

# Epson ePOS SDK for Android

## ユーザーズマニュアル

[機能](#)

[使い方](#)

[API リファレンス](#)

[機器仕様](#)

[サンプルプログラム](#)

[活用ガイド](#)

[付録](#)

## 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、上項にかか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、EXCEED YOUR VISION、ESC/POS は、セイコーエプソン株式会社の登録商標です。

Android™は、Google LLC の商標です。

Java™は、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における商標です。

銀聯、UnionPay は、CHINA UNIONPAY CO., Ltd. の登録商標です。

楽天 Edy は、楽天 Edy 株式会社の登録商標です。

ID は株式会社 NTT ドコモの登録商標です。

nanaco は、株式会社セブン・カードサービスの登録商標です。

QUICPay は、株式会社ジェーシービーの登録商標です。

Suica は、東日本旅客鉄道株式会社の登録商標です。

WAON は、イオン株式会社の登録商標です。

Wi-Fi® は、Wi-Fi Alliance® の登録商標です。

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。

QR コードは株式会社デンソーウェーブの登録商標です。

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。

## ESC/POS® コマンドシステム

エプソンは、独自の POS プリンターコマンドシステム、ESC/POS により、業界のイニシアチブをとってきました。ESC/POS は特許取得済みのものを含む数多くの独自のコマンドを持ち、高い拡張性で多才な POS システムの構築を実現します。ほとんどのエプソン POS プリンターとディスプレイに互換性を持つ他、この独自の制御システムにはフレキシビリティーもあるため、将来アップグレードが行いやすくなります。その機能と利便性は世界中で評価されています。

# 安全のために

## 記号の意味

本書では以下の記号が使われています。それぞれの記号の意味をよく理解してから製品を取り扱ってください。



ご使用上、必ずお守りいただきたいことを記載しています。この表示を無視して誤った取り扱いをすると、製品の故障や動作不良の原因になる可能性があります。



補足説明や知っておいていただきたいことを記載しています。

## 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。

## 本書について

### 本書の目的

Epson ePOS SDK for Android を利用したアプリケーション開発に必要な情報を提供します。

### 本書の構成

本書は次のように構成されています。

第1章	機能
第2章	使い方
第3章	API リファレンス
第4章	機器仕様
第5章	サンプルプログラム
第6章	活用ガイド
付録	オープンソースソフトウェアライセンス

# もくじ

■ 安全のために .....	3
記号の意味.....	3
■ 使用制限 .....	3
■ 本書について .....	3
本書の目的.....	3
本書の構成.....	3
■ もくじ .....	4

## 機能 ..... 12

■ Epson ePOS SDK for Android でのアプリケーション開発 .....	12
TM プリンターモデル .....	13
カスタマーディスプレイモデル .....	13
TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデル .....	14
POS ターミナルモデル .....	15
■ Epson ePOS SDK for Android の開発環境 .....	16
■ アプリケーションソフトの動作環境 .....	16
対象プリンター .....	16
プリンターごとの提供機能 .....	17
サポート周辺機器 .....	18
■ ダウンロードコンテンツ .....	20
■ Web コンテンツ .....	21
■ 制限事項 .....	21

## 使い方 ..... 22

■ プリンターとの接続 .....	22
■ Epson ePOS SDK for Android の組み込み方法 .....	22
■ プログラミングガイド .....	26
プログラミングフロー .....	26
設定用の命令バッファーの有効範囲 .....	32
ステータス .....	32

## API リファレンス ..... 34

■ ePOS SDK API 一覧 .....	34
Printer クラス .....	34
LineDisplay クラス .....	37
Keyboard クラス .....	37
POSKeyboard クラス .....	38
BarcodeScanner クラス .....	38
MSR クラス .....	38

SimpleSerial クラス.....	39
CashChanger クラス .....	39
CAT クラス.....	41
OtherPeripheral クラス.....	41
CommBox クラス .....	42
クラス共通 API.....	42
Discovery クラス.....	42
Epos2Exception クラス.....	42
Log クラス .....	43
EasySelect クラス.....	43
EasySelectInfo クラス.....	43
<b>■ Printer クラス .....</b>	<b>44</b>
Printer.....	44
connect.....	46
disconnect.....	49
startMonitor .....	50
stopMonitor.....	51
getStatus.....	52
sendData.....	54
beginTransaction .....	55
endTransaction.....	56
requestPrintJobStatus .....	57
clearCommandBuffer .....	58
addTextAlign.....	59
addLineSpace.....	60
addTextRotate .....	61
addText.....	62
addTextLang.....	63
addTextFont.....	64
addTextSmooth.....	65
addTextSize .....	66
addTextStyle.....	67
addHPosition .....	69
addFeedUnit.....	70
addFeedLine.....	71
addImage.....	72
addLogo.....	75
addBarcode .....	76
addSymbol.....	80
addHLine .....	85
addVLineBegin .....	87
addVLineEnd .....	89
addPageBegin.....	90
addPageEnd.....	91
addPageArea .....	92
addPageDirection .....	94
addPagePosition.....	95
addPageLine.....	97
addPageRectangle.....	99
addCut.....	101
addPulse .....	102
addSound.....	104
addFeedPosition.....	106
addLayout .....	107
addCommand.....	111
getMaintenanceCounter.....	112
resetMaintenanceCounter .....	114

getPrinterSetting .....	116
setPrinterSetting .....	119
forceRecover.....	122
forcePulse.....	123
forceStopSound.....	125
forceCommand.....	126
forceReset.....	127
setStatusChangeEventListener.....	128
setReceiveEventListener.....	130
interval .....	132
<b>■ LineDisplay クラス .....</b>	<b>133</b>
LineDisplay .....	133
connect.....	134
disconnect.....	137
getStatus.....	138
sendData.....	139
clearCommandBuffer .....	140
addInitialize.....	141
addCreateWindow.....	142
addDestroyWindow.....	144
addSetCurrentWindow .....	145
addClearCurrentWindow .....	146
addSetCursorPosition .....	147
addMoveCursorPosition.....	148
addSetCursorType.....	149
addText.....	150
addReverseText .....	152
addMarqueeText .....	154
addSetBlink .....	156
addSetBrightness.....	157
addShowClock.....	158
addCommand.....	159
setReceiveEventListener.....	160
<b>■ Keyboard クラス .....</b>	<b>162</b>
Keyboard .....	162
connect.....	163
disconnect.....	165
getStatus.....	166
setPrefix.....	167
getPrefix.....	168
setKeyPressEventListener.....	169
setReadStringEventListener .....	170
<b>■ POSKeyboard クラス .....</b>	<b>171</b>
PosKeyboard.....	171
connect.....	172
disconnect.....	174
getStatus.....	175
setKeyPressEventListener.....	176
<b>■ BarcodeScanner クラス .....</b>	<b>177</b>
BarcodeScanner .....	177
connect.....	178
disconnect.....	180
getStatus.....	181
setScanEventListener .....	182

■ MSR クラス.....	183
Msr .....	183
connect.....	184
disconnect.....	186
getStatus.....	187
setDataEventListener .....	188
■ SimpleSerial クラス .....	190
SimpleSerial .....	190
connect.....	191
disconnect.....	193
getStatus.....	194
sendCommand .....	195
setReceiveEventListener.....	196
■ CashChanger クラス (デバイス制御プログラム) .....	197
CashChanger .....	197
connect.....	198
disconnect.....	200
getStatus.....	201
getOposErrorCode .....	202
readCashCount .....	203
beginDeposit .....	204
pauseDeposit .....	205
restartDeposit.....	206
endDeposit .....	207
dispenseChange.....	208
dispenseCash.....	209
sendDirectIOWorkerCommand .....	210
setCashCountEventListener.....	211
setDepositEventListener .....	213
setDispenseEventListener .....	215
setDirectIOWorkerCommandReplyEventListener.....	217
setDirectIOWorkerEventListener .....	219
setStatusUpdateEventListener .....	220
■ CashChanger クラス (デバイス制御スクリプト).....	222
CashChanger .....	222
connect.....	223
disconnect.....	225
getStatus.....	226
setConfigCountMode.....	227
setConfigLeftCash .....	228
readCashCount .....	229
beginDeposit .....	230
pauseDeposit .....	231
restartDeposit.....	232
endDeposit .....	233
dispenseChange.....	234
dispenseCash.....	235
collectCash .....	236
openDrawer .....	237
sendCommand .....	238
setCashCountEventListener.....	239
setDepositEventListener .....	241
setDispenseEventListener .....	243
setConfigChangeEventListener.....	245
setCollectEventListener.....	247

setCommandReplyEventListener.....	249
setStatusChangeEventListener.....	251
<b>■ CAT クラス .....</b>	<b>253</b>
Cat.....	253
connect.....	254
disconnect.....	256
getStatus.....	257
getOposErrorCode .....	258
authorizeSales.....	259
authorizeVoid.....	261
authorizeRefund.....	263
authorizeCompletion .....	264
accessDailyLog .....	265
sendDirectIOCommand .....	266
setTimeout .....	268
getTimeout.....	269
setTrainingMode .....	270
getTrainingMode .....	271
setAuthorizeSalesEventListener.....	272
setAuthorizeVoidEventListener .....	275
setAuthorizeRefundEventListener .....	278
setAuthorizeCompletionEventListener .....	281
setAccessDailyLogEventListener .....	284
setDirectIOCommandReplyEventListener.....	286
setStatusUpdateEventListener .....	289
<b>■ OtherPeripheral クラス .....</b>	<b>291</b>
OtherPeripheral.....	291
connect.....	292
disconnect.....	294
getStatus.....	295
sendData.....	296
setReceiveEventlistener.....	297
<b>■ CommBox クラス .....</b>	<b>298</b>
CommBox.....	298
connect.....	299
disconnect.....	301
getStatus.....	302
getCommHistory .....	303
sendMessage.....	305
setReceiveEventlistener .....	307
<b>■ クラス共通 API .....</b>	<b>308</b>
getAdmin.....	308
getLocation.....	309
setConnectionEventlistener .....	310
<b>■ Discovery クラス .....</b>	<b>312</b>
start .....	312
stop .....	316
<b>■ Epos2Exception クラス .....</b>	<b>317</b>
getErrorStatus .....	317
<b>■ Log クラス .....</b>	<b>318</b>
setLogSettings.....	318
SdkVersion .....	320

■ EasySelect クラス .....	321
parseNFC .....	321
parseQR .....	322
createQR .....	323
■ EasySelectInfo クラス .....	324
deviceType .....	324
printerName .....	324
macAddress .....	324
■ Status 一覧 .....	325
Error Status と対処方法 .....	325
Callback Code と対処方法 .....	327
Printer Status と対処方法 .....	329
■ Key Code 一覧 .....	331
機器仕様 .....	333
■ サポートクラス一覧 .....	333
■ サポート API 一覧 .....	335
Printer クラス .....	335
LineDisplay クラス .....	342
クラス共通 API .....	345
EasySelect クラス .....	346
■ プリンター別サポート情報 .....	347
TM-m10 .....	347
TM-m30 .....	350
TM-T20 .....	353
TM-T20II .....	355
TM-T20III .....	358
TM-T70 .....	360
TM-T70II .....	362
TM-T88V .....	364
TM-T88VI .....	366
TM-T90II .....	369
TM-P20 .....	371
TM-P60II (Peeler) .....	375
TM-P80 .....	378
TM-T70-i .....	381
TM-T88V-i .....	383
TM-T70II-DT .....	385
TM-T70II-DT2 .....	387
TM-T88V-DT .....	389
TM-T88VI-DT2 .....	391
TM-T88IV .....	393
TM-T90 .....	395
TM-T90KP .....	397
TM-L90 .....	399
■ フームウェアバージョンによる使用制限 .....	401
TM-i シリーズ / TM-DT シリーズ .....	401
TM-m30 .....	402
TM-T88VI .....	402

---

## サンプルプログラム ..... 403

■ 機能 .....	403
ePOS2Printer .....	403
ePOS2Discovery .....	403
ePOS2LineDisplay .....	404
ePOS2Keyboard .....	404
ePOS2POSKeyboard .....	404
ePOS2BarcodeScanner .....	404
ePOS2MSR .....	405
ePOS2SimpleSerial .....	405
ePOS2CashChanger .....	405
ePOS2CAT .....	406
ePOS2OtherPeripheral .....	406
ePOS2CommBox .....	407
ePOS2EasySelection .....	407
■ 使用環境 .....	408
■ インストール手順 .....	408

---

## 活用ガイド ..... 411

■ USB 接続するには .....	411
■ プリンターを検索するには .....	412
■ 常時監視するには .....	413
■ 一台のプリンターを複数のモバイル端末から使用するには .....	415
■ トランザクションを指定するには .....	416
■ ネットワークを自動で再接続するには .....	417
■ アプリケーション間でデータを送受信するには .....	418
コミュニケーションボックスを使用したアプリケーション例 .....	418
コミュニケーションボックスの仕組み .....	419
コミュニケーションボックスの仕様 .....	419
コミュニケーションボックスの使用したデータ処理例 .....	420
基本的なプログラミングシーケンス .....	421
■迂回印刷するには .....	423
■ NFC タグを使ってプリンターを選択するには .....	425
■ QR コードを使ってプリンターを選択するには .....	427
■ 自動つり銭機を制御するには .....	429
デバイス制御プログラム .....	429
デバイス制御スクリプト .....	430
■ 信用決済端末を制御するには .....	431
■ 複数の周辺機器を制御するには .....	432

---

## 付録 ..... 434

■ オープンソースソフトウェアライセンス .....	434
----------------------------	-----

サンプルプログラム .....	434
ライブラリー .....	434

# 機能

Epson ePOS SDK for Android は、Android アプリケーションソフト開発用の SDK です。Epson ePOS SDK for Android を使用すると、シンプルなコードでプリンターを制御できます。また、一部の TM プリンターでは、本体に接続したカスタマーディスプレイやバーコードスキャナーなどの周辺機器、ネットワーク接続した子機プリンターを制御できます。さらに、スプーラー機能やコミュニケーションボックスなどの独自機能を使用したアプリケーションソフトも開発できます。

本書ではプリンターを以下のように総称します。

総称	説明
TM プリンター	以下のレシートプリンターの総称 • 単機能モデル • モバイルモデル • TM-i シリーズ • TM-DT シリーズ
子機プリンター	ネットワーク接続した TM プリンターで、スマートデバイスから親機となる TM プリンターを経由して制御する TM プリンターの総称

## Epson ePOS SDK for Android でのアプリケーション開発

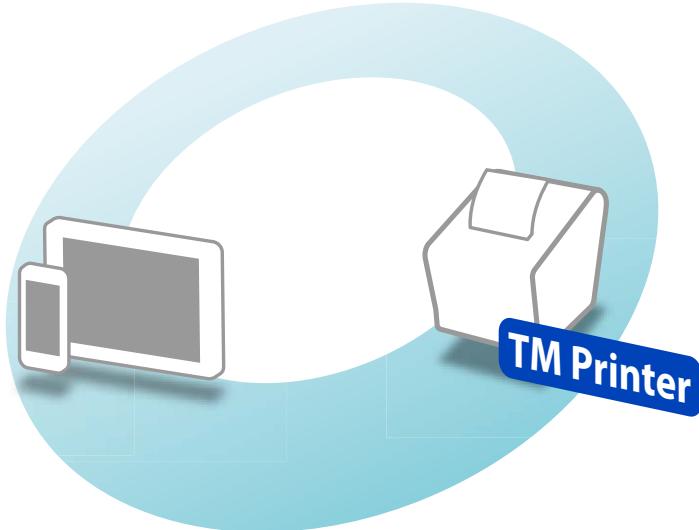
Epson ePOS SDK for Android で開発できるアプリケーションソフトのシステム構成は、以下の通りです。  
システムの構築方法は、各プリンターの詳細取扱説明書を参照してください。

