SDK APIs

You can perform migration to an Epson ePOS SDK-compatible application by modifying the existing application program. Many program modifications are necessary, but applications can be applied to new models and functions of TM printers.

Migration procedure

The migration procedure is described below.

Procedure		Description
1	SDK file replacement	Header file and library file replacement Refer to " SDK file replacement ".
2	Adding dynamic libraries	Adding dynamic libraries to the application project Refer to " Adding dynamic libraries ".
3	Changing import definitions	Changing the import definitions of the Objective-C header Refer to " Changing import definitions ".
4	Changing classes	Changing the ePOS-Print SDK classes to the Epson ePOS SDK classes Refer to " Changing classes ".
5	Changing APIs	<p>Modifying the program or changing the ePOS-Print SDK APIs that have different specifications from those of Epson ePOS SDK</p> <p>The changes are described below.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Modifying the program to enable specific functions Modify the functions listed below. <ul style="list-style-type: none"> • Establishing and cutting the communication with the printer Refer to "Establishing and cutting the communication with the printer". • Printing Refer to "Printing". • Obtaining callback Refer to "Obtaining callback". • Obtaining status Refer to "Obtaining status". • Printer search Refer to "Printer search". • Status monitoring Refer to "Monitoring of the status". <input type="checkbox"/> Changing API name API names to be changed (You may need to change parameters too) Refer to "Changing API name". <input type="checkbox"/> Changing API parameters API names do not change, but parameter changes are necessary. Refer to "Changing API parameters".
6	Application building	Building a project for the modified application.

This completes the migration with the Epson ePOS SDK API.

SDK file replacement

Replace the following files that are included in the application project with Epson ePOS SDK files.

Type	ePOS-Print SDK	Epson ePOS SDK
Header file	ePOS-Print.h	ePOS2.h
	ePOSEasySelect.h	ePOSEasySelect.h *
Library	libeposprint.a	libepos2.a
	libeposeasyselect.a	libeposeasyselect.a *

* The file names do not change. Use the files included in the Epson ePOS SDK package.

Adding dynamic libraries

Incorporate the following dynamic library file in the application project.

- libxml2.2.*

Changing import definitions

Change the import definition of the Objective-C header that is included in the *.m source file of an application.

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

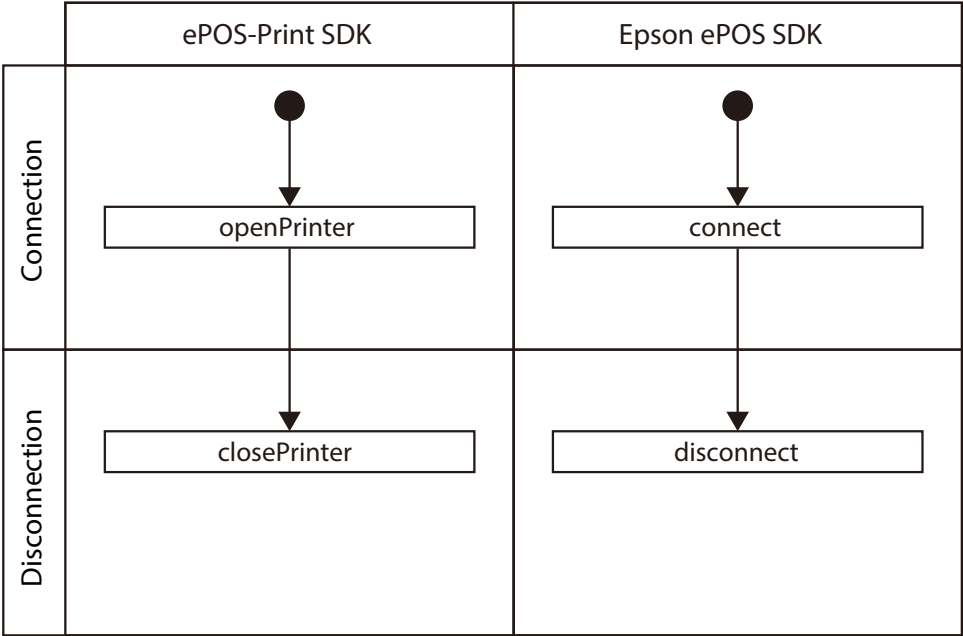
Changing classes

Change the classes of ePOS-Print SDK used in the application project to the classes of Epson ePOS SDK.

Type	ePOS-Print SDK	Epson ePOS SDK
Printing functions	EposBuilder class	Epos2Printer class
	EposPrint class	
Printer search	EpsonIoFinder class	Epos2Discovery class
Log output function	EposLog class	Epos2Log class
Bluetooth® connection	EposBluetoothConnection class	Epos2BluetoothConnection class

Establishing and cutting the communication with the printer

Execution procedure differences



Program differences

❑ 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];
    ...processing...

    errorStatus = [printer closePrinter];
}
```

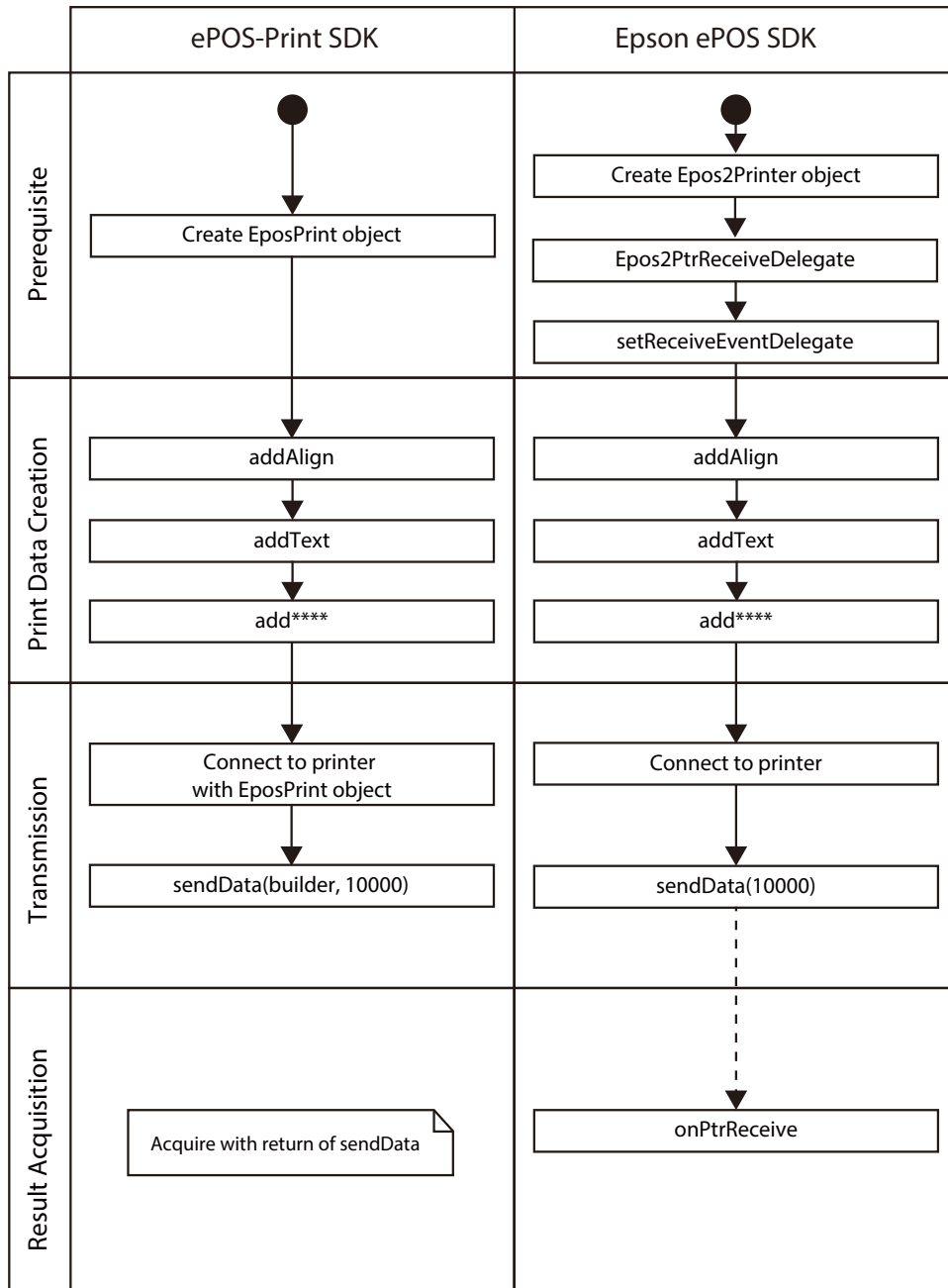
❑ Epson ePOS SDK

```
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];
    ...processing...
    errorStatus = [printer disconnect];
}
```

Printing

In ePOS-Print SDK, the printing results were obtained with the return value of the printing data transmission processing. whereas in Epson ePOS SDK, the printing results are obtained through callback.

Execution procedure differences



Callback: ---->

Program differences

❑ 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_DEVTTYPE_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_;
}

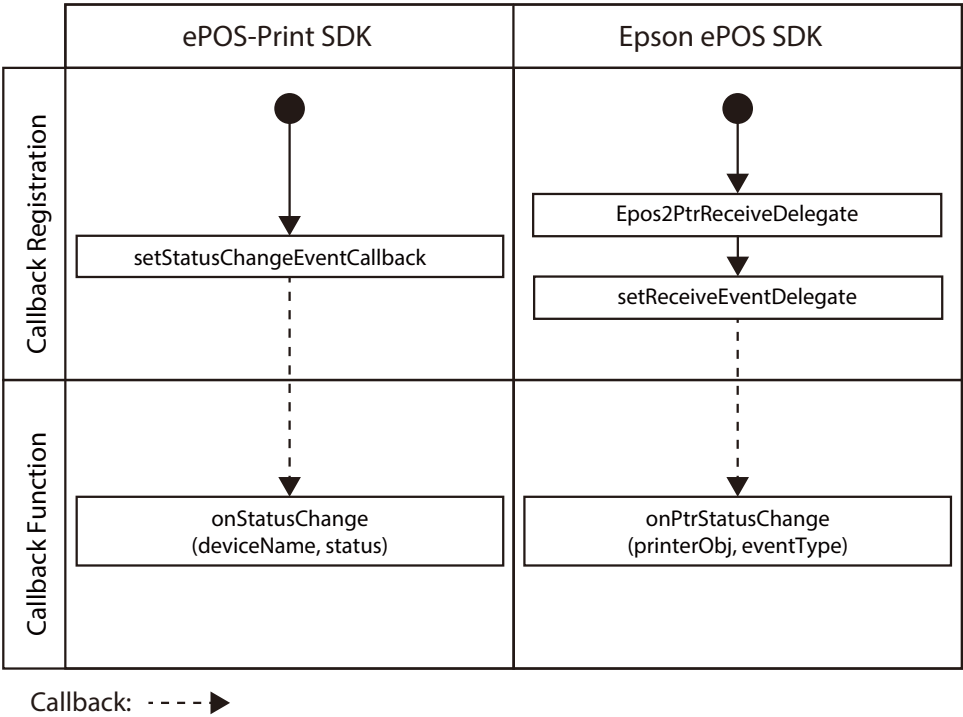
- (void) openPrinter
{
    [printer_ setReceiveEventDelegate:self];
    int errorStatus = [printer_ addText:@"ABCDE"];
    ...connection...
    errorStatus = [printer_ sendData:EPOS2_PARAM_DEFAULT];
}

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

Obtaining callback

In ePOS-Print SDK, a selector was used for callback processing, whereas in Epson ePOS SDK, a protocol is used for callback processing.

Execution procedure differences



Program differences

❑ ePOS-Print SDK