## TM プリンター モデル

スマートデバイスと TM プリンターを通信接続するシステムです。

使用するスマートデバイスと TM プリンターの仕様に合わせた接続方法（Bluetooth/Wi-Fi/Ethernet/USB）を選択できます。

Wi-Fi/Ethernet 接続では複数の TM プリンターを制御できます。

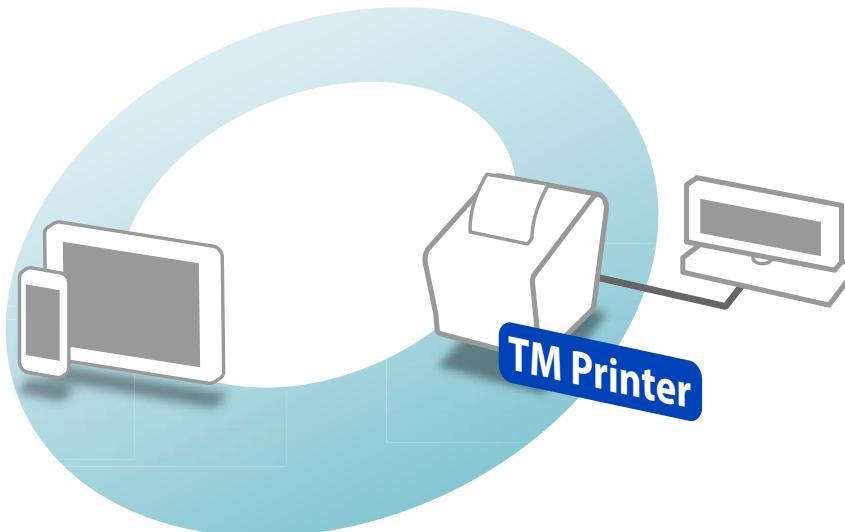


## カスタマーディスプレイ モデル

スマートデバイスから TM プリンターとカスタマーディスプレイを制御するシステムです。

TM プリンターにカスタマーディスプレイを接続し、スマートデバイスと TM プリンターを通信接続して使用します。

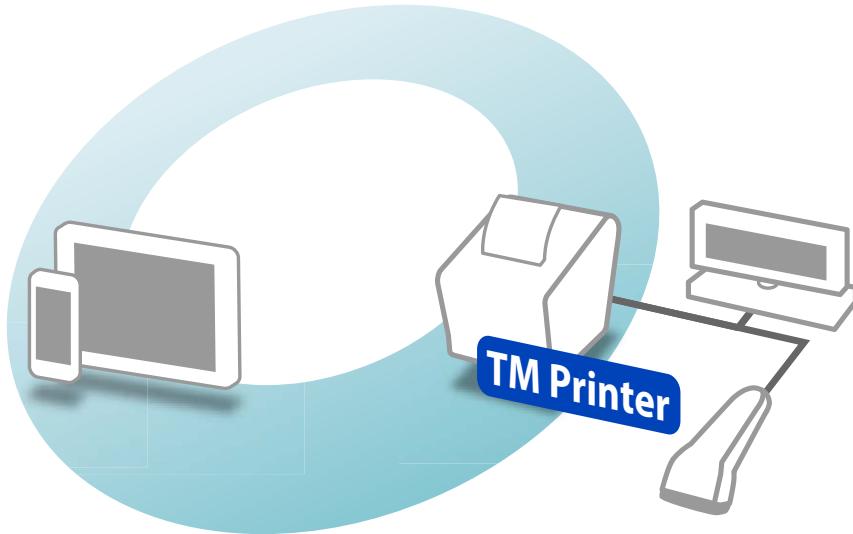
使用するスマートデバイスと TM プリンターの仕様に合わせた接続方法（Bluetooth/Wi-Fi/Ethernet/USB）を選択できます。



## TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデル

スマートデバイスから TM-m30 または TM-T88VI と DM-D30、バーコードスキャナーを制御するシステムです。 TM-m30 または TM-T88VI に DM-D30 とバーコードスキャナーを接続し、スマートデバイスと TM-m30 または TM-T88VI を通信接続して使用します。

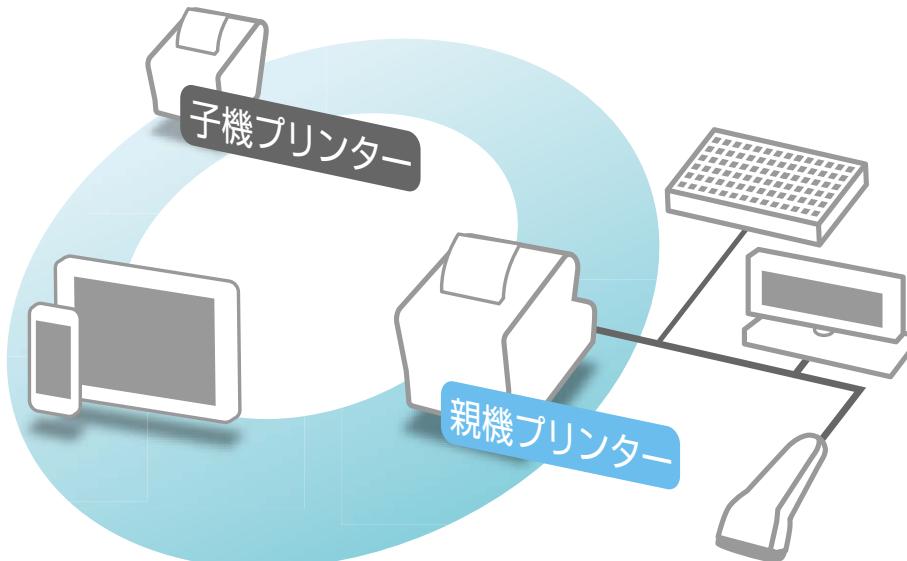
使用するスマートデバイスと TM-m30 または TM-T88VI の仕様に合わせた接続方法（Bluetooth/Wi-Fi/Ethernet/USB）を選択できます。



- TM プリンターの ePOS-Device 設定を「有効」にする必要があります。
  - \* TM-m30 の ePOS-Device 設定は、TM-m30 の EpsonNet Config (Web バージョン) で行います。
  - \* TM-T88VI の ePOS-Device 設定は、TM-T88VI Utility で行います。
- TM-m30 または TM-T88VI+ バーコードスキャナーの組み合わせでも使用できます。

## POS ターミナルモデル

スマートデバイスと POS ターミナルとなる TM プリンター（親機）を Wi-Fi/Ethernet のネットワーク通信で接続し、親機に子機プリンターやカスタマーディスプレイなど、複数の周辺機器を接続して運用するシステムです。



POS ターミナルモデルはエプソン製の周辺機器の他に、エプソンが独自に開発したデバイス制御プログラムやデバイス制御スクリプトから、様々な周辺機器を制御できます。

POS ターミナルとなる TM プリンターによって搭載するソフトウェアやそのバージョンが異なるため、制御できる周辺機器に違いがあります。

制御できる周辺機器については、各プリンターの詳細取扱説明書を参照してください。

デバイス制御プログラムは、TM-DT シリーズ（TM-DT ソフトウェア Ver.4.0 以降）で実装されています。

デバイス制御プログラムについては、TM-DT シリーズ周辺機器制御ガイドを参照してください。

# Epson ePOS SDK for Android の開発環境

Epson ePOS SDK for Android を使用してアプリケーションソフトを開発するには、以下の開発環境が必要です。

- Android SDK r15 以降
- Java Development Kit 7 以降

## アプリケーションソフトの動作環境

Epson ePOS SDK for Android を使用して開発するアプリケーションソフトの動作環境は、以下の通りです。  
対象となるスマートデバイスの最新情報は、Epson ePOS SDK for Android パッケージの README.jp.txt を参考してください。

### 対象プリンター

アプリケーションソフトから制御できるプリンターは、以下の通りです。



使用できるインターフェイスは TM プリンターによって異なります。詳細は、各プリンターの  
詳細取扱説明書を参照してください。

### 単機能モデル

- |             |            |
|-------------|------------|
| □ TM-m10    | □ TM-T70   |
| □ TM-m30    | □ TM-T70II |
| □ TM-T20    | □ TM-T88V  |
| □ TM-T20II  | □ TM-T88VI |
| □ TM-T20III | □ TM-T90II |

### モバイルモデル

- TM-P20
- TM-P60II
- TM-P80

### TM-i シリーズ

- TM-T70-i (TM-i ファームウェア Ver.4.0 以降)
- TM-T88V-i (TM-i ファームウェア Ver.4.0 以降)

### TM-DT シリーズ

- |                |                |
|----------------|----------------|
| □ TM-T70II-DT  | □ TM-T88V-DT   |
| □ TM-T70II-DT2 | □ TM-T88VI-DT2 |

## プリンターごとの提供機能

Epson ePOS SDK for Android が提供する機能は、プリンターによって使用できる機能が異なります。以下の一覧表を参照してください。

	SSL/TLS 通信 <sup>*1</sup>	内蔵ブザーの鳴動制御	外付けオプションブザーの鳴動制御	複数インターフェイスによる印刷	子機プリンターの制御	スプーラー / 迂回印刷	コミュニケーションボックス	周辺機器の制御 <sup>*2</sup>	NFC タグを使った簡単選択	QR コードを使った簡単選択
TM-m10	-	-	-	-	-	-	-	-	-	-
TM-m30	-	-	✓	✓ <sup>*5</sup>	-	-	-	✓	✓	✓
TM-T20	-	-	✓	-	-	-	-	-	-	✓
TM-T20II	-	-	✓	-	-	-	-	-	-	✓
TM-T20III	-	-	✓	-	-	-	-	-	-	✓
TM-T70	-	-	✓	-	-	-	-	-	-	✓
TM-T70II	-	✓ <sup>*4</sup>	✓	-	-	-	-	-	-	✓
TM-T88V	-	✓ <sup>*4</sup>	✓	-	-	-	-	-	-	✓
TM-T88VI	✓ <sup>*3</sup>	✓ <sup>*4</sup>	✓	✓ <sup>*5</sup>	-	-	-	✓	✓	✓
TM-T90II	-	-	✓	-	-	-	-	-	-	✓
TM-P20	-	✓	-	-	-	-	-	-	✓	✓
TM-P60II	-	✓	-	-	-	-	-	-	✓	✓
TM-P80	-	✓	-	-	-	-	-	-	✓	✓
TM-T70-i	✓	-	-	-	✓	✓	✓	✓	-	-
TM-T88V-i	✓	-	✓	-	✓	✓	✓	✓	-	-
TM-T70II-DT	✓	-	✓	-	✓	✓	✓	✓	-	-
TM-T70II-DT2	✓	-	✓	-	✓	✓	✓	✓	-	-
TM-T88V-DT	✓	-	✓	-	✓	✓	✓	✓	-	-
TM-T88VI-DT2	✓	-	✓	-	✓	✓	✓	✓	-	-

\*1: connect API 使用時に接続形式として "TCPS" を指定します。

\*2: 使用できる周辺機器は、プリンターによって異なります。各プリンターの詳細取扱説明書を参照してください。

\*3: カスタマーディスプレイ + バーコードスキャナーモデルのシステム構成では使用できません。

\*4: ドローアーキックコマンドでの鳴動制御に対応しています。

\*5: TM プリンター + バーコードスキャナーのシステム構成では使用できません。

## サポート周辺機器

Epson ePOS SDK for Android では、以下の周辺機器制御をサポートしています。



- 使用できる周辺機器は、プリンターによって異なります。[プリンターごとの提供機能](#)、または各プリンターの詳細取扱説明書を参照してください。
- TM-DT シリーズのプリンターは、デバイス制御プログラムを開発することで、ここで紹介する以外の周辺機器も制御できます。

開発方法については、TM-DT シリーズ周辺機器制御ガイドを参照してください。

### 子機プリンター

以下の TM プリンターの Wi-Fi モデル

- TM-P20
- TM-P60II
- TM-P80

Wi-Fi/Ethernet インターフェイスを搭載した以下の TM プリンター

- |                                   |                                   |
|-----------------------------------|-----------------------------------|
| <input type="checkbox"/> TM-T20II | <input type="checkbox"/> TM-T88VI |
| <input type="checkbox"/> TM-T70   | <input type="checkbox"/> TM-T90   |
| <input type="checkbox"/> TM-T70II | <input type="checkbox"/> TM-T90II |
| <input type="checkbox"/> TM-T88IV | <input type="checkbox"/> TM-T90KP |
| <input type="checkbox"/> TM-T88V  | <input type="checkbox"/> TM-L90   |

以下の TM-i シリーズ /TM-DT シリーズ

- TM-T70-i (TM-i フームウェア Ver.4.0 以降)
- TM-T70II-DT (TM-DT ソフトウェア Ver.4.0 以降)
- TM-T88V-i (TM-i フームウェア Ver.4.0 以降)
- TM-T88V-DT (TM-DT ソフトウェア Ver.4.0 以降)

### カスタマーディスプレイ

- DM-D30
- DM-D110

---

## デバイス制御プログラムで制御する周辺機器

TM-DT シリーズ（TM-DT ソフトウェア Ver.4.0 以降）が持つデバイス制御プログラムから、OPOS 仕様準拠の周辺機器を制御できます。

以下のカテゴリーで、OPOS CCO 1.14.001 と組み合わせて動作するドライバーを持つ周辺機器を制御できます。

- 自動取り扱い機
- 信用照会端末
- MSR
- POS キーボード
- バーコードスキャナー

---

## デバイス制御スクリプトで制御する周辺機器

TM-i シリーズ / TM-DT シリーズが持つデバイス制御スクリプトから、以下の周辺機器を制御できます。

- キー入力デバイス
  - MSR（日立オムロン V3TU-FK）
  - キーボード（標準 HID）
  - バーコードスキャナー（標準 HID）
- シリアル通信デバイス
  - 自動取り扱い機（グローリー RT-200/RAD-200）
  - シリアル通信デバイス

# ダウンロードコンテンツ

Epson ePOS SDK for Android パッケージには、以下のファイルが含まれています。

ファイル名 / フォルダ名	説明
ePOS2.jar	API を Java プログラムから利用するための jar 形式ファイルにアーカイブされたコンパイル済み Java のクラスファイルです。
ePOSEasySelect.jar	簡単にプリンターを選択するための Java のクラスファイルです。
libepos2.so	機能実行用ライブラリーです (ARMv5TE に対応)。
libposeeasyselect.so	簡単にプリンターを選択する機能用のライブラリーです (ARMv5TE に対応)。
armeabi	ARMv5TE 向けのライブラリーを格納するフォルダです。 以下のファイルを格納しています。 <ul style="list-style-type: none"> <li>• libepos2.so</li> <li>• libposeeasyselect.so</li> </ul>
arm64-v8a	AArch64 向けのライブラリーを格納するフォルダです。 以下のファイルを格納しています。 <ul style="list-style-type: none"> <li>• libepos2.so</li> <li>• libposeeasyselect.so</li> </ul>
x86_64	x86-64 向けのライブラリーを格納するフォルダです。* 以下のファイルを格納しています。 <ul style="list-style-type: none"> <li>• libepos2.so</li> <li>• libposeeasyselect.so</li> </ul>
OPOS_CCOs_1.14.001.msi	OPOS CCO インストーラーパッケージです。
ePOS2_Sample_Android.zip	サンプルプログラムファイルです。
DeviceControlProgram_Sample.zip	デバイス制御プログラム用のサンプルプログラムファイルです。
README.jp.txt	日本語版 README ファイルです。
README.en.txt	英語版 README ファイルです。
EULA.jp.txt	日本語版 SOFTWARE LICENSE AGREEMENT を記載しています。
EULA.en.txt	英語版 SOFTWARE LICENSE AGREEMENT を記載しています。
ePOS_SDK_Android_um_ja_revx.pdf	日本語版ユーザーズマニュアルです。
ePOS_SDK_Android_um_en_revx.pdf	英語版ユーザーズマニュアルです。
TM-DT_Peripherals_ja_revx.pdf	日本語版 TM-DT シリーズ周辺機器制御ガイドです。TM-DT シリーズで周辺機器を制御する方法を説明しています。
TM-DT_Peripherals_en_revx.pdf	英語版 TM-DT シリーズ周辺機器制御ガイドです。
ePOS_SDK_Android_Migration_Guide_ja_revx.pdf	日本語版マイグレーションガイドです。ePOS-Print SDK、ePOS-Device SDK からの移行方法を説明しています。
ePOS_SDK_Android_Migration_Guide_en_revx.pdf	英語版マイグレーションガイドです。

\* アーキテクチャが x86\_64 かつ、Android API 28 (Google APIs) の仮想デバイスで動作確認しています。

## Webコンテンツ

下記サイトにて、Webリファレンスを公開しています。

<https://reference.epson-biz.com/pos/reference/>

## 制限事項

- バーコードリーダーのサフィックス（区切り文字）の設定は CR（改行コード）にしてください。これ以外の設定ではデータを取得できません。
- 2次元バーコードリーダーを使用する場合、日本語などのマルチバイト文字は正しく取得できません。ただし、TM-m30/TM-T88VIと2次元バーコードリーダーを組み合わせて使用し、QRコードを読み取る場合は、UTF-8でエンコードしたマルチバイト文字は正しく取得できます。
- 2次元バーコードのデータにASCII制御コード(0x00～0x1F)が含まれる場合、制御コードは取得できません。
- 同じアプリケーション内で同時にオープンできるデバイスポート数は16個です。
- 画面の回転時にActivityが破棄されることがあります。PrintインスタンスをActivityで保持する場合、Activityが破棄される前にPrintクラスのclosePrinterを呼び出す必要があります。
- Bluetooth接続でプリンターとの通信中に端末がスリープ状態になると、通信が切断されます。

# 使い方

Epson ePOS SDK for Android の組み込みと、Epson ePOS SDK for Android を使用した基本的なプログラミング方法を説明します。

## プリンターとの接続

構築するシステム構成に合わせて、スマートデバイスとプリンターを接続します。

詳細は、各プリンターの詳細取扱説明書を参照してください。



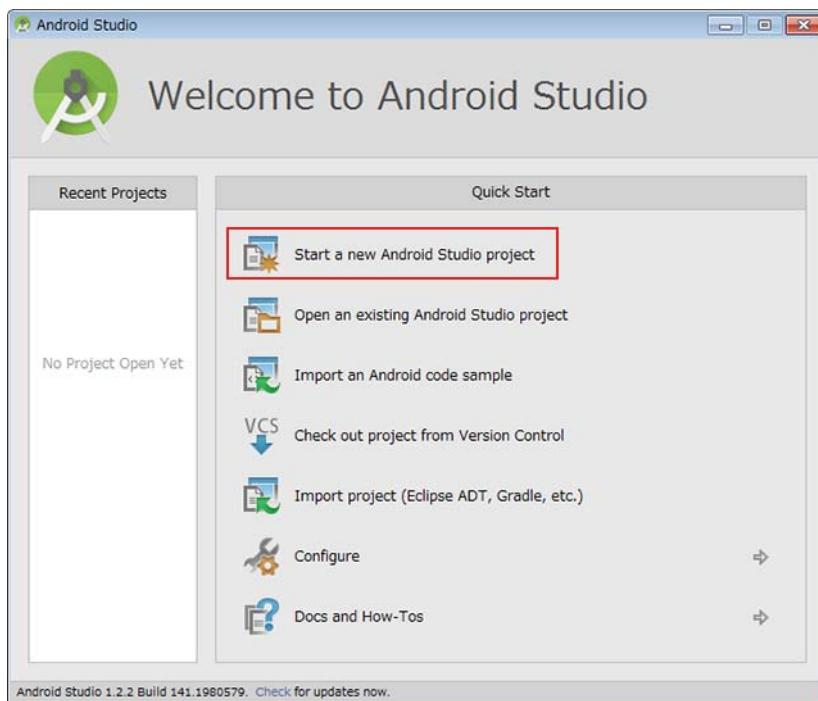
USB で接続する場合は、[USB 接続するには](#)を参照してください。

## Epson ePOS SDK for Android の組み込み方法

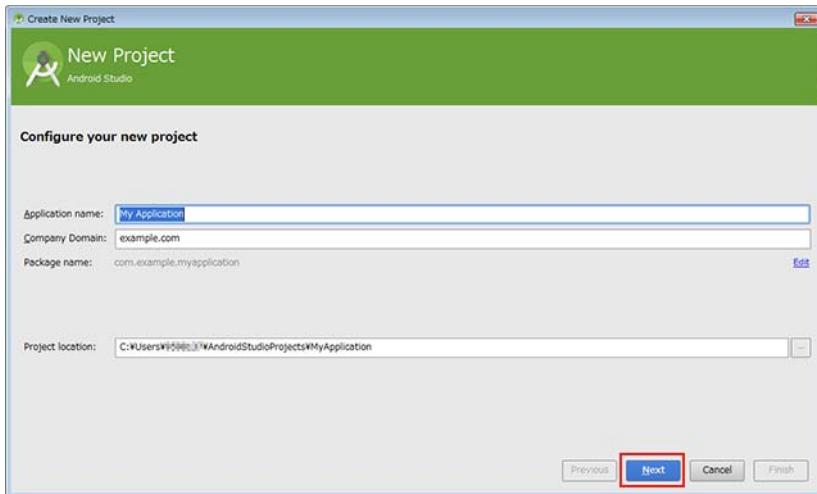


Android Studio は以下の URL から入手してください。  
<http://developer.android.com/sdk/index.html>

- 1 Android Studio を起動し、「Start a new Android Studio project」をクリックして新規プロジェクトの作成を開始します。



- 2** 作成するアプリケーションソフトに応じて、プロジェクトの設定を行います。  
画面の指示に従って設定してください。



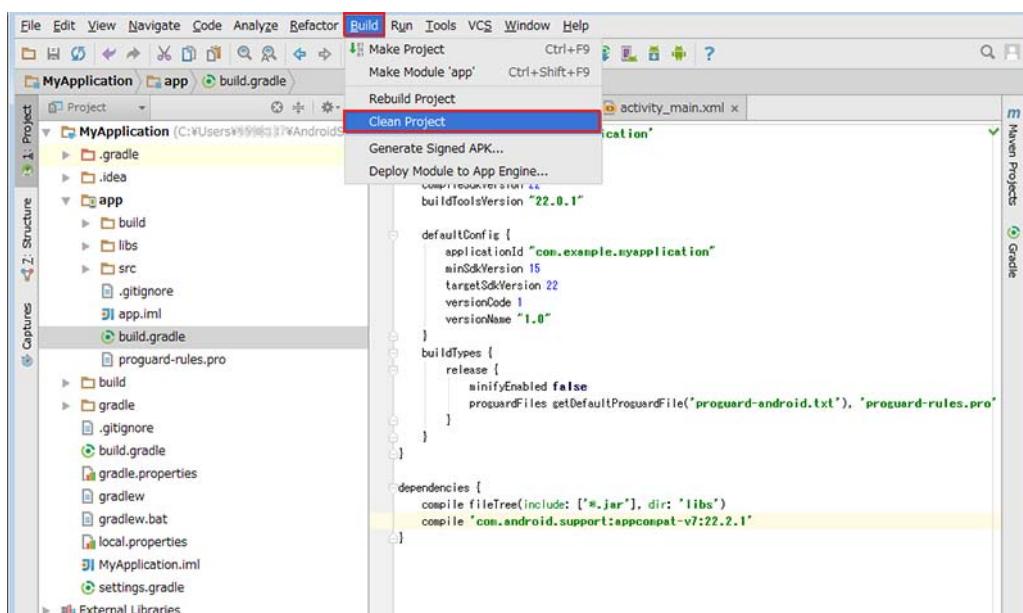
- 3** ePOS2.jar と ePOSEasySelect.jar を以下の場所に配置します。  
C:\Users\XXX\AndroidStudioProjects\zzz\app\libs  
(XXXにはアカウント名、zzzには作成したプロジェクト名が入ります。)



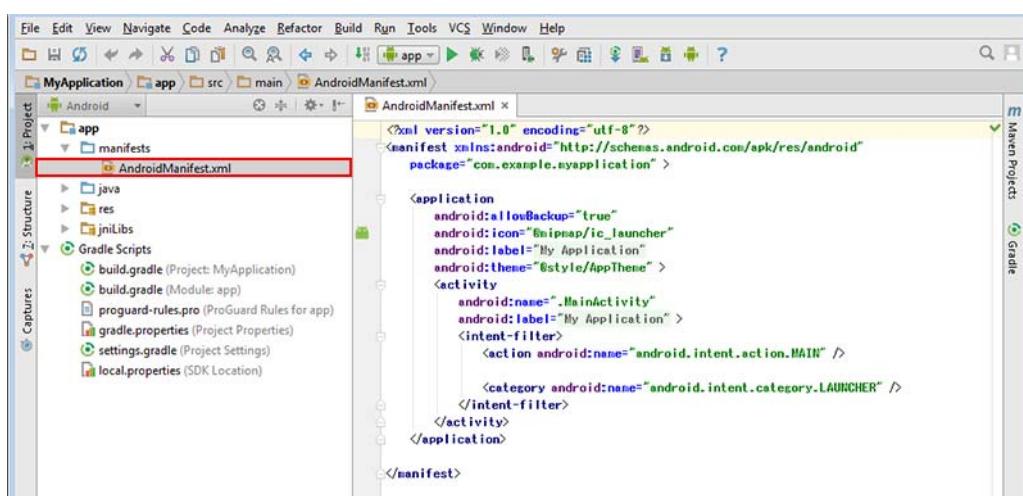
zzz 以下のフォルダ階層は、プロジェクトを作成すると自動生成されます。

- 4** 下記フォルダ階層を作成します。  
C:\Users\XXX\AndroidStudioProjects\zzz\app\src\main\jniLibs
- 5** ご利用の CPU に合わせ、本パッケージに同梱されている以下のフォルダを jniLibs フォルダの中に配置します。
- armeabi フォルダ
  - arm64-v8a フォルダ
  - x86\_64 フォルダ
- 6** Android Studio で作成したプロジェクトを開きます。

**7 「Build」から「Clean Project」を実行します。**



**8 「AndroidManifest.xml」をダブルクリックします。**



## 9 デバイスとの接続形式に合わせ、<manifest> 要素下に Permission を指定します。



デバイスと USB で接続する場合も Permission の指定を推奨します。  
詳細は、README.ja.txt を参照してください。

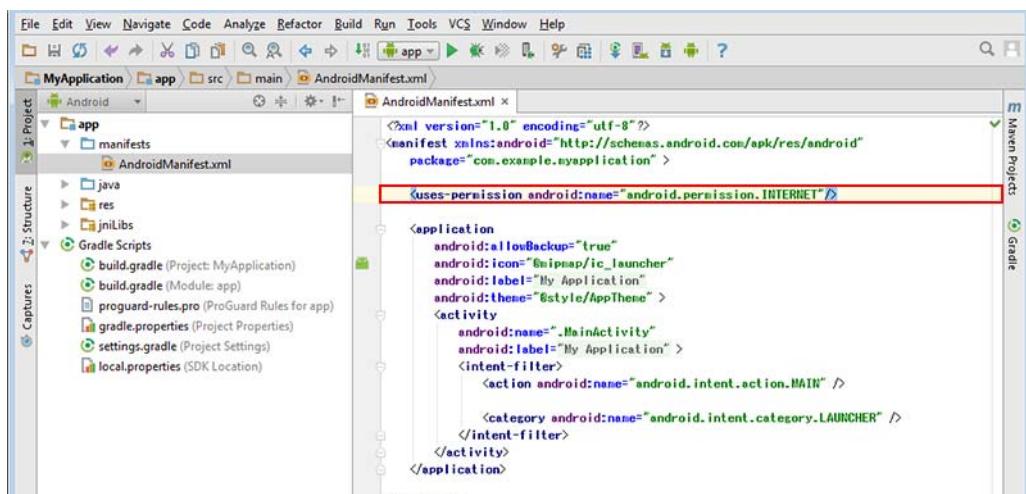
### □ Wi-Fi

```
<uses-permission android:name="android.permission.INTERNET"/>
```

### □ Bluetooth

“ACCESS\_COARSE\_LOCATION” は、Android 6.0 以降で対応しています。

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
<uses-permission
    android:name="android.permission.ACCESS_COARSE_LOCATION"/>
```



# プログラミングガイド

Epson ePOS SDK for Android を使用した基本的なプログラミング方法を説明します。

## プログラミングフロー

開発するアプリケーションソフトで制御するプリンターや周辺機器別に、プログラミングフローを説明します。

- プリンターを制御する
- カスタマーディスプレイを制御する
- キーボード、バーコードスキャナーを制御する

### プリンターを制御する

TM プリンターや子機プリンターを制御し、レシート印刷するプログラミング方法を説明します。

#### 1. プリンターの選択（クラスの初期化）



#### 2. 印刷データの作成（データのバッファリング）



#### 3. 印刷データの送信（デバイスと接続 / 通信 / 印刷 / 切断）

### プリンターの選択(クラスの初期化)

Printer クラスを初期化し、制御するプリンターを選択します。

```
Printer printer = null;
try {
    printer = new Printer(Printer.TM_T88, Printer.MODEL_ANK, this);
} catch (Epos2Exception e) {
    //エラーメッセージ表示
}
```

印刷完了イベントリスナーを登録します。

```
printer.setReceiveEventListerner(this);
```

## 印刷データの作成(データのバッファリング)

Printer クラスの addXXX 系の API を使用して、印刷データを作成します。

以下の例では、中央揃えで “Hello World” と印刷するデータを作成しています。

```
try {
    printer.addTextAlign(Printer.ALIGN_CENTER);
    printer.addText("Hello World");
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

## 印刷データの送信(デバイスと接続 / 通信 / 印刷 / 切断)

作成した印刷データを送るプリンターと接続します。

```
try {
    printer.connect("TCP:192.168.192.168", Printer.PARAM_DEFAULT);
    printer.beginTransaction();
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

印刷処理を行い、印刷結果を取得します。

コールバックメソッド内で、コマンド送信などの API を実行しないでください。