```
- (void)onStatusChange:(NSString *)deviceName Status:(NSNumber *)status
{
...connection...
}
- (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];
    }
}
```

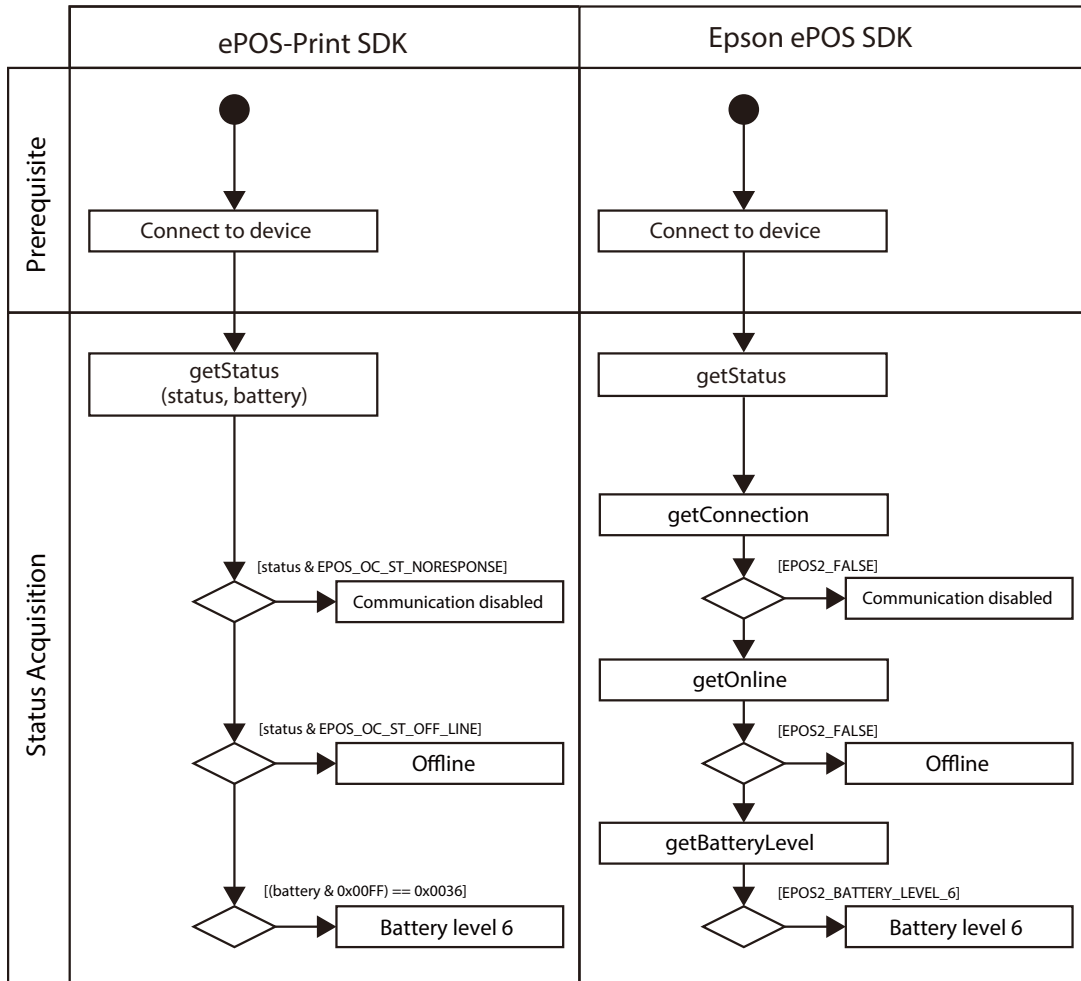
❑ Epson ePOS SDK

```
@interface Sample() <Epos2PtrStatusChangeDelegate>
{
    Epos2Printer *printer_;
}
- (void) openPrinter
{
    ...connection...
    [printer_ setStatusChangeEventDelegate:self];
    [printer_ startMonitor];
}
- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ...processing...
}
```

Obtaining status

In ePOS-Print SDK, the combinations of multiple printer statuses were obtained with the return value, whereas in Epson ePOS SDK, each status is obtained from PrinterStatusInfo-type properties.

Execution procedure differences



Program differences

❑ 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_;
}

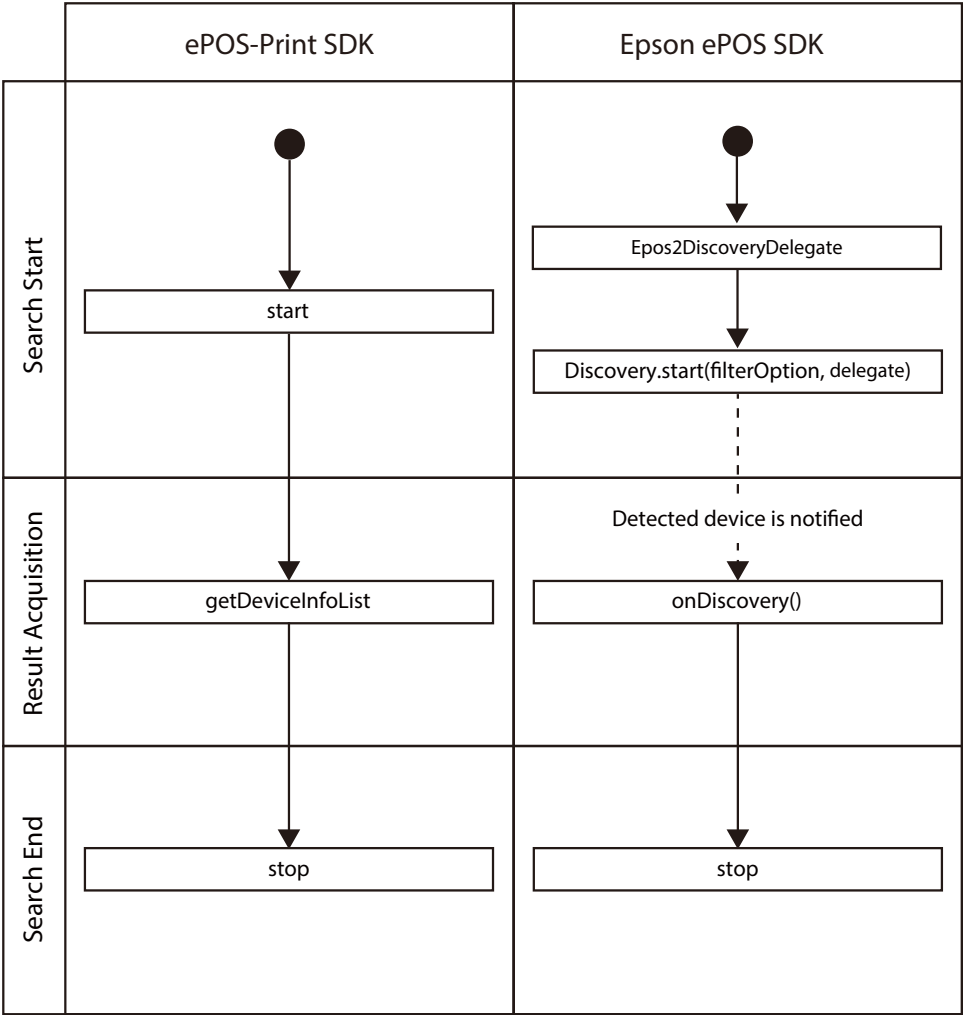
- (void) getStatus
{
    ...connection...
    Epos2PrinterStatusInfo *status = [printer_ getStatus];

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

Printer search

In ePOS-Print SDK, the printer search results were obtained through the API, whereas in Epson ePOS SDK the search results are obtained with the callback method after the filtering settings have been configured.

Execution procedure differences



Callback: - - - - ➡

Program differences

❑ 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];

    ...searching...

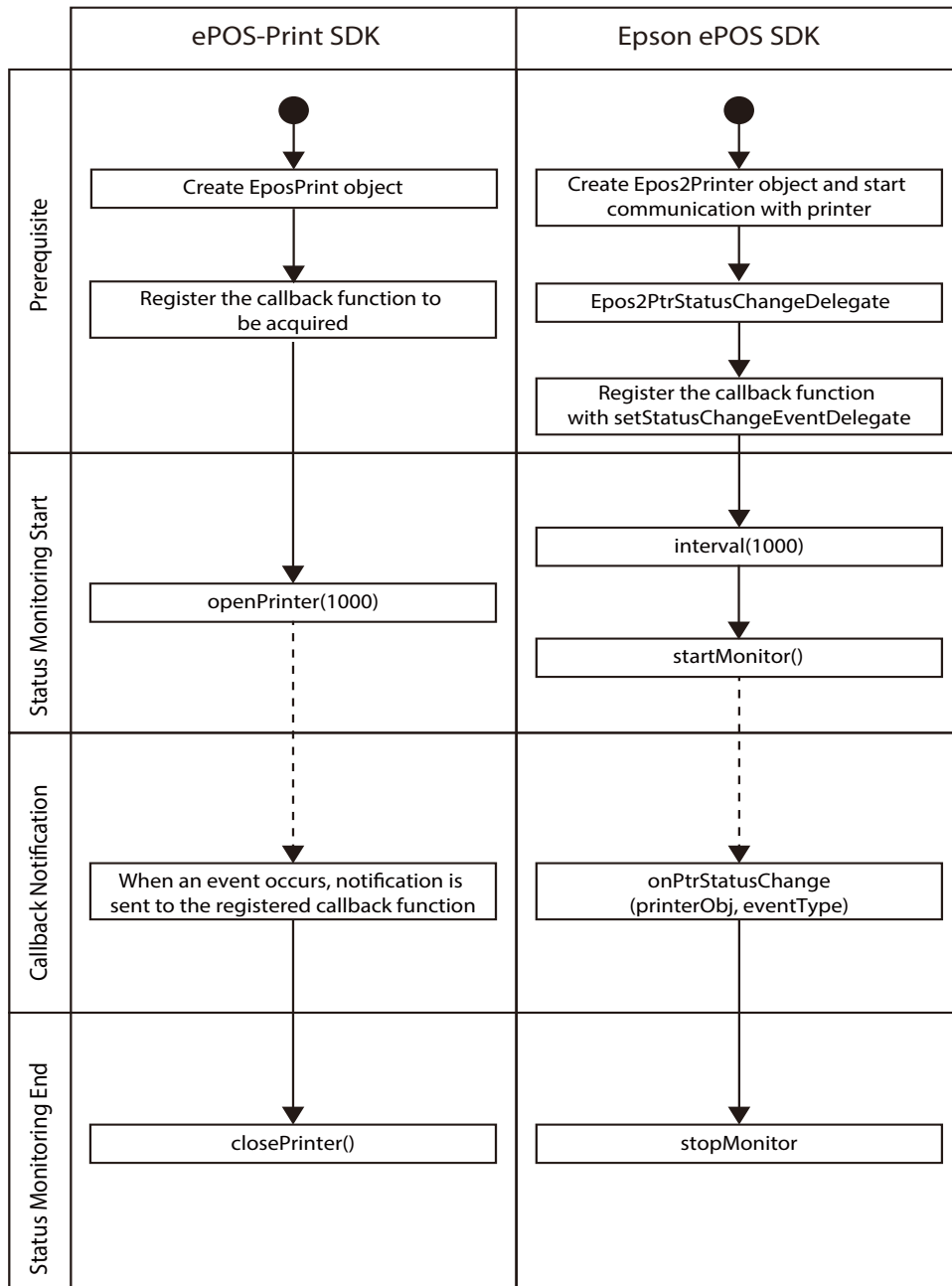
    [Epos2Discovery stop];
}

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

Monitoring of the status

In ePOS-Print SDK, the communication with the printer and status monitoring started at the same time, whereas in Epson ePOS SDK, status monitoring starts after the communication with the printer has been established.

Execution procedure differences



Callback: ---->

Program differences

❑ ePOS-Print SDK

```
- (void)onStatusChange:(NSString *)deviceName Status:(NSNumber *)status
{
    ...processing...
}

- (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];
        ...connection...

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

❑ Epson ePOS SDK

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

- (void) openPrinter
{
    [printer_ setStatusChangeEventDelegate:self];
    ...connection...
    [printer_ startMonitor];

    [printer_ stopMonitor];
}

- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ...processing...
}
```

Changing API name

When performing migration from ePOS-Print SDK to Epson ePOS SDK, change the API name as follows.

- Change the initial letters of the argument keywords from upper case to lower case. The example below is for addImage.

ePOS-Print SDK
<code>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];</code>
Epson ePOS SDK
<code>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];</code>

- Rename the APIs. The APIs that need to be renamed are listed in the table below. In some cases, multiple APIs are bundled in one API, or one API is divided into several APIs. Among the APIs in the table below, there are APIs where specifications other than the name have changed.

To see what has changed, compare APIs in the "ePOS-Print SDK for iOS User's Manual" and "Epson ePOS SDK for iOS User's Manual".

A list of API names to be changed

Function	ePOS-Print SDK	Epson ePOS SDK
Class initialization	initWithPrinterModel	initWithPrinterSeries
	init	
Adding the text line space settings to the instruction buffer	addTextLineSpace	addLineSpace
Adding the character size settings to the instruction buffer	addTextDouble	addTextSize
Adding the text printing position settings to the instruction buffer	addTextPosition	addHPosition
Starting communication	openPrinter	connect
Starting status monitoring		startMonitor
Terminating communication	closePrinter	disconnect
Terminating status monitoring		stopMonitor
Registering the notification destination of printer statuses	setStatusChangeEventCallback	setStatusChangeEventDelegate
Registering the notification destination of online events	setOnlineEventCallback	
Registering the notification destination of offline events	setOfflineEventCallback	
Registering the notification destination of power-off events	setPowerOffEventCallback	
Registering the notification destination of cover-closed events	setCoverOkEventCallback	
Registering the notification destination of cover-open events	setCoverOpenEventCallback	

Function	ePOS-Print SDK	Epson ePOS SDK
Registering the notification destination of paper OK events	setPaperOkEventCallback	setStatusChangeEventDelegate
Registering the notification destination of paper near-end events	setPaperNearEndEventCallback	
Registering the notification destination of paper end events	setPaperEndEventCallback	
Registering the notification destination of drawer-closed events	setDrawerClosedEventCallback	
Registering the notification destination of drawer-open events	setDrawerOpenEventCallback	
Registering the notification destination of battery-low events	setBatteryLowEventCallback	
Registering the notification destination of battery level OK events	setBatteryOkEventCallback	
Registering the notification destination of battery statuses	setBatteryStatusChangeEventCallback	
Obtaining printer search results	start	start
	getDeviceInfoList	

Changing API parameters

When performing migration from ePOS-Print SDK to Epson ePOS SDK, change the parameters as follows.

- Change the parameters for Printer Easy Select to parameters for Epson ePOS SDK as shown in the table below.

ePOS-Print SDK	Epson ePOS SDK
EPOS_OC_DEVTYPE_TCP	EPOS_EASY_SELECT_DEVTYPE_TCP
EPOS_OC_DEVTYPE_BLUETOOTH	EPOS_EASY_SELECT_DEVTYPE_BLUETOOTH

- Change the parameter naming rule for ePOS-Print SDK to the rule for Epson ePOS SDK.

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

- Add or merge parameters, and add or delete setting values. The APIs where the parameters need to be changed are listed below.

To see what has changed, compare APIs in the "ePOS-Print SDK for iOS User's Manual" and "Epson ePOS SDK for iOS User's Manual".

A list of APIs with parameters to be changed

API	Parameter change
addTextAlign	align settings added
addTextRotate	rotate settings added
addTextLang	lang settings added
addTextFont	font settings added
addTextSmooth	smooth settings added
addTextSize	width/height settings added
addTextStyle	reverse/ ul/ em/ color settings added
addImage	compress settings added
addBarcode	hri/ font settings added
addPageDirection	direction settings added
addPagePosition	x/ y settings added
addSound	pattern/ repeat settings added The pattern setting name changed
sendData	Change to timeout only
getStatus	No parameter, and value obtained with the return value
createQR	The deviceType setting name changed

Migration from ePOS-Device SDK

This chapter explains the method to perform migration of the applications that use ePOS-Device SDK to the applications that are compatible with Epson ePOS SDK.

Migration with ePOS-Device SDK-compatible APIs

You can perform migration to an Epson ePOS SDK-compatible application by replacing the configuration file without modifying the existing application program.

Migration procedure

The migration procedure is described below.

Procedure		Description
1	SDK file replacement	Library file replacement Refer to " SDK file replacement ".
2	Application building	Building a project for the application with SDK files being replaced

This completes the migration with ePOS-Device SDK-compatible APIs.

SDK file replacement

Replace the following files that are included in the application project with Epson ePOS SDK files.

Type	ePOS-Device SDK	ePOS-Device SDK-compatible API
Library	libeposdevice.a	libepos2.a

Application development with ePOS-Device SDK-compatible APIs

Refer to the following manuals for information that is necessary for the development and maintenance of the applications that use ePOS-Device SDK-compatible APIs.

Specifications for ePOS-Device SDK-compatible APIs: "ePOS-Device SDK for iOS User's Manual"

Migration with the Epson ePOS SDK APIs

You can perform migration to an Epson ePOS SDK-compatible application by modifying the existing application program. Many program modifications are necessary, but applications can be applied to new models and functions of TM printers or peripheral devices.

Migration procedure

The migration procedure is described below.

Procedure		Description
1	SDK file replacement	Header file and library file replacement Refer to " SDK file replacement ".
2	Adding frameworks	Adding frameworks to the application project Refer to " Adding frameworks ".
3	Changing import definitions	Changing the import definitions of the Objective-C header Refer to " Changing import definitions ".
4	Changing classes	Changing the ePOS-Device SDK classes to the Epson ePOS SDK classes Refer to " Changing classes ".
5	Changing APIs	<p>Modifying the program or changing the ePOS-Device SDK APIs that have different specifications from those of Epson ePOS SDK</p> <p>The changes are described below.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Modifying the program to enable specific functions Modify the functions listed below <ul style="list-style-type: none"> • Establishing and cutting the communication with the device Refer to "Establishing and cutting the communication with the device". • Reconnection notifications Refer to "Reconnection notifications". • Printing Refer to "Printing". • Forced sending Refer to "Forced sending". • Obtaining callback Refer to "Obtaining callback". • Obtaining status Refer to "Obtaining status". • Status monitoring Refer to "Monitoring of the status". <input type="checkbox"/> Changing API names API names to be changed (You may need to change parameters too) Refer to "Changing API names". <input type="checkbox"/> Changing API parameters API names do not change, but parameter changes are necessary. Refer to "Changing API parameters".
6	Application building	Building a project for the modified application.

This completes the migration with the Epson ePOS SDK APIs.

SDK file replacement

Replace the following files that are included in the application project with Epson ePOS SDK files.

Type	ePOS-Device SDK	Epson ePOS SDK
Header file	ePOS-Device.h	ePOS2.h
Library	libeposdevice.a	libepos2.a

Adding frameworks

Incorporate the following framework file in the application project.

- ExternalAccessory.framework

Changing import definitions

Change the import definition of the Objective-C header that is included in the *.m source file of an application.

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

Changing classes

Change the classes of ePOS-Device SDK used in the application project to the classes of Epson ePOS SDK.

ePOS-Device SDK class names to be deleted

Class name	Details
EposDevice class	<p>The following functions of the EposDevice class are assigned to the APIs of each class.</p> <ul style="list-style-type: none"> • Establishing the communication path • Disconnecting the communication path • Obtaining the establishment status of the communication path • Obtaining administrator information • Obtaining installation location information • Registering the callback method for the reconnection processing start event • Registering the callback method for the reconnection end event • Registering the callback method for the network disconnection event
EposCommBoxManager class	This is integrated in the CommBox class

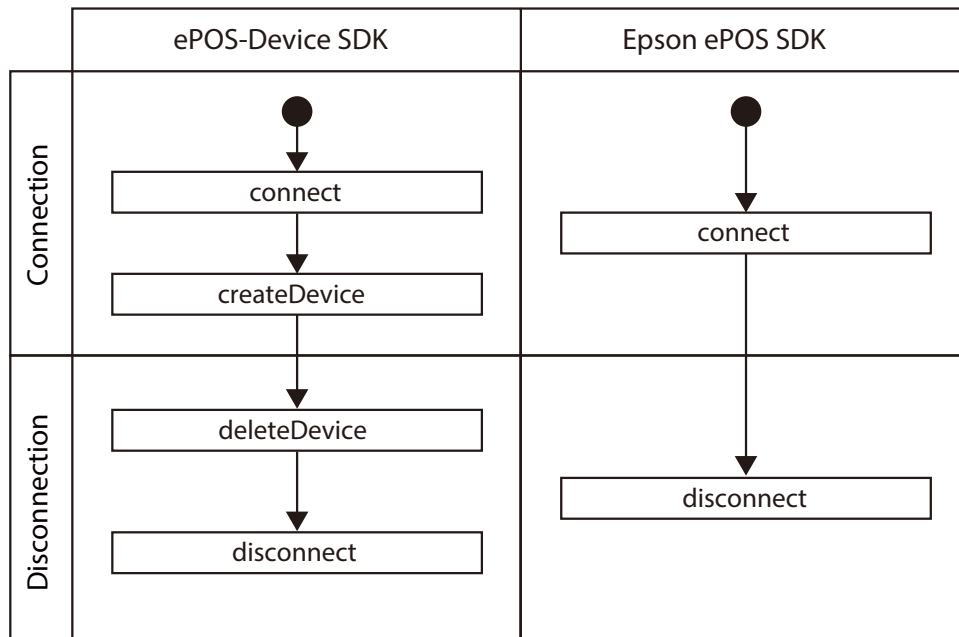
Class names to be changed

Type	ePOS-Device SDK	Epson ePOS SDK
Printing function	EposPrinter class	Epos2Printer class
Device control	EposDisplay class	Epos2LineDisplay class
	EposKeyboard class	Epos2Keyboard class
	EposScanner class	Epos2BarcodeScanner class
Command trans- mission	EposSimpleSerial class	Epos2SimpleSerial class
Communication box	EposCommBoxManager class	Epos2CommBox class
	EposCommBox class	
Log function	EposDeviceLog class	Epos2Log class

Establishing and cutting the communication with the device

In ePOS-Device SDK, the communication with each device used to start after the connection with ePOS-Device Service had been established, whereas in Epson ePOS SDK the communication starts for each individual device. Also, in ePOS-Device SDK, the connection with ePOS-Device Service used to terminate after the communication with the device had ended, whereas in Epson ePOS SDK the communication terminates for each individual device.

Execution procedure differences



Program differences

❑ 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" Callback:@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

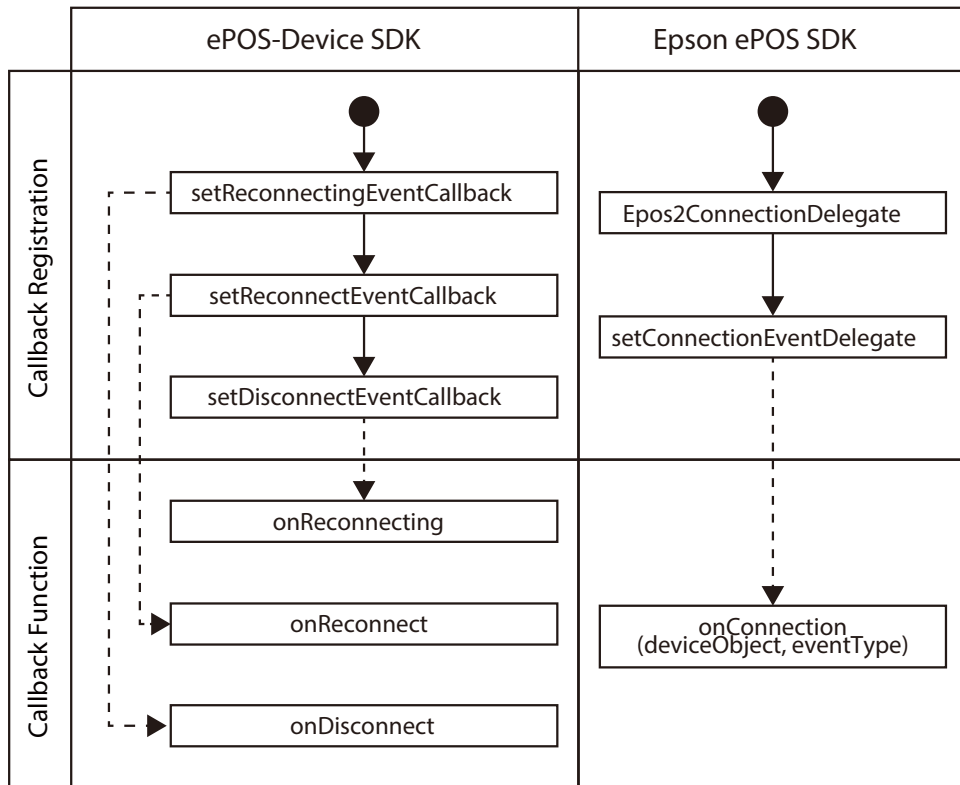
```
Epos2Printer printer = [[Epos2Printer alloc] initWithPrinterSeries:EPOS2_TM_T88
lang:EPOS2_MODEL_ANK];