```
PrinterStatusInfo status = printer.getStatus();
if (status.getConnection() && status.getOnline()) {
    try {
        printer sendData(Printer.PARAM_DEFAULT);
    } catch (Epos2Exception e) {
        // エラーメッセージ表示
        // 切断処理
    }
} else {
    // エラーメッセージ表示
    // 切断処理
}

public void onPtrReceive(final Printer printerObj, final int code, final PrinterStatusInfo status,
final String printJobId) {
    runOnUiThread(new Runnable() {
        @Override
        public synchronized void run() {
            if (code == Epos2CallbackCode.CODE_SUCCESS) {
                // 成功メッセージ表示
            } else {
                // エラーメッセージ表示
            }
        }
    });
    new Thread(new Runnable() {
        @Override
        public synchronized void run() {
            // 切断処理
        }
    }).start();
}
```



エラー発生時、作成した印刷データは命令バッファーをクリアするまで消えません。  
エラー処理終了後などに、その印刷データを使って再印刷できます。

プリンターと切断します。

```
try {
    printer.endTransaction();
    printer.disconnect();
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

命令バッファーをクリアし、プリンターの制御を終了します。

```
printer.clearCommandBuffer();
printer.setReceiveEventlistener(null);
```

## カスタマーディスプレイを制御する

カスタマーディスプレイを制御し、ディスプレイにテキストを表示するプログラミング方法を説明します。

### 1. デバイスの選択（クラスの初期化）



### 2. 表示データの作成（データのバッファリング）



### 3. 表示データの送信（デバイスと接続 / 通信 / 表示 / 切断）

#### デバイスの選択(クラスの初期化)

LineDisplay クラスを初期化し、カスタマーディスプレイと接続します。

```
LineDisplay linedisplay = null;
try {
    linedisplay = new LineDisplay(LineDisplay.DM_D30, this);
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

表示完了イベントリスナーを登録します。

```
linedisplay.setReceiveEventlistener(this);
```

#### 表示データの作成(データのバッファリング)

LineDisplay クラスの addXXX 系の API を使用して、表示データを作成します。

以下の例では、“Hello World” と表示するデータを作成しています。

```
try {
    linedisplay.addText("Hello World");
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

## 表示データの送信(デバイスと接続 / 通信 / 表示 / 切断)

作成した表示データを送るカスタマーディスプレイと接続します。

```
try {
    linedisplay.connect("TCP:192.168.192.168", LineDisplay.PARAM_DEFAULT);
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

ディスプレイ表示処理を行い、表示結果を取得します。

```
if (status.connection) {
    try {
        linedisplay.sendData();
    }catch (Epos2Exception e) {
        // エラーメッセージ表示
        // 切断処理
    }
}else {
    // エラーメッセージ表示
    // 切断処理
}

public void onDispReceive(final LineDisplay displayObj, final int code) {
    runOnUiThread(new Runnable() {
        @Override
        public synchronized void run() {
            if (code == Epos2CallbackCode.CODE_SUCCESS) {
                // 成功メッセージ表示
            }
            else {
                // エラーメッセージ表示
            }
        }
    });
    new Thread(new Runnable() {
        @Override
        public synchronized void run() {
            // 切断処理
        }
    }).start();
}
```

カスタマーディスプレイと切断します。

```
try {
    linedisplay.disconnect();
}
catch (Epos2Exception e) {
    // エラーメッセージ表示
}
```

命令バッファーをクリアし、カスタマーディスプレイの制御を終了します。

```
linedisplay.clearCommandBuffer();
linedisplay.setReceiveEventListerner(null);
```

## キーボード、バーコードスキャナーを制御する

キーボードやバーコードスキャナーを制御し、入力データを受信するプログラミング方法を説明します。

以下の説明では、キーボードを制御する場合のプログラミングフローを説明します。

### 1. デバイスの接続



### 2. データ受信



### 3. デバイスとの切断

#### デバイスの接続

Keyboard クラスを初期化してキーボードと接続し、入力データを受信するイベントリスナーを登録します。

```
Keyboard keyboard = null;

try {
    keyboard = new Keyboard(this);

    keyboard.setKeyPressEventListener(mKeyPressEvent);

    keyboard.connect("TCP:192.168.192.168[local_keyboard]", Keyboard.PARAM_DEFAULT);
}
catch (Epos2Exception e) {
    // エラー表示

    keyboard.setKeyPressEventListener(null);
}
```

#### データ受信

キー押下イベントリスナーを作成します。

```
private KeyPressListener mKeyPressEvent = new KeyPressListener() {
    @Override
    public void onKbdKeyPress(Keyboard keyboardObj, final int keyCode, final String ascii) {
        runOnUiThread(new Runnable() {
            @Override
            public synchronized void run() {
                if (keyCode != 0) {
                    // 受信メッセージを表示
                }
            }
        });
    }
};
```

## デバイスとの切断

通信を切断し、キーボードからの入力データ受信を終了します。

```
try {
    keyboard.disconnect();
}
catch (Epos2Exception e) {
    // エラー表示
}

keyboard.setKeyPressEventListener(null);

keyboard = null;
```

## 設定用の命令バッファーの有効範囲

設定用に使用される Printer クラスの addXXX の有効範囲は、addXXX 設定後、sendData が実行されるまでです。設定した値は、sendData の実行ごとに初期化されます。以下を参考にしてください。

```
printer.addText("Hello World!\n");
printer.addTextFont(Printer.FONT_B);
printer.addText("Hello World!\n");
printer.addText("Hello World!\n");
printer.sendData(Printer.PARAM_DEFAULT);
printer.clearCommandBuffer;
printer.addText("Hello World!\n");
printer.sendData(Printer.PARAM_DEFAULT);
printer.clearCommandBuffer;
```

赤字：フォント A

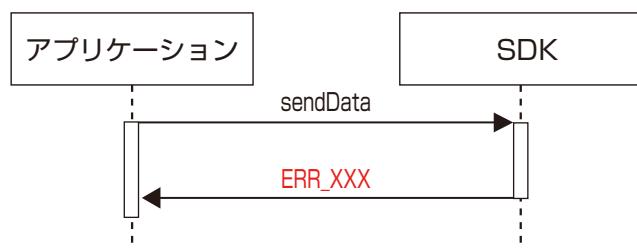
緑字：フォント B

## ステータス

Epson ePOS SDK for Android には、以下のステータスが定義されています。

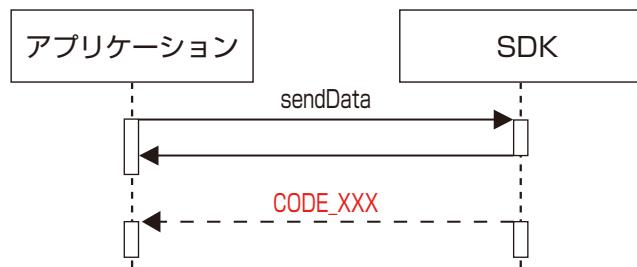
□ エラーステータス

各クラスのデバイスに対するデータ送信の結果です。API の例外として、すぐに結果が通知されます。詳細は、[Error Status と対処方法](#) を参照してください。



□ コールバックコード

`sendData` 実行後のデバイスの処理結果です。印刷完了イベント (`onPtrReceive/onDispReceive`) で通知されます。詳細は、[Callback Code と対処方法](#) を参照してください。



□ プリンターステータス

印刷データ送信時のプリンターの状態です。プリンターステータスは、`sendData` の実行時に取得します。詳細は、[Printer Status と対処方法](#) を参照してください。

## 例外処理方法

Epson ePOS SDK for Android では、エラー発生時に数値 (int) 型のパラメーターを持つ独自の例外を発生させ、呼び元にエラーを通知します。通知されるエラーは、Epos2Exception クラスの `getErrorStatus` で取得します。以下のプログラミングを参考にしてください。

```
Printer printer = null;
printer = new Printer(Printer.TM_T88, Printer.MODEL_ANK, this);
try {
    printer.connect("TCP:192.168.192.168", Print.PARAM_DEFAULT);
    printer.disconnect();
}
catch (Epos2Exception e) {
    //エラーステータスの取得
    int ErrorStatus = ((Epos2Exception) e).getErrorStatus();
}
```

# API リファレンス

本章では、Epson ePOS SDK for Android で用意されている API について説明しています。

## ePOS SDK API一覧

Epson ePOS SDK for Android には、以下のクラスが用意されています。



- プリンターや周辺機器によって、使用可能な API やパラメーターが異なります。  
[サポートクラス一覧](#)を参照してください。
- 各クラスの持つ API とプリンターごとの対応一覧は、[サポート API 一覧](#)を参照してください。

- [□ Printer クラス](#)
- [□ LineDisplay クラス](#)
- [□ Keyboard クラス](#)
- [□ POSKeyboard クラス](#)
- [□ BarcodeScanner クラス](#)
- [□ MSR クラス](#)
- [□ SimpleSerial クラス](#)
- [□ CashChanger クラス](#)
- [□ CAT クラス](#)
- [□ OtherPeripheral クラス](#)
- [□ CommBox クラス](#)
- [□ クラス共通 API](#)
- [□ Discovery クラス](#)
- [□ Epos2Exception クラス](#)
- [□ Log クラス](#)
- [□ EasySelect クラス](#)
- [□ EasySelectInfo クラス](#)

### Printer クラス

プリンターへの印刷を制御します。

印字モードには、スタンダードモードとページモードがあります。

- スタンダードモード  
1 行単位で印字する印字モードです。文字サイズ、画像、バーコードなどの高さに合わせて改行量が調整されるため、印字量によって用紙の長さが変化するレシート印刷に適しています。
- ページモード  
印字領域を設定して印字データ（文字 / 画像 / バーコード等）を展開し、一括印字する印字モードです。  
ページモードでの処理を行う際は、[addPageBegin](#) でページモード処理を開始し、[addPageEnd](#) でページモード処理を終了してください。

API		説明	スタンダードモード	ページモード
初期化	Printer	Printer クラス初期化	-	-
通信経路	connect	プリンターと接続	-	-
	disconnect	プリンターと切断	-	-
状態監視	startMonitor	ステータスイベントの通知を有効化	✓	✓
	stopMonitor	ステータスイベントの通知を無効化	✓	✓
状態取得	getStatus	接続プリンターの状態を取得	✓	✓
送信	sendData	プリンターにコマンドを送信	✓	✓
トランザクション	beginTransaction	トランザクションの開始	✓	✓
	endTransaction	トランザクションの終了	✓	✓
印刷ジョブ	requestPrintJobStatus	印刷結果を取得	✓	✓
バッファークリア	clearCommandBuffer	命令バッファーをクリア	✓	✓
テキスト	addTextAlign	位置揃え設定を命令バッファーに追加	✓	-
	addLineSpace	改行量設定を命令バッファーに追加	✓	✓
	addTextRotate	倒立印字設定を命令バッファーに追加	✓	-
	addText	文字印字を命令バッファーに追加	✓	✓
	addTextLang	言語設定を命令バッファーに追加	✓	✓
	addTextFont	文字フォント設定を命令バッファーに追加	✓	✓
	addTextSmooth	文字スムージング設定を命令バッファーに追加	✓	✓
	addTextSize	文字倍率設定を命令バッファーに追加	✓	✓
	addTextStyle	文字装飾設定を命令バッファーに追加	✓	✓
	addHPosition	文字印字位置設定を命令バッファーに追加	✓	✓
紙送り	addFeedUnit	ドット単位の紙送りを命令バッファーに追加	✓	✓
	addFeedLine	行単位の紙送りを命令バッファーに追加	✓	✓
グラフィック	addImage	ラスターイメージ印字を命令バッファーに追加	✓	✓
	addLogo	NV ロゴ印字を命令バッファーに追加	✓	✓
バーコード	addBarcode	バーコード印字を命令バッファーに追加	✓	✓
	addSymbol	2 次元シンボル印字を命令バッファーに追加	✓	✓
罫線	addHLine	横罫線印字を命令バッファーに追加	✓	-
	addVLineBegin	縦罫線開始を命令バッファーに追加	✓	-
	addVLineEnd	縦罫線終了を命令バッファーに追加	✓	-

API		説明	スタンダードモード	ページモード
ページモード	<a href="#">addPageBegin</a>	ページモード開始を命令バッファーに追加	✓	-
	<a href="#">addPageEnd</a>	ページモード終了を命令バッファーに追加	-	✓
	<a href="#">addPageArea</a>	ページモード印字領域設定を命令バッファーに追加	-	✓
	<a href="#">addPageDirection</a>	ページモード印字方向設定を命令バッファーに追加	-	✓
	<a href="#">addPagePosition</a>	ページモード印字位置設定を命令バッファーに追加	-	✓
	<a href="#">addPageLine</a>	ページモード直線描画を命令バッファーに追加	-	✓
	<a href="#">addPageRectangle</a>	ページモード四角形描画を命令	-	✓
カット	<a href="#">addCut</a>	用紙カットを命令バッファーに追加	✓	-
ドロアー	<a href="#">addPulse</a>	ドロアーキックを命令バッファーに追加	✓	-
ブザー	<a href="#">addSound</a>	ブザー鳴動を命令バッファーに追加	✓	-
ラベル紙 / ブラック マーク紙	<a href="#">addFeedPosition</a>	紙送りを命令バッファーに追加	✓	-
	<a href="#">addLayout</a>	用紙レイアウトの設定を命令バッファーに追加	✓	-
コマンド	<a href="#">addCommand</a>	コマンドを命令バッファーに追加	✓	✓

API		説明
メンテナンスカウンター	<a href="#">getMaintenanceCounter</a>	カウンター値を取得
	<a href="#">resetMaintenanceCounter</a>	カウンター値をリセット
プリンター設定	<a href="#">getPrinterSetting</a>	プリンター設定の設定値を取得
	<a href="#">setPrinterSetting</a>	プリンター設定を変更
強制送信	<a href="#">forceRecover</a>	強制送信関数
	<a href="#">forcePulse</a>	
	<a href="#">forceStopSound</a>	
	<a href="#">forceCommand</a>	
	<a href="#">forceReset</a>	
結果受信	<a href="#">setStatusChangeEventListener</a>	イベントのリスナー登録メソッドを登録
	<a href="#">setReceiveEventListener</a>	
監視間隔	<a href="#">interval</a>	ステータス監視間隔

## LineDisplay クラス

カスタマーディスプレイへの文字表示制御を行います。

API		説明
初期化	<a href="#">LineDisplay</a>	LineDisplay クラス初期化
通信経路	<a href="#">connect</a>	カスタマーディスプレイと接続
	<a href="#">disconnect</a>	カスタマーディスプレイと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
送信	<a href="#">sendData</a>	カスタマーディスプレイにコマンドを送信
バッファークリア	<a href="#">clearCommandBuffer</a>	命令バッファーをクリア
リセット	<a href="#">addInitialize</a>	カスタマーディスプレイを初期化
ウィンドウ	<a href="#">addCreateWindow</a>	命令バッファーへ実行処理を追加
	<a href="#">addDestroyWindow</a>	
	<a href="#">addSetCurrentWindow</a>	
	<a href="#">addClearCurrentWindow</a>	
カーソル	<a href="#">addSetCursorPosition</a>	
	<a href="#">addMoveCursorPosition</a>	
	<a href="#">addSetCursorType</a>	
テキスト表示	<a href="#">addText</a>	
	<a href="#">addReverseText</a>	
	<a href="#">addMarqueeText</a>	
表示促成	<a href="#">addSetBlink</a>	
	<a href="#">addSetBrightness</a>	
時計	<a href="#">addShowClock</a>	
コマンド送信	<a href="#">addCommand</a>	
結果受信	<a href="#">setReceiveEvent Listener</a>	イベントのリスナー登録メソッドを登録

## Keyboard クラス

キーボードによる文字入力の制御を行います。

API		説明
初期化	<a href="#">Keyboard</a>	Keyboard クラス初期化
通信経路	<a href="#">connect</a>	キーボードと接続
	<a href="#">disconnect</a>	キーボードと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
文字列設定	<a href="#">setPrefix</a>	連続した文字列として扱う条件の設定

API		説明
文字列設定取得	<a href="#">getPrefix</a>	設定している接頭辞の取得
キー検出	<a href="#">setKeyPressEventListener</a>	
文字列検出	<a href="#">setReadStringEventListener</a>	イベントのリスナー登録メソッドを登録

## POSKeyboard クラス

POS キーボードによる文字入力を制御します。

API		説明
初期化	<a href="#">PosKeyboard</a>	POSKeyboard クラス初期化
通信経路	<a href="#">connect</a>	キーボードと接続
	<a href="#">disconnect</a>	キーボードと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
キー検出	<a href="#">setKeyPressEventListener</a>	イベントのリスナー登録メソッドを登録

## BarcodeScanner クラス

バーコードスキャナーによるバーコード読み取りを制御します。

API		説明
初期化	<a href="#">BarcodeScanner</a>	BarcodeScanner クラス初期化
通信経路	<a href="#">connect</a>	バーコードスキャナーと接続
	<a href="#">disconnect</a>	バーコードスキャナーと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
入力データ受信	<a href="#">setScanEventListener</a>	イベントのリスナー登録メソッドを登録

## MSR クラス

MSR によるカード読み取りを制御します。

API		説明
初期化	<a href="#">Msr</a>	MSR クラス初期化
通信経路	<a href="#">connect</a>	MSR と接続
	<a href="#">disconnect</a>	MSR と切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
データ検出	<a href="#">setDataEventListener</a>	イベントのリスナー登録メソッドを登録

## SimpleSerial クラス

プリンターとデバイスのシリアル通信を制御します。

API		説明
初期化	<a href="#">SimpleSerial</a>	SimpleSerial クラス初期化
通信経路	<a href="#">connect</a>	シリアルと接続
	<a href="#">disconnect</a>	シリアルと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
コマンド送信	<a href="#">sendCommand</a>	任意コマンドを送信
入力データ受信	<a href="#">setReceiveEventListener</a>	イベントのリスナー登録メソッドを登録

## CashChanger クラス

自動つり銭機を用いた出入金処理を制御します。

デバイス制御プログラムから制御する場合と、デバイス制御スクリプトから制御する場合で、使用できる API が異なります。

### デバイス制御プログラム

API		説明
初期化	<a href="#">CashChanger</a>	CashChanger クラス初期化
通信経路	<a href="#">connect</a>	自動つり銭機と接続
	<a href="#">disconnect</a>	自動つり銭機と切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
エラー取得	<a href="#">getOposErrorCode</a>	デバイスベンダー定義のエラーコードを取得
計数	<a href="#">readCashCount</a>	デバイス内現金の金額を取得
入金	<a href="#">beginDeposit</a>	入金処理を開始
	<a href="#">pauseDeposit</a>	入金処理を一時停止
	<a href="#">restartDeposit</a>	入金処理を再開
	<a href="#">endDeposit</a>	入金処理を終了
出金	<a href="#">dispenseChange</a>	金額を指定して出金
	<a href="#">dispenseCash</a>	金種を指定して出金
コマンド送信	<a href="#">sendDirectIOCommand</a>	OPOS の DirectIO メソッドを送信

API		説明
結果受信	setCashCountEventListener	イベントのリスナー登録メソッドを登録
	setDepositEventListener	
	setDispenseEventListener	
	setDirectIOCommandReplyEventListener	
	setDirectIOEventListener	
	setStatusUpdateEventListener	

## デバイス制御スクリプト

API		説明
初期化	CashChanger	CashChanger クラス初期化
通信経路	connect	自動つり銭機と接続
	disconnect	自動つり銭機と切断
状態取得	getStatus	接続デバイスの状態を取得
計数	setConfigCountMode	計数モードを変更
	setConfigLeftCash	デバイス内の残置金額を指定
	readCashCount	デバイス内現金の金額を取得
入金	beginDeposit	入金処理を開始
	pauseDeposit	入金処理を一時停止
	restartDeposit	入金処理を再開
	endDeposit	入金処理を終了
出金	dispenseChange	金額を指定して出金
	dispenseCash	金種を指定して出金
回収	collectCash	デバイス内の現金を回収
ドロアー	openDrawer	キャッシュドロアーを開く
コマンド送信	sendCommand	任意コマンドを送信
結果受信	setCashCountEventListener	イベントのリスナー登録メソッドを登録
	setDepositEventListener	
	setDispenseEventListener	
	setConfigChangeEventLister	
	setCollectEventListener	
	setCommandReplyEventListener	
	setStatusChangeEventListener	

## CAT クラス

信用照会端末での電子決済処理を制御します。

API		説明
初期化	<a href="#">Cat</a>	CAT クラス初期化
通信経路	<a href="#">connect</a>	信用照会端末と接続
	<a href="#">disconnect</a>	信用照会端末と切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
エラー取得	<a href="#">getOposErrorCode</a>	デバイスベンダー定義のエラーコードを取得
売上	<a href="#">authorizeSales</a>	売上処理を実行
取消	<a href="#">authorizeVoid</a>	取消処理を実行
返品	<a href="#">authorizeRefund</a>	返品処理を実行
承認後売上	<a href="#">authorizeCompletion</a>	承認後売上処理を実行
日計	<a href="#">accessDailyLog</a>	日計処理を実行
コマンド送信	<a href="#">sendDirectIOCommand</a>	OPOS の DirectIO メソッドを送信
タイムアウト	<a href="#">setTimeout</a>	決済メディア読み取りタイムアウトを指定
	<a href="#">getTimeout</a>	決済メディア読み取りタイムアウトを取得
動作モード	<a href="#">setTrainingMode</a>	動作モードの切り替え
	<a href="#">getTrainingMode</a>	動作モードを取得
結果受信	<a href="#">setAuthorizeSalesEventListener</a>	イベントのリスナー登録メソッドを登録
	<a href="#">setAuthorizeVoidEventListener</a>	
	<a href="#">setAuthorizeRefundEventListener</a>	
	<a href="#">setAuthorizeCompletionEventListener</a>	
	<a href="#">setAccessDailyLogEventListener</a>	
	<a href="#">setDirectIOCommandReplyEventListener</a>	
	<a href="#">setStatusUpdateEventListener</a>	

## OtherPeripheral クラス

開発したデバイス制御プログラム経由で、周辺機器を制御します。

API		説明
初期化	<a href="#">OtherPeripheral</a>	OtherPeripheral クラス初期化
通信経路	<a href="#">connect</a>	デバイスと接続
	<a href="#">disconnect</a>	デバイスと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得

API		説明
送信	<a href="#">sendData</a>	周辺機器へデータを送信
結果受信	<a href="#">setReceiveEventListener</a>	イベントのリスナー登録メソッドを登録

## CommBox クラス

コミュニケーションボックスを使用したデータ送受信を制御します。

API		説明
初期化	<a href="#">CommBox</a>	CommBox クラス初期化
通信経路	<a href="#">connect</a>	コミュニケーションボックスと接続
	<a href="#">disconnect</a>	コミュニケーションボックスと切断
状態取得	<a href="#">getStatus</a>	接続デバイスの状態を取得
送信履歴	<a href="#">getCommHistory</a>	メッセージ送信履歴を取得
送信	<a href="#">sendMessage</a>	メッセージをコミュニケーションボックスに送信
受信	<a href="#">setReceiveEventListener</a>	イベントのリスナー登録メソッドを登録

## クラス共通 API

各クラス共通で使用できる API クラスです。

API		説明
管理情報	<a href="#">getAdmin</a>	管理者情報の取得
	<a href="#">getLocation</a>	設置場所情報の取得
再接続	<a href="#">setConnectionEventListener</a>	イベントのリスナー登録メソッドを登録

## Discovery クラス

デバイス検索機能を制御します。

API		説明
開始	<a href="#">start</a>	デバイス検索を開始
終了	<a href="#">stop</a>	デバイス検索を終了

## Epos2Exception クラス

例外発生時のエラーステータス取得を制御します。

API		説明
エラーステータス取得	<a href="#">getErrorHandlerStatus</a>	例外発生のスロー時に、エラーステータスを取得

## Log クラス

ログ出力機能を制御します。

API		説明
ログ	<a href="#">setLogSettings</a>	ログ出力機能の設定
バージョン取得	<a href="#">SdkVersion</a>	Epson ePOS SDK のバージョン取得

## EasySelect クラス

EasySelect クラスと EasySelectInfo クラスは、NFC タグまたは QR コードを使ってプリンターを選択するための API とメンバー変数です。

EasySelect クラスは、NFC タグと QR コードのデータを解析します。

API		説明
NFC タグ解析	<a href="#">parseNFC</a>	NFC タグデータの解析
QR コード解析	<a href="#">parseQR</a>	QR コードデータの解析
QR コード作成	<a href="#">createQR</a>	parseQR で解析可能な QR コードデータの作成

## EasySelectInfo クラス

[parseNFC](#) または [parseQR](#) を使って解析されたデータを格納するクラスです。このメンバー変数の内容に基づいて、[connect](#) API を呼び出すことでプリンターと接続できます。

メンバー変数		説明
プリンター種類	<a href="#">deviceType</a>	プリンターの種類を格納
プリンターナー名	<a href="#">printerName</a>	プリンターナー名を格納
MAC アドレス / BD アドレス	<a href="#">macAddress</a>	MAC アドレスまたは BD アドレスを格納

# Printer クラス

## Printer

Printer クラスを初期化します。

### 構文

```
public Printer(int printerSeries, int lang, Context context)
throws Epos2Exception
```

### パラメーター

#### printerSeries

対象のプリンターを指定します。

設定値	説明
Printer.TM_M10	TM-m10
Printer.TM_M30	TM-m30
Printer.TM_P20	TM-P20
Printer.TM_P60II	TM-P60II (Peeler)
Printer.TM_P80	TM-P80
Printer.TM_T20	TM-T20, TM-T20II, TM-T20III
Printer.TM_T70	TM-T70, TM-T70-i, TM-T70II, TM-T70II-DT, TM-T70II-DT2
Printer.TM_T88	TM-T88IV, TM-T88V, TM-T88VI, TM-T88V-i, TM-T88V-DT, TM-T88VI-DT2
Printer.TM_T90	TM-T90, TM-T90II
Printer.TM_T90KP	TM-T90KP
Printer.TM_L90	TM-L90

#### lang

プリンターの言語仕様を指定します。

設定値	説明
Printer.MODEL_JAPANESE	日本語モデル

#### context

アプリケーションのコンテキストを指定します。

ログ出力機能を使用せず、かつ USB 接続以外の場合に null を指定できます。

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_UNSUPPORTED	サポートしていない機種名または言語仕様が指定された。

## 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。

OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。複数のプリンターと接続して運用する場合、それぞれの機種に合わせてインスタンスを生成してください。

## connect

プリンターとの通信を開始します。

### 構文

```
Public void connect(String target, int timeout) throws  
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

システム構成によって接続方法が異なります。

- TM プリンターモデル

<接続形式>:<識別子>

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCP"	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>プリンターホスト名</li> </ul>	"TCP:192.168.192.168"
Bluetooth®	"BT"	BD アドレス	"BT:00:22:15:7D:70:9C"
USB	"USB"	デバイスノード	"USB:/dev/udev/xxxxxxxxxx"

- カスタマーディスプレイモデル

<接続形式>:<識別子>

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCP"	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>プリンターホスト名</li> </ul>	"TCP:192.168.192.168"
Bluetooth®	"BT"	BD アドレス	"BT:00:22:15:7D:70:9C"
USB	"USB"	デバイスノード	"USB:/dev/udev/xxxxxxxxxx"

SSL/TLS 通信でプリンターと接続するには、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCPS"	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>プリンターホスト名</li> </ul>	"TCPS:192.168.192.168[local_printer]"

- TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデル /  
TM-m30 または TM-T88VI+ バーコードスキャナーモデル  
Wi-Fi/Ethernet のネットワーク通信で接続する場合は、以下の構文で指定してください。  
<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCP"	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>プリンターホスト名</li> </ul>	"TCP:192.168.192.168[local_printer]"

デバイス ID は [local\_printer] 固定です。

Bluetooth® または USB で接続する場合は、以下の構文で指定してください。

<接続形式>:<識別子>

I/F	接続形式	識別子	例
Bluetooth®	"BT"	BD アドレス	"BT:00:22:15:7D:70:9C"
USB	"USB"	デバイスノード	"USB:/dev/udev/xxxxxxxxxx"

- POS ターミナルモデル

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	<ul style="list-style-type: none"> <li>"TCP"</li> <li>"TCPS"</li> </ul>	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>プリンターホスト名</li> </ul>	"TCP:192.168.192.168[local_printer]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

## timeout

プリンターと通信確立するための最大待ち時間 (msec 単位) を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間 (msec 単位)
Print.PARAM_DEFAULT	既定値 (15000) を指定

---

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	デバイスが見つからない。
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- プリンターとの通信が不要になった場合、必ず [disconnect](#) を呼び出し、プリンターとの通信を終了してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンターステータスは、Printer クラスで登録したイベントに通知されます。  
詳細は、2 章「プログラミングフロー」、6 章「一台のプリンターを複数のモバイル端末から使用するには」を参照してください。
- デバイス ID は、周辺機器を TM プリンター（親機）へ登録する際に設定する任意の文字列です。  
詳細は、各プリンターの詳細取扱説明書を参照してください。
- プリンターの IP アドレスを DHCP に設定している場合、識別子に MAC アドレスまたはプリンターホスト名を指定してください。
- 識別子にプリンターホスト名を指定する場合、DNS サーバーからプリンターホスト名が検索可能な環境で使用してください。
- 本 API は TM プリンター（親機）と周辺機器の接続状態に関わらず成功します。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

プリンターとの通信を終了します。

### 構文

```
Public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## startMonitor

プリンターのステータスイベントの通知を有効化します。

`interval` プロパティーで指定した間隔でプリンターステータスを取得、更新し、`setStatusChangeEventListener` に登録したリスナー登録メソッドへ通知します。

### 構文

```
public void startMonitor() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の `Epos2Exception` が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

### 補足説明

- 本 API はプリンターとの通信開始後（`connect` API 実行後）に使用してください。
- ESC/POS 制御時は、`interval` プロパティーで指定した更新間隔の間で変化のあった ASB 情報を全て通知します。  
ePOS-Device XML 制御時は、更新時点での ASB 情報を通知します。

## stopMonitor

ステータスイベントを無効化します。

### 構文

```
public void stopMonitor() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public PrinterStatusInfo getStatus ()
```