if ( printer != nil) {
    int errorStatus = EPOS2_SUCCESS;
    errorStatus = [printer connect:@"TCP:192.168.192.168" timeout:EPOS2_PARAM_DEFAULT];
    ...processing...
    errorStatus = [printer disconnect];
}
```

Reconnection notifications

In ePOS-Device SDK, the registration was carried out through the API for each notification type, whereas in Epson ePOS SDK the APIs for the registration of notification destination are merged in one API and the processing is carried out for each notification type using the notification destination method.

Execution procedure differences



Callback: - - - - ➡

Program differences

❑ 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];
        ...connection...
    }
}

- (void) onReconnecting:(NSString *)ipAddress
{
    ...starting reconnection...
}

- (void) onReconnect:(NSString *)ipAddress
{
    ...reconnection end...
}

- (void) onDisconnect:(NSString *)ipAddress
{
    ...reconnection failed...
}
}
```

❑ 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];
        ...connection...
    }
}

- (void) onConnection:(id)deviceObj eventType:(int)eventType
{
    if(eventType == EPOS2_EVENT_RECONNECTING) {
        ...starting reconnection...
    }
    if(eventType == EPOS2_EVENT_RECONNECT) {
        ...reconnection end...
    }
    if(eventType == EPOS2_EVENT_DISCONNECT) {
        ...reconnection failed...
    }
}
}
```

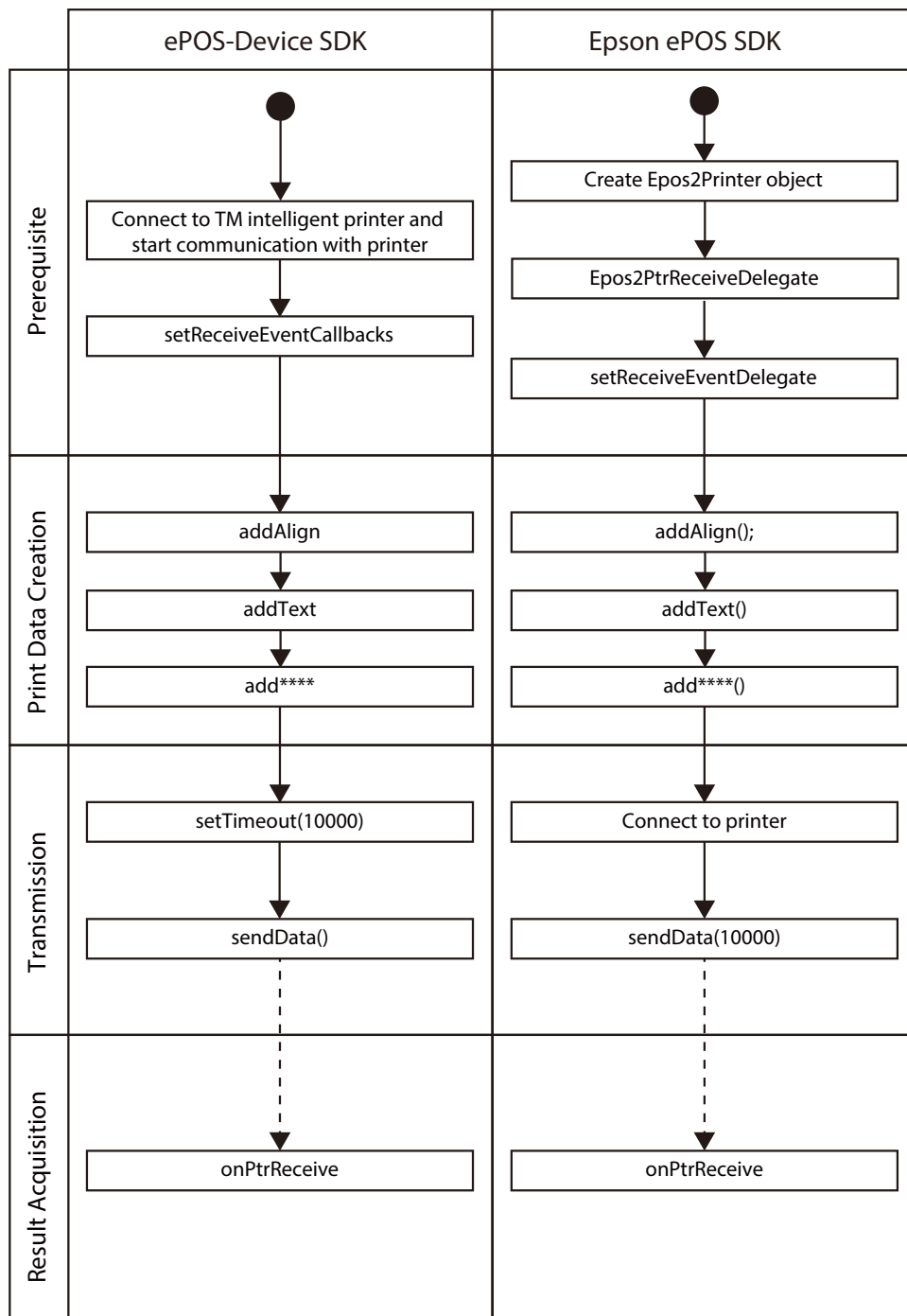

Printing

In ePOS-Device SDK, the printing data was created after the connection with the printer had been established, whereas in Epson ePOS SDK, the data can be created either before or after the connection with the printer is established.

Printing is possible without any modifications to the execution procedures of the existing programs.

Execution procedure differences

The execution procedure of Epson ePOS SDK is the procedure for creating printing data before establishing a connection with the printer.



Callback: ---->

Program differences

❑ ePOS-Device SDK

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

- (void) openPrinter
{
    ...connection...

    [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
{
    ...processing...
}
```