### 戻り値

現在の状態が PrinterStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
connection : 接続状態	Printer.TRUE	接続中
	Printer.FALSE	状態が不明
online : オンライン状態	Printer.TRUE	オンライン
	Printer.FALSE	オフライン
	Printer.UNKNOWN	状態が不明
coverOpen : カバー状態	Printer.TRUE	カバーが開いている
	Printer.FALSE	カバーが閉じている
	Printer.UNKNOWN	状態が不明
paper : 用紙状態	Printer.PAPER_OK	用紙あり
	Printer.PAPER_NEAR_END	ニアエンド
	Printer.PAPER_EMPTY	用紙なし
	Printer.UNKNOWN	状態が不明
paperFeed : 紙送り	Printer.TRUE	紙送り中
	Printer.FALSE	停止中
	Printer.UNKNOWN	状態が不明
panelSwitch : パネルスイッチ (Feed ボタン) 状態	Printer.SWITCH_ON	押している
	Printer.SWITCH_OFF	押していない
	Printer.UNKNOWN	状態が不明
drawer : バッテリー残量によるオフライン 状態 (モバイルモデルのみ)	Printer.DRAWER_HIGH	バッテリー残量が少ないためオフ ライン状態
	Printer.DRAWER_LOW	オフライン状態ではない
	Printer.UNKNOWN	状態が不明
drawer : ドローアキックコネクター 3 番 ピン状態 (モバイルモデル以外)	Printer.DRAWER_HIGH	High 状態
	Printer.DRAWER_LOW	Low 状態
	Printer.UNKNOWN	状態が不明

状態	ステータス	説明
errorStatus : エラー状態	Printer.NO_ERR	正常
	Printer.MECHANICAL_ERR	メカニカルエラー発生
	Printer.AUTOCUTTER_ERR	オートカッターエラー発生
	Printer.UNRECOVER_ERR	復帰不可能エラー発生
	Printer.AUTORECOVER_ERR	自動復帰エラー発生
	Printer.UNKNOWN	状態が不明
autoRecoverError : 自動復帰エラー状態	Printer.HEAD OVERHEAT	ヘッド高温エラー
	Printer.MOTOR OVERHEAT	モータードライバー IC 高温エラー
	Printer.BATTERY OVERHEAT	バッテリー高温エラー
	Printer.WRONG PAPER	用紙エラー
	Printer.COVER_OPEN	カバーが開いている
	Printer.UNKNOWN	状態不明
buzzer : ブザー鳴動状態	Printer.TRUE	鳴動中（対応機種のみ）
	Printer.FALSE	停止中（対応機種のみ）
	Printer.UNKNOWN	状態が不明
adapter : AC アダプター接続状態	Printer.TRUE	接続中
	Printer.FALSE	切断中
	Printer.UNKNOWN	状態が不明
batteryLevel : バッテリー残量	Printer.BATTERY_LEVEL_6	バッテリー残量 6
	Printer.BATTERY_LEVEL_5	バッテリー残量 5
	Printer.BATTERY_LEVEL_4	バッテリー残量 4
	Printer.BATTERY_LEVEL_3	バッテリー残量 3
	Printer.BATTERY_LEVEL_2	バッテリー残量 2
	Printer.BATTERY_LEVEL_1	バッテリー残量 1 (ニアエンド)
	Printer.BATTERY_LEVEL_0	バッテリー残量 0 (リアルエンド)
	Printer.UNKNOWN	状態が不明

## 補足説明

- 通信不可状態で本 API が呼び出された場合、接続状態以外のプリンターステータスは全て Printer.UNKNOWN が格納されます。
- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。

## sendData

印刷コマンドを送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、Printer クラスの [setReceiveEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

add 系 API (例 [addText](#)) でバッファリングしたデータを送信します。

---

### 構文

```
public void sendData (int timeout) throws Epos2Exception
```

---

### パラメーター

#### timeout

印刷完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
5000 ~ 600000 の整数	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	印刷コマンドがバッファリングされていない。 通信が開始されていない状態で、本 API が呼び出された。

---

### 補足説明

Bluetooth® 接続の場合、オフライン状態が検出できずタイムアウトエラーになることがあります。

6 章「一台のプリンターを複数のモバイル端末から使用するには」を参照してください。

## beginTransaction

トランザクションを開始します。

トランザクションは、一枚のレシートや、一枚のクーポンなど、ひとまとめの印刷内容を示します。

本 API を呼び出した後、[endTransaction](#) でトランザクションを終了するまでをひとまとめの印刷内容として扱います。

### 構文

```
public void beginTransaction() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。 既に本関数によりトランザクションが開始されていた。
ERR_FAILURE	その他のエラーが発生した。

## endTransaction

トランザクションを終了します。

トランザクションは、一枚のレシートや、一枚のクーポンなど、ひとまとめの印刷内容を示します。

[beginTransaction](#) を呼び出した後、本 API でトランザクションを終了するまでをひとまとめの印刷内容として扱います。

### 構文

```
public void endTransaction() throws Eswv2Exception
```

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。 トランザクションが開始されていない状態で、本 API が呼び出された。
ERR_FAILURE	その他のエラーが発生した。

## requestPrintJobStatus

指定した印刷ジョブ ID の印刷結果を取得します。

### 構文

```
public void requestPrintJobStatus (String printJobId) throws  
Epos2Exception
```

### パラメーター

#### printJobId

印刷ジョブ ID を指定します。

使用可能な文字は 1 ~ 30 文字の英数字、アンダースコア、ハイフン、ピリオドです。

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信エラー。
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

### 補足説明

- 本 API の結果は、Printer クラスの [setReceiveEventListener API](#) で設定したリスナー登録メソッドへ通知されます。
- 同じ印刷ジョブ ID で、複数の印刷処理を実行した場合、一番新しいジョブのステータスを取得します。

## clearCommandBuffer

命令バッファーをクリアします。

バッファリングされた命令バッファーは、本 API を実行するまで保存されます。

### 構文

```
public void clearCommandBuffer()
```

## addTextAlign

位置揃え設定を命令バッファーに追加します。

### 構文

```
public void addTextAlign(int align) throws Epos2Exception
```

---

### パラメーター

#### align

位置揃えを指定します。

設定値	説明
Printer.ALIGN_LEFT (初期値)	左揃え
Printer.ALIGN_CENTER	中央揃え
Printer.ALIGN_RIGHT	右揃え
Printer.PARAM_DEFAULT	既定値（左揃え）を選択

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- 本 API は「行の先頭」で使用してください。先頭以外で使用した場合、無効になります。
- 本 API の設定は、バーコード /2 次元シンボルにも適用されます。
- 印字モードがページモードで位置揃え設定をする場合、本 API ではなく、[addPagePosition](#) で設定してください。

## addLineSpace

改行量設定を命令バッファーに追加します。

### 構文

```
public void addLineSpace(int linespc) throws Epos2Exception
```

---

### パラメーター

#### linespc

改行量（ドット単位）を指定します。

設定値	説明
0 ~ 255 の整数	改行量（ドット単位）

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

1 行の改行量を印字文字より小さい値に設定すると、印字のために設定量より多く紙送りする場合があります。

## addTextRotate

倒立印字設定を命令バッファーに追加します。

### 構文

```
public void addTextRotate(int rotate) throws Epos2Exception
```

---

### パラメーター

#### rotate

倒立印字の有無を指定します。

設定値	説明
Printer.TRUE	倒立印字を指定
Printer.FALSE (初期値)	倒立印字を解除
Printer.PARAM_DEFAULT	既定値（倒立印字解除）を選択

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- 本 API は「行の先頭」で使用してください。先頭以外で使用した場合、無効になります。
- 本 API の設定は、バーコード /2 次元シンボルにも適用されます。
- 印字モードがページモードで倒立印字を設定する場合、本 API ではなく、[addPageDirection API](#) で設定してください。

## addText

文字の印字を命令バッファーに追加します。

### 構文

```
public void addText(String data) throws Epos2Exception
```

---

### パラメーター

#### data

印字する文字列を指定します。

水平タブ / 改行は、以下のエスケープシーケンスを使用します。

文字列	説明
\t	水平タブ (HT)
\n	改行 (LF)
\\	バックスラッシュ

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- テキストの印字後、テキスト以外を印字する場合、改行または紙送りを実行してください。  
末尾が改行で終わっていない行は、未確定データとして次の [sendData](#) で破棄されます。
- ページモードでは、文字のベースラインドットを基準に現在の印字位置へ印字文字を展開します。

## addTextLang

言語設定を命令バッファーに追加します。

本 API で指定された言語情報に従って、[addText](#) API で指定された文字列をエンコードします。

### 構文

```
public void addTextLang(int lang) throws Epos2Exception
```

### パラメーター

#### lang

対象言語を指定します。

設定値	説明
Printer.LANG_EN (初期値)	英語 (ANK 仕様)
Printer.LANG_JA	日本語
Printer.LANG_ZH_CN	簡体字中国語
Printer.LANG_ZH_TW	繁体字中国語
Printer.LANG_KO	韓国語
Printer.LANG_TH	タイ語 (南アジア仕様)
Printer.LANG_VI	ベトナム語 (南アジア仕様)
Printer.PARAM_DEFAULT	既定値 (英語) を選択する

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- 本 API は、[addText](#) API を呼び出す前に呼び出す API です。
- 印字可能言語は、使用するプリンターの文字仕様に依存します。各プリンターの詳細取扱説明書を参照してください。

## addTextFont

文字のフォント設定を命令バッファーに追加します。

### 構文

```
public void addTextFont(int font) throws Epos2Exception
```

---

### パラメーター

#### font

フォントを指定します。

設定値	説明
Printer.FONT_A (初期値)	フォントA
Printer.FONT_B	フォントB
Printer.FONT_C	フォントC
Printer.FONT_D	フォントD
Printer.FONT_E	フォントE
Printer.PARAM_DEFAULT	既定値（フォントA）を選択

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addTextSmooth

スムージング設定を命令バッファーに追加します。

### 構文

```
public void addTextSmooth(int smooth) throws  
Epos2Exception
```

### パラメーター

#### smooth

スムージングの有無を指定します。

設定値	説明
Printer.TRUE	スムージングを指定
Printer.FALSE (初期値)	スムージングを解除
Printer.PARAM_DEFAULT	既定値 (スムージング解除) を選択

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addTextSize

文字の倍率設定を命令バッファーに追加します。

### 構文

```
public void addTextSize(int width, int height) throws  
Epos2Exception
```

### パラメーター

#### width

文字の横倍率を指定します。

設定値	説明
1 ~ 8 の整数	横方向の倍率を指定（初期値：1）
Printer.PARAM_DEFAULT	既定値（1）を指定
Printer.PARAM_UNSPECIFIED	設定を変更しない

#### height

文字の縦倍率を指定します。

設定値	説明
1 ~ 8 の整数	縦方向の倍率を指定（初期値：1）
Printer.PARAM_DEFAULT	既定値（1）を指定
Printer.PARAM_UNSPECIFIED	設定を変更しない

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

全てのパラメーターに Printer.PARAM\_UNSPECIFIED を指定した場合、ERR\_PARAM を返します。

## addTextStyle

文字の装飾設定を命令バッファーに追加します。

### 構文

```
public void addTextStyle(int reverse, int ul, int em, int color)
throws Epos2Exception
```

### パラメーター

#### reverse

白黒反転文字を指定します。

設定値	説明
Printer.TRUE	白黒反転文字を指定
Printer.FALSE (初期値)	白黒反転文字を解除
Printer.PARAM_UNSPECIFIED	設定を変更しない
Printer.PARAM_DEFAULT	既定値（白黒反転文字を解除）を選択

#### ul

アンダーラインを指定します。

設定値	説明
Printer.TRUE	アンダーラインを指定
Printer.FALSE (初期値)	アンダーラインを解除
Printer.PARAM_UNSPECIFIED	設定を変更しない
Printer.PARAM_DEFAULT	既定値（アンダーラインを解除）を選択

#### em

太字を指定します。

設定値	説明
Printer.TRUE	太字を指定
Printer.FALSE (初期値)	太字を解除
Printer.PARAM_UNSPECIFIED	設定を変更しない
Printer.PARAM_DEFAULT	既定値（太字を解除）を選択

#### color

色を指定します。

設定値	説明
Printer.COLOR_NONE	非印字

設定値	説明
Printer.COLOR_1 (初期値)	第1色
Printer.COLOR_2	第2色
Printer.COLOR_3	第3色
Printer.COLOR_4	第4色
Printer.PARAM_UNSPECIFIED	設定を変更しない
Printer.PARAM_DEFAULT	既定値（第1色）を選択

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

全てのパラメーターに Printer.PARAM\_UNSPECIFIED を指定した場合、ERR\_PARAM を返します。

## addHPosition

横方向の印字開始位置を命令バッファーに追加します。

### 構文

```
public void addHPosition(int x) throws Epos2Exception
```

---

### パラメーター

x

横方向の印字開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の印字開始位置（ドット単位）

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- 本 API により、プリンターは『行の先頭』ではない状態になります。  
X に 0 を指定した場合でも『行の先頭』ではない状態になります。
- 本 API 実行後、[addTextAlign](#)、[addTextRotate](#) は使用できません。

## addFeedUnit

ドット単位の紙送りを命令バッファーに追加します。

### 構文

```
public void addFeedUnit(int unit) throws Epos2Exception
```

### パラメーター

#### unit

紙送り量（ドット単位）を指定します。

設定値	説明
0 ~ 255 の整数	紙送り量（ドット単位）

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

本 API により、プリンターは『行の先頭』状態になります。

## addFeedLine

行単位の紙送りを命令バッファーに追加します。

### 構文

```
public void addFeedLine(int line) throws Epos2Exception
```

### パラメーター

#### line

紙送り量（行単位）を指定します。

設定値	説明
0 ~ 255 の整数	紙送り量（行単位）

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

本 API により、プリンターは『行の先頭』状態になります。

## addImage

ラスターイメージの印字を命令バッファーに追加します。

Android.graphics.Bitmap クラスのグラフィックを印字します。

Android.graphics.Bitmap クラスのグラフィックのうち、指定範囲を mode、halftone、brightness パラメーターで二値化し、ラスターイメージデータに変換します。

Compress パラメーターの設定にしたがって、圧縮送信または非圧縮送信を設定します。

画像の 1 ピクセルがプリンターの 1 ドットに相当します。透明色が含まれている場合、画像の背景を白とみなします。

---

### 構文