❑ Epson ePOS SDK

The following program creates printing data before establishing a connection with the printer.

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

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

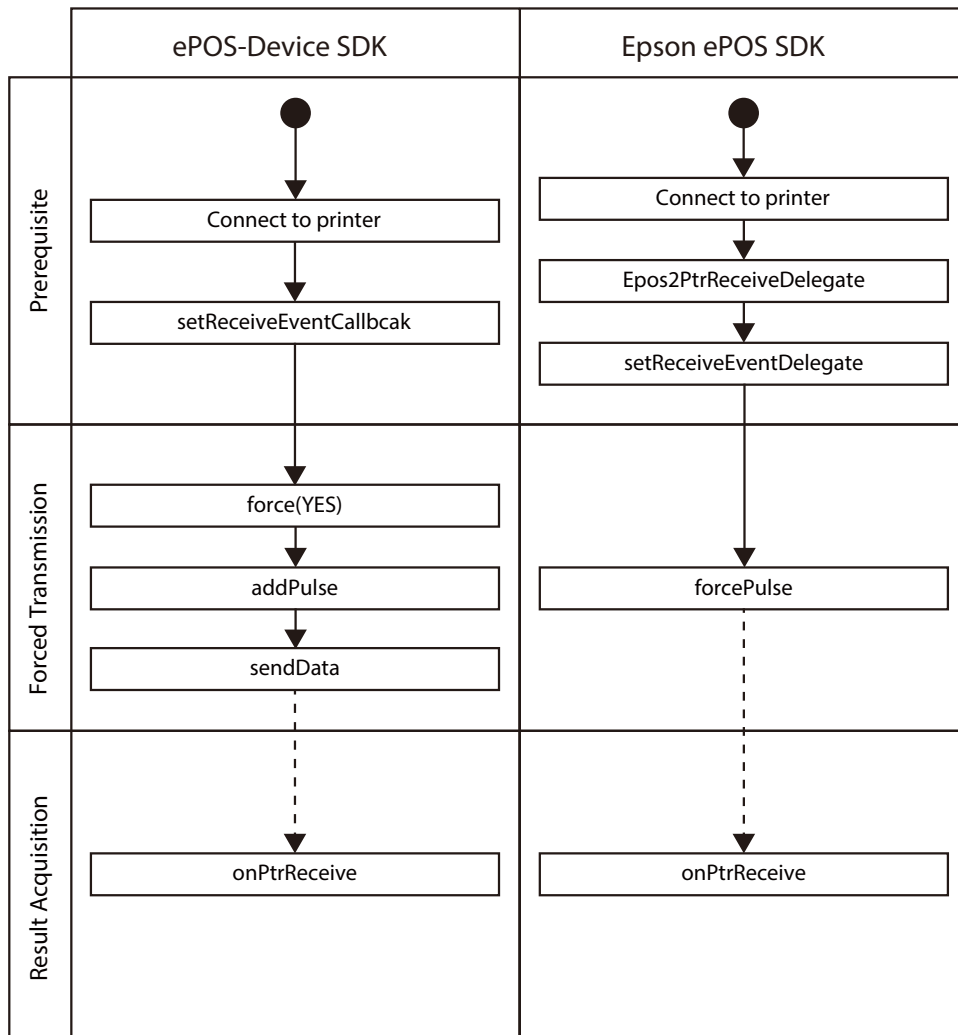
- (void) onPtrReceive:(Epos2Printer *)printerObj code:(int)code
status:(Epos2PrinterStatusInfo *)status printJobId:(NSString *)printJobId
{
    ...processing...
}
```

Forced sending

In ePOS-Device SDK, forced sending was carried out with 3 APIs, whereas in Epson ePOS SDK it is carried out with 1 API.

Also, in ePOS-Device SDK, forced sending was only enabled in offline mode, whereas in Epson ePOS SDK it can be carried out either in online or offline mode.

Execution procedure differences



Callback: - - - - ➤

Program differences

❑ ePOS-Device SDK

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

- (void) openPrinter
{
    ...connection...

    [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
{
    ...processing...
}
```

❑ Epson ePOS SDK

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

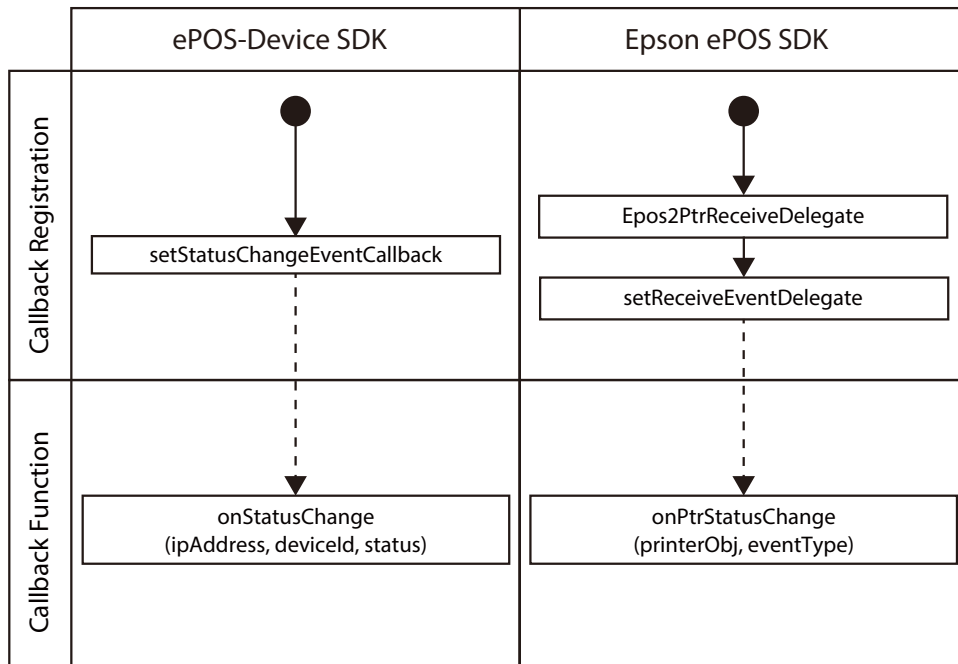
- (void) openPrinter
{
    ...connection...
    [printer_ setReceiveEventDelegate:self];
    int errorStatus = [printer_ forcePulse:EPOS2_PARAM_DEFAULT pulseTime:EPOS2_PULSE_100
                                timeout:EPOS2_PARAM_DEFAULT];
}

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

Obtaining callback

In ePOS-Device SDK, a selector was used for callback processing, whereas in Epson ePOS SDK, a protocol is used for callback processing.

Execution procedure differences



Callback: ----▶

Program differences

❑ ePOS-Device SDK

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

- (void) openPrinter
{
    ...connection...