```
public void addImage(Bitmap data, int x, int y, int width, int height, int color, int mode, int halftone, double brightness, int compress) throws Epos2Exception
```

---

### パラメーター

#### data

android.graphics.Bitmap クラスのインスタンスを指定します。

#### x

印字範囲の横方向の開始位置（ピクセル単位）を指定します。

設定値	説明
0 ~ 65534 の整数	印字範囲の横方向の開始位置（ピクセル単位）

#### y

印字範囲の縦方向の開始位置（ピクセル単位）を指定します。

設定値	説明
0 ~ 65534 の整数	印字範囲の縦方向の開始位置（ピクセル単位）

#### width

印字範囲の幅（ピクセル単位）を指定します。

設定値	説明
1 ~ 65535 の整数	印字範囲の幅（ピクセル単位）

#### height

印字範囲の高さ（ピクセル単位）を指定します。

設定値	説明
1 ~ 65535 の整数	印字範囲の高さ（ピクセル単位）

**color**

色を指定します。

設定値	説明
Printer.COLOR_NONE	非印字
Printer.COLOR_1	第1色
Printer.COLOR_2	第2色
Printer.COLOR_3	第3色
Printer.COLOR_4	第4色
Printer.PARAM_DEFAULT	既定値(第1色)を選択

**mode**

カラー モードを指定します。

設定値	説明
Printer.MODE_MONO	モノクロ(2階調)
Printer.MODE_GRAY16	多階調(16階調)
Printer.MODE_MONO_HIGH_DENSITY	モノクロ(2階調)倍密度
Printer.PARAM_DEFAULT	既定値(モノクロ(2階調))を選択

**halftone**

ハーフトーン処理方法を指定します。

設定値	説明
Printer.HALFTONE_DITHER	ディザ(グラフィックのみの印刷に適している。)
Printer.HALFTONE_ERROR_DIFFUSION	誤差拡散(文字とグラフィックが混在する印刷に適している。)
Printer.HALFTONE_THRESHOLD	しきい値(文字のみの印刷に適している。)
Printer.PARAM_DEFAULT	既定値(ディザ)を選択

モノクロ(2階調)印刷の場合のみ有効です。

**brightness**

明るさ補正值を指定します。

設定値	説明
0.1 ~ 10.0 の実数	明るさ補正值(ガンマ値)
Printer.PARAM_DEFAULT	既定値(1.0)

明るさ補正值に1.0以外を指定した場合は、処理時間が遅くなります。

**compress**

圧縮処理の有無を指定します。

圧縮処理すると、印刷画像の白スジを防止できます。

プリンターと Bluetooth<sup>®</sup> 通信で接続している場合に効果があります。

設定値	説明
Printer.COMPRESS_DEFLATE	画像を圧縮する。
Printer.COMPRESS_NONE	画像を非圧縮する。
Printer.COMPRESS_AUTO	圧縮、非圧縮を内部で自動設定する。
Printer.PARAM_DEFAULT	既定値 (圧縮、非圧縮を内部で自動設定する)

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- x/yパラメーターとwidth/heightパラメーターで指定された領域がdataパラメーターで指定した画像のサイズに収まっていない場合、例外に ERR\_PARAM が返されます。
- compress は、下記プリンターの Bluetooth<sup>®</sup> モデルでのみ指定できます。該当プリンター以外は、Printer.COMPRESS\_AUTO を指定してください。

モデル	ファームウェアバージョン
TM-m10	制限なし
TM-m30	制限なし
TM-P20	制限なし
TM-P60II (Peeler)	6.14 ESC/POS 以降
TM-P80	制限なし
TM-T20II	制限なし
TM-T20III	制限なし
TM-T70II	制限なし
TM-T88V	制限なし
TM-T88VI	制限なし

- 透過画像を印字する場合、印字速度が遅くなる場合があります。
- ページモードでは多階調印字、画像圧縮をサポートしていません。これらを指定した場合は何も印字しません。
- プリンターに合わせた適切な画像サイズを指定してください。大きな画像を指定した場合、API 上は成功しますが、印刷されないことがあります。
- Android 端末の仕様によって印字可能な画像サイズであっても ERR\_MEMORY が発生する場合があります。この場合は画像サイズを小さくしてください。

## addLogo

NV ロゴの印字を命令バッファーに追加します。  
プリンターの NV メモリーに登録されているロゴを印字します。

### 構文

```
public void addLogo(int key1, int key2) throws  
Epos2Exception
```

### パラメーター

#### key1

NV ロゴのキーコード 1 を指定します。

設定値	説明
1 ~ 255 の整数	NV ロゴのキーコード 1

#### key2

NV ロゴのキーコード 2 を指定します。

設定値	説明
1 ~ 255 の整数	NV ロゴのキーコード 2

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- NV ロゴの登録方法は、各プリンターの詳細取扱説明書を参照してください。
- ページモードでは多階調印字をサポートしていません。スタンダードモードでのみ多階調グラフィックスの印字が可能です。
- 本 API で指定する NV ロゴは、事前にバッファリングした [addTextStyle](#) の色設定で印刷されます。

## addBarcode

バーコード印字を命令バッファーに追加します。

### 構文

```
public void addBarcode(String data, int type, int hri, int font,
int width, int height) throws Epos2Exception
```

### パラメーター

#### data

バーコードデータを文字列で指定します。

Type で指定するバーコードの規格に従った文字列を指定してください。

種類	説明
UPC-A	11 行の数字を指定した場合、チェックデジットを自動的に付加します。 12 行の数字を指定した場合、12 行目をチェックデジットとして処理しますが、チェックデジットの検算は行いません。
UPC-E	最初の行に 0 を指定してください。 2 ~ 6 行目にメーカーコードを指定してください。 7 ~ 11 行目にアイテムコードを右詰めで指定してください。 アイテムコードの行数はメーカーコードにより異なります。 使用しない行は 0 を指定してください。 11 行の数字を指定した場合、チェックデジットを自動的に付加します。 12 行の数字を指定した場合、12 行目をチェックデジットとして処理しますが、チェックデジットの検算は行いません。
EAN13	12 行の数字を指定した場合、チェックデジットを自動的に付加します。
JAN13	13 行の数字を指定した場合、13 行目をチェックデジットとして処理しますが、チェックデジットの検算は行いません。
EAN8	7 行の数字を指定した場合、チェックデジットを自動的に付加します。
JAN8	8 行の数字を指定した場合、8 行目をチェックデジットとして処理しますが、チェックデジットの検算は行いません。
CODE39	先頭の文字が * の場合、この文字をスタートキャラクターとして処理します。 それ以外の場合、スタートキャラクターを自動的に付加します。
ITF	スタートコードおよびストップコードを自動的に付加します。 チェックデジットの付加および検算は行いません。
CODABAR	スタートキャラクター (A ~ D, a ~ d) を指定してください。 ストップキャラクター (A ~ D, a ~ d) を指定してください。 チェックデジットの付加および検算は行いません。
CODE93	スタートキャラクターおよびストップキャラクターを自動的に付加します。 チェックデジットを計算して自動的に付加します。

種類	説明
CODE128	<p>スタートキャラクター (CODE A, CODE B, CODE C) を指定してください。 ストップキャラクターを自動的に付加します。 チェックデジットを計算して自動的に付加します。 以下の文字をエンコードするには、文字 { で始まる 2 文字を指定してください。</p> <ul style="list-style-type: none"> <li>• FNC1 : {1</li> <li>• FNC2 : {2</li> <li>• FNC3 : {3</li> <li>• FNC4 : {4</li> <li>• CODE A : {A</li> <li>• CODE B : {B</li> <li>• CODE C : {C</li> <li>• SHIFT : {S</li> <li>• { : {{</li> </ul> <p>CODE C を指定する場合、バーコードデータはエスケープシーケンスのコントロールコードとして指定してください。</p>
GS1-128	<p>スタートキャラクター、FNC1、チェックデジット、ストップキャラクターを自動的に付加します。ただし、データの区切りを示す FNC1 は付加しません。 アプリケーション識別子 (AI) とそれに続くデータのチェックデジットを計算して自動的に付加するには、チェックデジットの位置に文字 * を指定します。 アプリケーション識別子 (AI) を括弧で囲むことができます。 括弧は HRI の印字文字として使用し、データとしてエンコードしません。 アプリケーション識別子 (AI) とデータの間に空白を挿入することができます。 空白は HRI の印字文字として使用し、データとしてエンコードしません。 以下の文字をエンコードするには、文字 { で始まる 2 文字を指定してください。</p> <ul style="list-style-type: none"> <li>• FNC1 : {1</li> <li>• FNC3 : {3</li> <li>• ( : {{</li> <li>• ) : }}</li> <li>• * : {*}*</li> <li>• { : {{</li> </ul>
GS1 DataBar Omnidirectional	
GS1 DataBar Truncated	アプリケーション識別子 (AI) とチェックデジットを除く 13 桁の商品識別番号 (GTIN) を指定してください。
GS1 DataBar Limited	
GS1 DataBar Expanded	<p>アプリケーション識別子 (AI) を括弧で囲むことができます。 括弧は HRI の印字文字として使用し、データとしてエンコードしません。 以下の文字をエンコードするには、文字 { で始まる 2 文字を 指定してください。</p> <ul style="list-style-type: none"> <li>• FNC1 : {1</li> <li>• ( : {{</li> <li>• ) : }}</li> </ul>

文字列で表現できないバイナリーデータを指定する場合、以下のエスケープシーケンスで指定します。

文字列	説明
\xnn	コントロールコード (nn は 16 進数で指定)
\\"	バックスラッシュ

## type

バーコードの種類を指定します。

設定値	説明
Printer.BARCODE_UPC_A	UPC-A
Printer.BARCODE_UPC_E	UPC-E
Printer.BARCODE_EAN13	EAN13
Printer.BARCODE_JAN13	JAN13
Printer.BARCODE_EAN8	EAN8
Printer.BARCODE_JAN8	JAN8
Printer.BARCODE_CODE39	CODE39
Printer.BARCODE_ITF	ITF
Printer.BARCODE_CODABAR	CODABAR
Printer.BARCODE_CODE93	CODE93
Printer.BARCODE_CODE128	CODE128
Printer.BARCODE_GS1_128	GS1-128
Printer.BARCODE_GS1_DATABAR_OMNIDIRECTIONAL	GS1 DataBar Omnidirectional
Printer.BARCODE_GS1_DATABAR_TRUNCATED	GS1 DataBar Truncated
Printer.BARCODE_GS1_DATABAR_LIMITED	GS1 DataBar Limited
Printer.BARCODE_GS1_DATABAR_EXPANDED	GS1 Databar Expanded

## hri

HRI の位置を指定します。

設定値	説明
Printer.HRI_NONE (初期値)	印字しない
Printer.HRI ABOVE	バーコードの上
Printer.HRI BELOW	バーコードの下
Printer.HRI BOTH	バーコードの上と下の両方
Printer.PARAM_DEFAULT	既定値（印字しない）を選択
Printer.PARAM_UNSPECIFIED	設定を変更しない

## font

HRI フォントを指定します。

設定値	説明
Printer.FONT_A (初期値)	フォント A
Printer.FONT_B	フォント B
Printer.FONT_C	フォント C
Printer.FONT_D	フォント D
Printer.FONT_E	フォント E
Printer.PARAM_DEFAULT	既定値 (フォント A) を選択
Printer.PARAM_UNSPECIFIED	設定を変更しない

## width

1 モジュールの幅をドット単位で指定します。

設定値	説明
2 ~ 6 の整数	1 モジュールの幅 (ドット単位)
Printer.PARAM_UNSPECIFIED	設定を変更しない

## height

バーコードの高さをドット単位で指定します。

設定値	説明
1 ~ 255 の整数	バーコードの高さ (ドット単位)
Printer.PARAM_UNSPECIFIED	設定を変更しない

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 本 API は、「行の先頭」で使用してください。
- data で指定したバーコードデータが、type で指定するバーコード規格に従っていない場合、例外でエラーを返さず、バーコードは印刷されません。

## addSymbol

2次元シンボル印字を命令バッファーに追加します。

### 構文

```
public void addSymbol(String data, int type, int level, int width, int height, int size) throws Epos2Exception
```

### パラメーター

#### data

2次元シンボルデータを文字列で指定します。

Type で指定する 2次元シンボルの規格に従った文字列を指定してください。

種類	説明
Standard PDF417	文字列を UTF-8 に変換後、エスケープシーケンスの処理を行い、エンコードします。
Truncated PDF417	データ領域の最大コードワード数は 928 個、1 段あたりのデータ領域の最大コードワード数は 30 個、最大段数は 90 段です。
QR Code Model 1	文字列をシフト JIS に変換後、エスケープシーケンスの処理を行い、データの種類を以下の中から選択してエンコードします。
QR Code Model 2	<ul style="list-style-type: none"> <li>• 数字 : 0 ~ 9</li> <li>• 英数字 : 0 ~ 9, A ~ Z, スペース, \$, %, *, +, -, ., /, :</li> <li>• 漢字 : シフト JIS で表現できる漢字</li> <li>• 8 ビットバイトデータ : 0x00 ~ 0xff</li> </ul>
QR Code Micro	
MaxiCode Mode 2	文字列を UTF-8 に変換後、エスケープシーケンスの処理を行い、エンコードします。
MaxiCode Mode 3	モード 2 およびモード 3 の場合、最初のデータが []>\x1e01\x1dyy (yy は 2 衔の数字) の場合、これをメッセージヘッダーとして処理し、次のデータからプライマリメッセージとして処理します。
MaxiCode Mode 4	それ以外の場合、最初のデータからプライマリメッセージとして処理します。
MaxiCode Mode 5	モード 2 の場合、プライマリメッセージを以下の形式で指定してください。
MaxiCode Mode 6	<ul style="list-style-type: none"> <li>• 郵便コード (1 ~ 9 衔の数字) GS:(\x1d)</li> <li>• ISO 国名コード (1 ~ 3 衔の数字) GS:(\x1d)</li> <li>• サービスクラスコード (1 ~ 3 衔の数字)</li> </ul> <p>モード 3 の場合、プライマリメッセージを以下の形式で指定してください。</p> <ul style="list-style-type: none"> <li>• 郵便コード (1 ~ 6 個のコードセット A で変換可能なデータ) GS (\x1d)</li> <li>• ISO 国名コード (1 ~ 3 衔の数字) GS (\x1d)</li> <li>• サービスクラスコード (1 ~ 3 衔の数字)</li> </ul>
GS1 DataBar Stacked	文字列を UTF-8 に変換後、エスケープシーケンスの処理を行い、エンコードします。
GS1 DataBar Stacked Omnidirectional	アプリケーション識別子 (AI) とチェックデジットを除く 13 衔の商品識別番号 (GTIN) を指定してください。

種類	説明
GS1 DataBar Expanded Stacked	文字列を UTF-8 に変換後、エスケープシーケンスの処理を行い、エンコードします。 アプリケーション識別子 (AI) を括弧で囲むことができます。 括弧は HRI の印字文字として使用し、データとしてエンコードしません。 以下の文字をエンコードするには、文字「で始まる 2 文字を指定してください。 <ul style="list-style-type: none"><li>• FNC1 : {1</li><li>• ( : {(</li><li>• ) : })</li></ul>
Aztec Code	文字列を UTF-8 に変換後、エスケープシーケンスの処理を行い、エンコードします。
DataMatrix	文字列を UTF-8 に変換後、エスケープシーケンスの処理を行い、エンコードします。

文字列で表現できないバイナリーデータを指定する場合、以下のエスケープシーケンスで指定します。

文字列	説明
\xnn	コントロールコード (nn は 16 進数で指定)
\\"	バックスラッシュ

## type

2 次元シンボルの種類を指定します。

設定値	説明
Printer.SYMBOL_PDF417_STANDARD	Standard PDF417
Printer.SYMBOL_PDF417_TRUNCATED	Truncated PDF417
Printer.SYMBOL_QRCODE_MODEL_1	QR Code Model 1
Printer.SYMBOL_QRCODE_MODEL_2	QR Code Model 2
Printer.SYMBOL_QRCODE_MICRO	QR Code Micro
Printer.SYMBOL_MAXICODE_MODE_2	MaxiCode Mode 2
Printer.SYMBOL_MAXICODE_MODE_3	MaxiCode Mode 3
Printer.SYMBOL_MAXICODE_MODE_4	MaxiCode Mode 4
Printer.SYMBOL_MAXICODE_MODE_5	MaxiCode Mode 5
Printer.SYMBOL_MAXICODE_MODE_6	MaxiCode Mode 6
Printer.SYMBOL_GS1_DATABAR_STACKED	GS1 DataBar Stacked
Printer.SYMBOL_GS1_DATABAR_STACKED_O MNIDIRECTIONAL	GS1 DataBar Stacked Omnidirectional
Printer.SYMBOL_GS1_DATABAR_EXPANDED_ STACKED	GS1 DataBar Expanded Stacked
Printer.SYMBOL_AZTECCODE_FULLRANGE	Aztec Code Full-Range モード
Printer.SYMBOL_AZTECCODE_COMPACT	Aztec Code Compact モード
Printer.SYMBOL_DATAMATRIX_SQUARE	DataMatrix 正方形

設定値	説明
Printer.SYMBOL_DATAMATRIX_RECTANGLE_8	DataMatrix 長方形、行数 8
Printer.SYMBOL_DATAMATRIX_RECTANGLE_12	DataMatrix 長方形、行数 12
Printer.SYMBOL_DATAMATRIX_RECTANGLE_16	DataMatrix 長方形、行数 16

### level

エラー訂正レベルを指定します。

2次元シンボルの種類に合わせて選択してください。

MaxiCode, 2次元 GS1 DataBar, DataMatrix の場合、PARAM\_DEFAULT を選択します。

#### □ PDF417

設定値	説明
Printer.LEVEL_0	エラー訂正レベル 0
Printer.LEVEL_1	エラー訂正レベル 1
Printer.LEVEL_2	エラー訂正レベル 2
Printer.LEVEL_3	エラー訂正レベル 3
Printer.LEVEL_4	エラー訂正レベル 4
Printer.LEVEL_5	エラー訂正レベル 5
Printer.LEVEL_6	エラー訂正レベル 6
Printer.LEVEL_7	エラー訂正レベル 7
Printer.LEVEL_8	エラー訂正レベル 8
Printer.PARAM_DEFAULT	既定値（エラー訂正レベル 1）を選択
Printer.PARAM_UNSPECIFIED	設定を変更しない

#### □ QR Code

設定値	説明
Printer.LEVEL_L	エラー訂正レベル L
Printer.LEVEL_M	エラー訂正レベル M
Printer.LEVEL_Q	エラー訂正レベル Q
Printer.LEVEL_H	エラー訂正レベル H
Printer.PARAM_DEFAULT	既定値（エラー訂正レベル M）を選択
Printer.PARAM_UNSPECIFIED	設定を変更しない

#### □ Aztec Code

設定値	説明
5～95 の整数	エラー訂正レベル（パーセント単位）
Printer.PARAM_DEFAULT	既定値（エラー訂正レベル 23）を選択
Printer.PARAM_UNSPECIFIED	設定を変更しない

**width**

モジュールの幅を指定します。

1 ~ 255 の整数値で指定可能です。

Printer.PARAM\_UNSPECIFIED を指定した場合、設定を変更しません。

2 次元シンボルの種類によって範囲が変更されます。

2 次元シンボルの種類	有効な値	初期値
PDF417	2 ~ 8	3
QR Code	3 ~ 16	3
MaxiCode	1 ~ 255 (無視されます)	
2 次元 GS1 DataBar	2 ~ 8	2
Aztec Code	2 ~ 16	3
DataMatrix	2 ~ 16	3

**height**

モジュールの高さを指定します。

1 ~ 255 の整数値で指定可能です。

Printer.PARAM\_UNSPECIFIED を指定した場合、設定を変更しません。

2 次元シンボルの種類によって範囲が変更されます。

2 次元シンボルの種類	有効な値	初期値
PDF417	2 ~ 8 (width に対する倍率)	3
QR Code		
MaxiCode		
2 次元 GS1 DataBar	1 ~ 255 (無視されます)	
Aztec Code		
DataMatrix		

**size**

2 次元シンボルの最大サイズを指定します。

0 ~ 65535 の整数値で指定可能です。

Printer.PARAM\_UNSPECIFIED を指定した場合、設定を変更しません。

2 次元シンボルの種類によって範囲が変更されます。

2 次元シンボルの種類	初期値	説明	
PDF417	0 (自動)	1 段あたりのコードワード数を指定	
QR Code	0 ~ 65535 (無視されます)		
MaxiCode	0 ~ 65535 (無視されます)		
2 次元 GS1 DataBar	Expanded Stacked	0 (自動)	2 次元シンボルの最大幅を指定 (106 以上)
	その他	0 ~ 65535 (無視されます)	
Aztec Code	0 ~ 65535 (無視されます)		
DataMatrix	0 ~ 65535 (無視されます)		

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 本 API は、「行の先頭」で使用してください。
- data で指定した 2 次元シンボルデータが、type で指定する 2 次元シンボルの種類に従っていない場合、例外でエラーを返さず、2 次元シンボルは印刷されません。
- ESC/POS 制御時、width と height パラメーターに有効な値の範囲外を指定すると初期値で印刷されます。
- ePOS-Device XML 制御時、width と height パラメーターに有効な値の範囲外を指定すると [sendData](#) の例外に ERR\_FAILURE が返ります。

## addHLine

横罫線印字を命令バッファーに追加します。

横罫線の描画をします。

### 構文