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

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

❑ Epson ePOS SDK

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

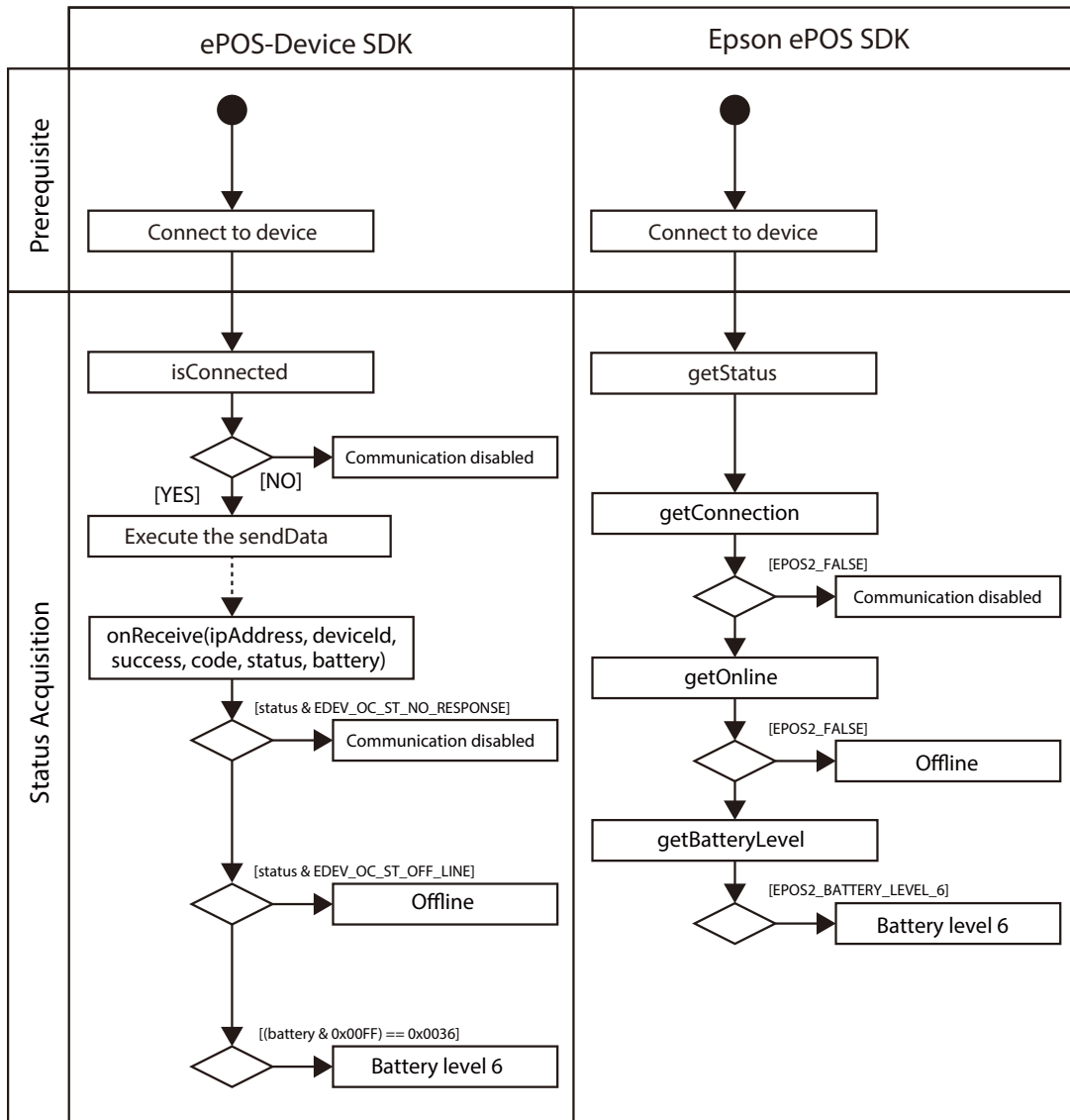
- (void) openPrinter
{
    ...connection...
    [printer_ setStatusChangeEventDelegate:self];
    [printer_ startMonitor];
}

- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ...processing...
}
```

Obtaining status

In ePOS-Device SDK, the combinations of multiple printer statuses were obtained with the return value, whereas in Epson ePOS SDK, each status is obtained from PrinterStatusInfo-type properties.

Execution procedure differences



Callback: ----▶

Program differences

❑ ePOS-Device SDK

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

- (void) getStatus
{
    ...connection...

    if([device_ isConnected] != YES) {
        ...disconnecting...
    }
    [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_;
}

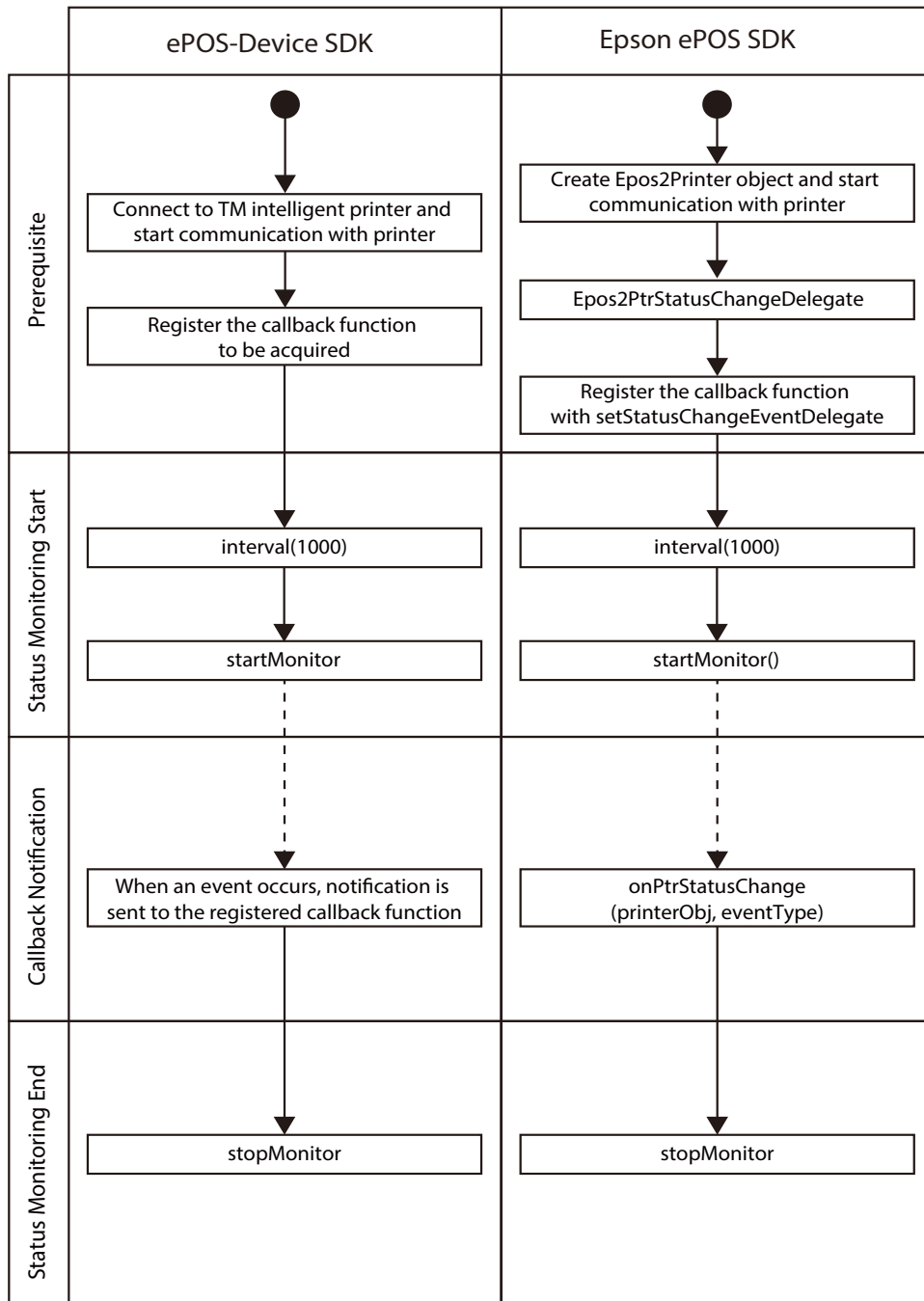
- (void) getStatus
{
    ...connection...
    Epos2PrinterStatusInfo *status = [printer_ getStatus];

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

Monitoring of the status

In ePOS-Device SDK, the registration was carried out through the API for each status type, whereas in Epson ePOS SDK the APIs for the registration of notification destination are merged in one API and processing is selected using the notification destination method.

Execution procedure differences



Callback: ----▶

Program differences

❑ ePOS-Device SDK

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

- (void) openPrinter
{
    ...connection...

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

    [printer_ stopMonitor];
}

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

❑ Epson ePOS SDK

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

- (void) openPrinter
{
    [printer_ setStatusChangeEventDelegate:self];
    ...connection...
    [printer_ startMonitor];

    [printer_ stopMonitor];
}

- (void) onPtrStatusChange:(Epos2Printer *)printerObj eventType:(int)eventType
{
    ...processing...
}
```

Changing API names

When performing migration from ePOS-Device SDK to Epson ePOS SDK, change the API name as follows.

- Change the initial letters of the argument keywords from upper case to lower case. The example below is for addImage.

ePOS-Device SDK
<code>errorStatus = [printer addImage: imageData X: 0 Y: 0 Width: 256 Height: 256 Color: EDEV_OC_PARAM_DEFAULT Mode: EDEV_OC_MODE_MONO];</code>
Epson ePOS SDK
<code>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];</code>

- Rename the APIs. The APIs that need to be renamed are listed in the table below. In some cases, multiple APIs are bundled in one API, one API is divided into several APIs, or some APIs have been deleted. Among the APIs in the table below, there are APIs where specifications other than the name have changed.

To see what has changed, compare APIs in the "ePOS-Device SDK for iOS User's Manual" and "Epson ePOS SDK for iOS User's Manual".

A list of API names to be changed

Class		ePOS-Device SDK	Epson ePOS SDK
Function			
Class - common			
Class initialization		init	initWithPrinterSeries
			initWithDisplaySeries
			init
Obtaining the device object		connect	connect
		createDevice	
Discarding the device object		disconnect	disconnect
		deleteDevice	
Obtaining the present status information		isConnected	getStatus
Registering the notification destination of reconnection processing start event		setReconnectingEventCallback	setConnectionEventDelegate
Registering the notification destination of reconnection end event		setReconnectEventCallback	
Registering the notification destination of network disconnection event		setDisconnectEventCallback	
EposDisplay class			
Adding display area definition to the instruction buffer		createWindow	addCreateWindow
Adding display area settings cancellation to the instruction buffer		destroyWindow	addDestroyWindow
Adding display area switching to the instruction buffer		setCurrentWindow	addSetCurrentWindow

Class		ePOS-Device SDK	Epson ePOS SDK
	Function		
EposDisplay class			
	Adding display area settings deletion to the instruction buffer.	clearWindow	addClearCurrentWindow
	Adding the cursor position to the instruction buffer	setCursorPosition	addSetCursorPosition
	Adding the cursor position in the display area to the instruction buffer	moveCursorPosition	addMoveCursorPosition
	Adding cursor type change to the instruction buffer	setCursorType	addSetCursorType
	Adding the marquee view to the instruction buffer	addMarquee	addMarqueeText
	Adding blinking display information to the instruction buffer	setBlink	addSetBlink
	Adding display brightness information to the instruction buffer	setBrightness	addSetBrightness
	Adding time displayed on the clock to the instruction buffer	ShowClock	addShowClock
	Resetting the customer display	reset	addInitialize
	Registering the notification destination of control result reception events	SetReceiveEventCallback	setReceiveEventDelegate
EposKeyboard class			
	Registering the notification destination of pressed key detection events	setKeyPressEventCallback	setKeyPressEventDelegate
	Registering the notification destination of character string detection events	setStringEventCallback	setReadStringEventDelegate
EposPrinter class			
	Adding the text line space settings to the instruction buffer	addTextLineSpace	addLineSpace
	Adding the character size settings to the instruction buffer	addTextSize	addTextSize
		addTextDouble	
	Adding the text printing position settings to the instruction buffer	addTextPosition	addHPosition
	Adding the vertical printing start position settings to the instruction buffer	addTextVPosition	addPagePosition
	Adding line feeding to the instruction buffer	addFeedLine	addFeedLine
		addFeed	
	Recovery from a recoverable error	recover	forceRecover
	Adding the error recovery tag	addRecovery	
	Forced sending of the reset command to the printer	reset	forceReset
	Adding the reset tag of the printer	addReset	
	Obtaining printing results	getPrintJobStatus	requestPrintJobStatus
	Obtaining the present status	setOnlineEventCallback	getStatus
	Setting the processing method for the halftone of the raster image	halftone properties	addImage
	Setting the correction value for the brightness of the raster image	brightness properties	

Class		ePOS-Device SDK	Epson ePOS SDK
	Function		
EposPrinter class			
Forced sending		force properties	forceRecover
			forcePulse
			forceStopSound
			forceCommand
Setting the transmission timeout time		timeout properties	deleted
Status of the drawer signal line		drawerOpenLevel properties	deleted
Registering the notification destination of printer statuses		setStatusChangeEventCallback	setStatusChangeEventDelegate
Registering the notification destination of online events		setOnlineEventCallback	
Registering the notification destination of offline events		setOfflineEventCallback	
Registering the notification destination of power-off events		setPowerOffEventCallback	
Registering the notification destination of cover-closed events		setCoverOkEventCallback	
Registering the notification destination of cover-open events		setCoverOpenEventCallback	
Registering the notification destination of paper presence events		setPaperOkEventCallback	
Registering the notification destination of paper near-end events		setPaperNearEndEventCallback	
Registering the notification destination of paper end events		setPaperEndEventCallback	
Registering the notification destination of drawer-closed events		setDrawerClosedEventCallback	
Registering the notification destination of drawer-open events		setDrawerOpenEventCallback	
Registering the notification destination of battery-low events		setBatteryLowEventCallback	
Registering the notification destination of battery level OK events		setBatteryOkEventCallback	
Registering the notification destination of battery statuses		setBatteryStatusChangeEvent-Callback	
Registering the notification destination of response document reception events		setReceiveEventCallback	setReceiveEventDelegate
EposScanner class			
Registering the notification destination of barcode data input events		setDataEventCallback	setScanEventDelegate
EposSimpleSerial class			
Registering the notification destination of reception events from the device		setCommandReplyEventCallback	setReceiveEventDelegate

Class		ePOS-Device SDK	Epson ePOS SDK
	Function		
EposCommBox class			
	Creating a communication box	openCommBox	connect
	Discarding the communication box	closeCommBox	disconnect
	Sending a message to the communication box	sendData	sendMessage
	Registering the notification destination for receiving a message in the communication box	setReceiveEventCallback	setReceiveEventDelegate

Changing API parameters

When performing migration from ePOS-Device SDK to Epson ePOS SDK, change the parameters as follows.

- Change the parameter naming rule for ePOS-Device SDK to the rule for Epson ePOS SDK.

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

- Add or merge parameters, and add or delete setting values. The APIs where the parameters need to be changed are listed below.

To see what has changed, compare APIs in the "ePOS-Device SDK for iOS User's Manual" and "Epson ePOS SDK for iOS User's Manual".

A list of APIs with parameters to be changed

Class	Parameter change	
	API	
EposDisplay class		
	addText	lang settings added
	addReverseText	lang settings added
	setReceiveEventListener	code value (for callback method) added
EposPrinter class		
	sendData	Change to timeout only
	addTextAlign	align settings added
	addTextRotate	rotate settings added
	addTextLang	lang settings added
	addTextFont	font settings added
	addTextSmooth	smooth settings added
	addTextSize	width/height settings added
	addTextStyle	reverse/ ul/ em/ color settings added
	addImage	halftone/ brightness parameters added compress settings added
	addBarcode	hri/ font/ width/ height settings added
	addSymbol	level/ width/ height/ size settings added
	addPageDirection	direction settings added
	addPagePosition	x/ y settings added
	setReceiveEventListener	Callback method status/ battery parameters merged code value (for callback method) deleted
	interval properties	Settings added
EposCommBox class		
	getCommHistory	code value (for callback method) deleted

Appendix

ePOS-Print SDK-compatible API

A list of support APIs for each printer model

The following table lists the support APIs available for each model of printer.

The symbols used in the table represent the following:

- ✓: Supported
- -: Not supported.

API	TM-m10	TM-m30	TM-T60	TM-T88VI	TM-T88VI-iHUB
addTextAlign	✓	✓	✓	✓	✓
addTextLineSpace	✓	✓	✓	✓	✓
addTextRotate	✓	✓	✓	✓	✓
addText	✓	✓	✓	✓	✓
addTextLang	✓	✓	✓	✓	✓
addTextFont	✓	✓	✓	✓	✓
addTextSmooth	✓	✓	✓	✓	✓
addTextDouble	✓	✓	✓	✓	✓
addTextSize	✓	✓	✓	✓	✓
addTextStyle	✓	✓	✓	✓	✓
addTextPosition	✓	✓	✓	✓	✓
addFeedUnit	✓	✓	✓	✓	✓
addFeedLine	✓	✓	✓	✓	✓
addImage	✓	✓	✓	✓	✓
addImage(Previous format)	✓	✓	✓	✓	✓
addImage(Previous format)	✓	✓	✓	✓	✓
addLogo	✓	✓	✓	✓	✓
addBarcode	✓	✓	✓	✓	✓
addSymbol	✓	✓	✓	✓	✓
addPageBegin	✓	✓	✓	✓	✓
addPageEnd	✓	✓	✓	✓	✓
addPageArea	✓	✓	✓	✓	✓
addPageDirection	✓	✓	✓	✓	✓
addPagePosition	✓	✓	✓	✓	✓
addPageLine	-	-	-	✓	✓
addPageRectangle	-	-	-	✓	✓
addCut	✓	✓	✓	✓	✓

API	TM-m10	TM-m30	TM-T60	TM-T88VI	TM-T88VI-iHUB
addPulse	✓	✓	✓	✓	✓
addSound	✓	✓	-	✓	✓
addSound(Previous format)	✓	✓	-	✓	✓
addFeedPosition	-	-	-	✓	✓
addLayout	-	-	-	✓	✓
addCommand	✓	✓	✓	✓	✓

TM-m10

The TM-m10 model information is listed in the table below.

		58 mm
Resolution		203 dpi x 203 dpi (W x H)
Country		ANK model Japanese model Traditional Chinese model
Print Width		420 dots
Characters in a Line	Font A	ANK: 35 characters Kanji ^{*1} : 17 characters
	Font B	ANK: 42 characters Kanji ^{*2} : 21 characters
	Font C	ANK: 46 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H) Kanji ^{*1} : 24 dots x 24 dots (W x H)
	Font B	ANK: 10 dots x 24 dots (W x H) Kanji ^{*2} : 20 dots x 24 dots (W x H)
	Font C	ANK: 9 dots x 17 dots (W x H)
Character Baseline	Font A	ANK; At the 21st dot from the top of the character Kanji ^{*1} : At the 21st dot from the top of the character
	Font B	ANK; At the 21st dot from the top of the character Kanji ^{*2} : At the 21st dot from the top of the character
	Font C	At the 16th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		420 dots x 2400 dots (W x H)
Page Mode Maximum Area		420 dots x 2400 dots (W x H)
Barcode		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
Two-Dimensional Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, Data Matrix (Composite Symbolology not supported)
Paper Cut		Cut, Feed cut
Drawer Kick-Out		Supported
Buzzer		Option (Pattern A ~ Pattern E, Error, No paper, Stop)
Battery		Not supported
Bluetooth connection		Supported only by the TM-m10 Bluetooth® model.

*1 Differs depending on the Multilingual Model specifications.

*2 Only for Japanese model.

API information

For information about the APIs, refer to "ePOS-Print SDK for iOS User's Manual".

The differential information for the APIs is listed in the table below.

API	Parameter	Specifiable Settings Value	Description
initWithPrinterModel	printerModel	"TM-m10"	<ul style="list-style-type: none"> • TM-m10 USB model • TM-m10 Ethernet model • TM-m10 Wi-Fi model • TM-m10 <i>Bluetooth</i>® model
	lang	EPOS_OC_MODEL_ANK	ANK model
		EPOS_OC_MODEL_JAPANESE	Japanese model
		EPOS_OC_MODEL_TAIWAN	Traditional Chinese model
addTextFont	font	EPOS_OC_FONT_A	Font A
		EPOS_OC_FONT_B	Font B
		EPOS_OC_FONT_C	Font C
addImage	mode	EPOS_OC_MODE_MONO	Monochrome (2 tone)
		EPOS_OC_MODE_GRAY16	Multiple tone (16 tone)
		EPOS_OC_PARAM_DEFAULT	Default value ((Monochrome (2 tone))
	compress	EPOS_OC_COMPRESS_DEFLATE	Image compression is carried out.
		EPOS_OC_COMPRESS_NONE	Image compression is not carried out.
		EPOS_OC_PARAM_DEFAULT	Default value (Image compression is not carried out)

TM-m30

The TM-m30 model information is listed in the table below.

		58 mm	80 mm
Resolution		203 dpi x 203 dpi (W x H)	
Country		ANK model Japanese model Simplified Chinese model Traditional Chinese model Korean model	
Print Width		420 dots	576 dots
Characters in a Line	Font A	ANK: 35 characters Kanji ^{*1} : 17 characters	ANK: 48 characters Kanji ^{*1} : 24 characters
	Font B	ANK: 42 characters Kanji ^{*2} : 21 characters Kanji ^{*3} : 26 characters	ANK: 57 characters Kanji ^{*2} : 28 characters Kanji ^{*3} : 36 characters
	Font C	ANK: 46 characters	ANK: 64 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H) / Kanji ^{*1} : 24 dots x 24 dots (W x H)	
	Font B	10 dots x 24 dots (W x H) Kanji ^{*2} : 20 dots x 24 dots (W x H) / Kanji ^{*3} : 16 dots x 16 dots (W x H)	
	Font C	9 dots x 17 dots (W x H)	
Character Baseline	Font A	ANK: At the 21st dot from the top of the character Kanji ^{*1} : At the 21st dot from the top of the character	
	Font B	At the 21st dot from the top of the character Kanji ^{*2} : At the 21st dot from the top of the character Kanji ^{*3} : At the 15th dot from the top of the character	
	Font C	At the 16th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		420 dots x 2400 dots (W x H)	576 dots x 2400 dots (W x H)
Page Mode Maximum Area		420 dots x 2400 dots (W x H)	576 dots x 2400 dots (W x H)
Barcode		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	
Two-Dimensional Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, Data Matrix (Composite Symbology not supported)	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Option (Pattern A ~ Pattern E, Error, No paper, Stop)	
Battery		Not supported	
Bluetooth connection		Supported only by the TM-m30 Bluetooth® model.	

*1 Differs depending on the Multilingual Model specifications.

*2 Only for Japanese model.

*3 Only for Korean model.

API information

For information about the APIs, refer to "ePOS-Print SDK for iOS User's Manual".

The differential information for the APIs is listed in the table below.

API	Parameter	Specifiable Settings Value	Description
initWithPrinterModel	printerModel	"TM-m30"	<ul style="list-style-type: none"> TM-m30 Standard model TM-m30 <i>Bluetooth</i>® model
	lang	EPOS_OC_MODEL_ANK	ANK model
		EPOS_OC_MODEL_JAPANESE	Japanese model
		EPOS_OC_MODEL_CHINESE	Simplified Chinese model
		EPOS_OC_MODEL_TAIWAN	Traditional Chinese model
		EPOS_OC_MODEL_KOREAN	Korean model
addTextFont	font	EPOS_OC_FONT_A	Font A
		EPOS_OC_FONT_B	Font B
		EPOS_OC_FONT_C	Font C
addImage	mode	EPOS_OC_MODE_MONO	Monochrome (2 tone)
		EPOS_OC_MODE_GRAY16	Multiple tone (16 tone)
		EPOS_OC_PARAM_DEFAULT	Default value ((Monochrome (2 tone))
	compress	EPOS_OC_COMPRESS_DEFLATE	Image compression is carried out.
		EPOS_OC_COMPRESS_NONE	Image compression is not carried out.
		EPOS_OC_PARAM_DEFAULT	Default value (Image compression is not carried out)

TM-T60

The TM-T60 model information is listed in the table below.

		80 mm
Resolution		203 dpi x 203 dpi (W x H)
Country		Simplified Chinese model
Print Width		576 dots
Characters in a Line	Font A	ANK: 48 characters Kanji: 24 characters
	Font B	ANK: 64 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H) Kanji: 24 dots x 24 dots (W x H)
	Font B	9 dots x 17 dots (W x H)
Character Baseline	Font A	ANK: At the 21st dot from the top of the character Kanji: At the 21th dot from the top of the character
	Font B	At the 16th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		576 dots x 831 dots (W x H)
Page Mode Maximum Area		576 dots x 1662 dots (W x H)
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128
Two-Dimensional Code		PDF417, QR Code
Paper Cut		Cut, Feed cut
Drawer Kick-Out		Supported
Buzzer		Not supported
Battery		Not supported

API information

For information about the APIs, refer to "ePOS-Print SDK for iOS User's Manual".

The differential information for the APIs is listed in the table below.

API	Parameter	Specifiable Settings Value	Description
initWithPrinterModel	printerModel	"TM-T60"	TM-T60
	lang	EPOS_OC_MODEL_CHINESE	Simplified Chinese model
addTextFont	font	EPOS_OC_FONT_A	Font A
		EPOS_OC_FONT_B	Font B
addImage	mode	EPOS_OC_MODE_MONO	Monochrome (2 tone)
		EPOS_OC_PARAM_DEFAULT	Default value ((Monochrome (2 tone))
	compress	EPOS_OC_COMPRESS_NONE	Image compression is not carried out.
		EPOS_OC_PARAM_DEFAULT	Default value (Image compression is not carried out)

TM-T88VI

The TM-T88VI model information is listed in the table below.

		58 mm	80 mm
Resolution		180 dpi x 180 dpi (W x H)	
Country		ANK model Japanese model Simplified Chinese model Traditional Chinese model Korean model South Asian model	
Print Width		360 dots	512 dots
Characters in a Line	Font A	ANK: 30 characters Kanji ^{*1} : 15 characters	ANK: 42 characters Kanji ^{*1} : 21 characters
	Font B	ANK: 40 characters Kanji ^{*2} : 22 characters	ANK: 56 characters Kanji ^{*2} : 32 characters
	Special font A ^{*3}	30 characters	42 characters
	Special font B ^{*3}	40 characters	56 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H) / Kanji ^{*1} : 24 dots x 24 dots (W x H)	
	Font B	ANK: 9 dots x 17 dots (W x H) / Kanji ^{*2} : 16 dots x 16 dots (W x H)	
	Special font A ^{*3}	12 dots x 24 dots (W x H)	
	Special font B ^{*3}	9 dots x 24 dots (W x H)	
Character Baseline	Font A	ANK: At the 21st dot from the top of the character Kanji ^{*1} : At the 21st dot from the top of the character	
	Font B	ANK: At the 16th dot from the top of the character Kanji ^{*2} : At the 15th dot from the top of the character	
	Special font A ^{*3}	At the 20th dot from the top of the character	
	Special font B ^{*3}	At the 20th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		360 dots x 831 dots (W x H)	512 dots x 831 dots (W x H)
Page Mode Maximum Area		360 dots x 2400 dots (W x H)	512 dots x 2400 dots (W x H)
Barcode		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	
Two-Dimensional Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, Data Matrix (Composite Symbology not supported)	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Option (Pattern A ~ Pattern E, Error, No paper, Stop)	
Battery		Not supported	
Bluetooth connection		Supported only by the TM-T88VI Bluetooth® model.	

*1 Differs depending on the Multilingual Model specifications.

*2 Only for Korean model.

*3 Only for South Asian model.

API information

For information about the APIs, refer to "ePOS-Print SDK for iOS User's Manual".

The differential information for the APIs is listed in the table below.

API	Parameter	Specifiable Settings Value	Description
initWithPrinterModel	printerModel	"TM-T88VI"	TM-T88VI
	lang	EPOS_OC_MODEL_ANK	ANK model
		EPOS_OC_MODEL_JAPANESE	Japanese model
		EPOS_OC_MODEL_CHINESE	Simplified Chinese model
		EPOS_OC_MODEL_TAIWAN	Traditional Chinese model
		EPOS_OC_MODEL_KOREAN	Korean model
		EPOS_OC_MODEL_SOUTHASIA	South Asian model
addTextFont	font	EPOS_OC_FONT_A	Font A
		EPOS_OC_FONT_B	Font B
addImage	mode	EPOS_OC_MODE_MONO	Monochrome (2 tone)
		EPOS_OC_MODE_GRAY16	Multiple tone (16 tone)
		EPOS_OC_PARAM_DEFAULT	Default value ((Monochrome (2 tone))
	compress	EPOS_OC_COMPRESS_DEFLATE	Image compression is carried out.
		EPOS_OC_COMPRESS_NONE	Image compression is not carried out.
		EPOS_OC_PARAM_DEFAULT	Default value (Image compression is not carried out)

TM-T88VI-iHUB

The TM-T88VI-iHUB model information is listed in the table below.

		58 mm	80 mm
Resolution		180 dpi x 180 dpi (W x H)	
Country		ANK model Japanese model Simplified Chinese model Traditional Chinese model Korean model South Asian model	
Print Width		360 dots	512 dots
Characters in a Line	Font A	ANK: 30 characters Kanji ^{*1} : 15 characters	ANK: 42 characters Kanji ^{*1} : 21 characters
	Font B	ANK: 40 characters Kanji ^{*2} : 22 characters	ANK: 56 characters Kanji ^{*2} : 32 characters
	Special font A ^{*3}	30 characters	42 characters
	Special font B ^{*3}	40 characters	56 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H) / Kanji ^{*1} : 24 dots x 24 dots (W x H)	
	Font B	ANK: 9 dots x 17 dots (W x H) / Kanji ^{*2} : 16 dots x 16 dots (W x H)	
	Special font A ^{*3}	12 dots x 24 dots (W x H)	
	Special font B ^{*3}	9 dots x 24 dots (W x H)	
Character Baseline	Font A	ANK: At the 21st dot from the top of the character Kanji ^{*1} : At the 21st dot from the top of the character	
	Font B	ANK: At the 16th dot from the top of the character Kanji ^{*2} : At the 15th dot from the top of the character	
	Special font A ^{*3}	At the 20th dot from the top of the character	
	Special font B ^{*3}	At the 20th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		360 dots x 831 dots (W x H)	512 dots x 831 dots (W x H)
Page Mode Maximum Area		360 dots x 2400 dots (W x H)	512 dots x 2400 dots (W x H)
Barcode		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	
Two-Dimensional Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, Data Matrix (Composite Symbology not supported)	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Option (Pattern A ~ Pattern E, Error, No paper, Stop)	
Battery		Not supported	

*1 Differs depending on the Multilingual Model specifications.

*2 Only for Korean model.

*3 Only for South Asian model.

API information

For information about the APIs, refer to "ePOS-Print SDK for iOS User's Manual".

The differential information for the APIs is listed in the table below.

API	Parameter	Specifiable Settings Value	Description
initWithPrinterModel	printerModel	"TM-T88VI"	TM-T88VI
	lang	EPOS_OC_MODEL_ANK	ANK model
		EPOS_OC_MODEL_JAPANESE	Japanese model
		EPOS_OC_MODEL_CHINESE	Simplified Chinese model
		EPOS_OC_MODEL_TAIWAN	Traditional Chinese model
		EPOS_OC_MODEL_KOREAN	Korean model
		EPOS_OC_MODEL_SOUTHASIA	South Asian model
addTextFont	font	EPOS_OC_FONT_A	Font A
		EPOS_OC_FONT_B	Font B
addImage	mode	EPOS_OC_MODE_MONO	Monochrome (2 tone)
		EPOS_OC_MODE_GRAY16	Multiple tone (16 tone)
		EPOS_OC_PARAM_DEFAULT	Default value ((Monochrome (2 tone))
	compress	EPOS_OC_COMPRESS_DEFLATE	Image compression is carried out.
		EPOS_OC_COMPRESS_NONE	Image compression is not carried out.
		EPOS_OC_PARAM_DEFAULT	Default value (Image compression is not carried out)