```
public void addHLine(int x1, int x2, int lineStyle) throws  
Epos2Exception
```

### パラメーター

x1

横罫線の描画開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横罫線の描画開始位置（ドット単位）

x2

横罫線の描画終了位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横罫線の描画終了位置（ドット単位）

lineStyle

罫線の種類を指定します。

設定値	説明
Printer.LINE_THIN	実線：細
Printer.LINE_MEDIUM	実線：中太
Printer.LINE_THICK	実線：太
Printer.LINE_THIN_DOUBLE	二重線：細
Printer.LINE_MEDIUM_DOUBLE	二重線：中太
Printer.LINE_THICK_DOUBLE	二重線：太
Printer.PARAM_DEFAULT	既定値（実線：細）を選択

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

## 補足説明

- ページモードでは使用できません。
- ページモードでの横罫線は、[addPageLine](#) を使用してください。

## addVLineBegin

縦墨線の開始を命令バッファーに追加します。

縦墨線の描画を開始します。

### 構文

```
public void addVLineBegin(int x, int lineStyle, int[] lineId)
throws Epos2Exception
```

### パラメーター

**x**

縦墨線の描画開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦墨線の描画開始位置（ドット単位）

**lineStyle**

墨線の種類を指定します。

設定値	説明
Printer.LINE_THIN	実線：細
Printer.LINE_MEDIUM	実線：中太
Printer.LINE_THICK	実線：太
Printer.LINE_THIN_DOUBLE	二重線：細
Printer.LINE_MEDIUM_DOUBLE	二重線：中太
Printer.LINE_THICK_DOUBLE	二重線：太
Printer.PARAM_DEFAULT	既定値（実線：細）を選択

**lineId**

本関数で印字する墨線の ID が整数で返ります。

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

## 補足説明

- ページモードでは使用できません。
- ページモードでの縦罫線は、[addPageLine](#) を使用してください。
- 縦罫線は、[addVLineEnd](#) API で終了を指定するまで描画します。
- 本 API は [addVLineEnd](#) API と一緒に使用してください。

## addVLineEnd

縦罫線の終了を命令バッファーに追加します。

縦罫線の描画を終了します。

### 構文

```
public void addVLineEnd(int lineId) throws Epos2Exception
```

### パラメーター

#### lineId

[addVLineBegin](#) で取得した lineId を指定します。

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- ページモードでは使用できません。
- ページモードでの縦罫線は、[addPageLine](#) を使用してください。
- 縦罫線は、本 API で終了を指定するまで描画します。
- 本 API は [addVLineBegin](#) API と一緒に使用してください。

## addPageBegin

ページモード開始を命令バッファーに追加します。

ページモードの処理が開始します。

### 構文

```
public void addPageBegin() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- ページモードでは、多階調印字をサポートしていません。
- 本 API は [addPageEnd](#) API と一緒に使用してください。

## addPageEnd

ページモード終了を命令バッファーに追加します。

ページモードの処理が終了します。

### 構文

```
public void addPageEnd() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- ページモードでは、多階調印字をサポートしていません。
- 本 API は [addPageBegin](#) APIと一緒に使用してください。

## addPageArea

ページモード印字領域を命令バッファーに追加します。

ページモード印字領域（座標）を指定します。本 API に続けて、[addText API](#) など印刷データの API を指定します。

### 構文

```
public void addPageArea(int x, int y, int width, int height)
throws Epos2Exception
```

### パラメーター

#### x

横方向の原点（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の原点（ドット単位）

0 はプリンターの印字可能領域の左端になります。

#### y

縦方向の原点（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦方向の原点（ドット単位）

0 は紙送りをしていない位置です。

#### width

印字領域の幅（ドット単位）を指定します。

設定値	説明
1 ~ 65535 の整数	印字領域の幅（ドット単位）

#### height

印字領域の高さ（ドット単位）を指定します。

設定値	説明
1 ~ 65535 の整数	印字領域の高さ（ドット単位）

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。

エラーステータス	説明
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 印字内容に合わせて印字領域を指定してください。印字データが印字領域をはみ出た場合、印字データが途中で切れた印字結果になります。
- 本 API は [addPageBegin API](#) と [addPageEnd API](#) に挟んで使用してください。
- 印字領域の幅と高さは、印字方向の設定に合わせて確定してください。  
印字データが切れてしまう場合があります。印字方向の設定は、[addPageDirection](#) で指定します。
- スタンダードモードでは機能しません。

## addPageDirection

ページモード印字方向設定を命令バッファーに追加します。

ページモードの印字方向を指定します。

### 構文

```
public void addPageDirection(int direction) throws  
Epos2Exception
```

### パラメーター

#### direction

ページモードの印字方向を指定します。

設定値	説明
Printer.DIRECTION_LEFT_TO_RIGHT (初期値)	回転しない（左上を始点に右方向へ印字）
Printer.DIRECTION_BOTTOM_TO_TOP	反時計回り 90 度回転（左下を始点に上方向へ印字）
Printer.DIRECTION_RIGHT_TO_LEFT	180 度回転（右下を始点に左方向へ印字）
Printer.DIRECTION_TOP_TO_BOTTOM	時計回り 90 度回転（右上を始点に下方向へ印字）
Printer.PARAM_DEFAULT	既定値（回転しない）を選択

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- スタンダードモードでは機能しません。
- 本 API は [addPageBegin API](#) と [addPageEnd API](#) に挟んで使用してください。

## addPagePosition

ページモード印字位置領域内の印字位置設定を命令バッファーに追加します。

[addPageArea API](#) で指定したエリア内の、印字開始位置（座標）を指定します。

---

### 構文

```
public void addPagePosition(int x, int y) throws  
Epos2Exception
```

---

### パラメーター

x

横方向の印字位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の印字位置（ドット単位）
Printer.PARAM_UNSPECIFIED	設定を変更しない

y

縦方向の印字位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦方向の印字位置をドット単位で指定
Printer.PARAM_UNSPECIFIED	設定を変更しない

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- スタンダードモードでは機能しません。
- 本 API は [addPageBegin API](#) と [addPageEnd API](#) に挟んで使用してください。
- 印字開始位置（座標）は、印字内容に合わせて指定します。以下を参考にしてください。

印字内容	指定方法
文字列を印字する場合	最初の文字のベースライン左端を指定します。標準の大きさで左詰めで印字する場合は省略可能です。高さが 2 倍の文字を印刷する場合は、y を 42 以上に指定します。
バーコードを印字する場合	シンボルの左下を指定します。y にバーコードの高さを指定します。
グラフィック / ロゴを印字する場合	グラフィックデータの左下を指定します。y にグラフィックデータの高さを指定します。
2 次元シンボルを印字する場合	シンボルの左上を指定します。左上から印字する場合は、省略可能です。

□ 全てのパラメーターに Printer.PARAM\_UNSPECIFIED を指定した場合、ERR\_PARAM を返します。

## addPageLine

ページモードの直線描画を命令バッファーに追加します。

ページモードで直線を描画します。

### 構文

```
public void addPageLine(int x1, int y1, int x2, int y2, int
lineStyle) throws Epos2Exception
```

### パラメーター

**x1**

横方向の描画開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の描画開始位置（ドット単位）

**y1**

縦方向の描画開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦方向の描画開始位置（ドット単位）

**x2**

横方向の描画終了位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の描画終了位置（ドット単位）

**y2**

縦方向の描画終了位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦方向の描画終了位置（ドット単位）

**lineStyle**

墨線の種類を指定します。

設定値	説明
Printer.LINE_THIN	実線：細
Printer.LINE_MEDIUM	実線：中太
Printer.LINE_THICK	実線：太
Printer.LINE_THIN_DOUBLE	二重線：細
Printer.LINE_MEDIUM_DOUBLE	二重線：中太

設定値	説明
Printer.LINE_THICK_DOUBLE	二重線：太
Printer.PARAM_DEFAULT	既定値（実線：細）を選択

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- スタンダードモードでは機能しません。
- 斜線は描画できません。
- 本 API は [addPageBegin API](#) と [addPageEnd API](#) に挟んで使用してください。

## addPageRectangle

ページモードの四角形描画を命令バッファーに追加します。

ページモードで四角形を描画します。

### 構文

```
public void addPageRectangle(int x1, int y1, int x2, int y2, int
lineStyle) throws Epos2Exception
```

### パラメーター

**x1**

横方向の描画開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の描画開始位置（ドット単位）

**y1**

縦方向の描画開始位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦方向の描画開始位置（ドット単位）

**x2**

横方向の描画終了位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	横方向の描画終了位置（ドット単位）

**y2**

縦方向の描画終了位置（ドット単位）を指定します。

設定値	説明
0 ~ 65535 の整数	縦方向の描画終了位置（ドット単位）

**lineStyle**

線の種類を指定します。

設定値	説明
Printer.LINE_THIN	実線：細
Printer.LINE_MEDIUM	実線：中太
Printer.LINE_THICK	実線：太
Printer.LINE_THIN_DOUBLE	二重線：細
Printer.LINE_MEDIUM_DOUBLE	二重線：中太

設定値	説明
Printer.LINE_THICK_DOUBLE	二重線：太
Printer.PARAM_DEFAULT	既定値（実線：細）を選択

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- スタンダードモードでは機能しません。
- 本 API は [addPageBegin](#) API と [addPageEnd](#) API に挟んで使用してください。

## addCut

用紙カットを命令バッファーに追加します。

用紙カットを設定します。

### 構文

```
public void addCut(int type) throws Epos2Exception
```

### パラメーター

#### type

用紙カット方法を指定します。

設定値	説明
Printer.CUT_FEED	フィードカット（紙送り後カット）
Printer.CUT_NO_FEED	フィード無しカット（紙送りせずにカット）
Printer.CUT_RESERVE	カット予約（後に続く印字を実行後、カット位置でカット）
Printer.PARAM_DEFAULT	既定値（フィードカット（紙送り後カット））を選択

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- ページモードでは使用できません。
- カット予約 (Printer.CUT\_RESERVE) に続く印刷データが指定されていないとき、プリンターは予約したカット位置まで紙を送ってからカットを実行します。
- プリンターによっては、カット予約 (Printer.CUT\_RESERVE) に続く印刷データを約 2 秒間待ってから紙送りの動作を開始します。
- カット予約 (Printer.CUT\_RESERVE) を使用するときは、1 枚のレシートの長さを 20mm 以上に設定してください。

## addPulse

ドローアーキックを命令バッファーに追加します。

ドローアーキックを設定します。

### 構文

```
public void addPulse(int drawer, int time) throws  
Epos2Exception
```

### パラメーター

#### drawer

ドローアーキックコネクターを指定します。

設定値	説明
Printer.DRAWER_2PIN	ドローアーキックコネクター 2 番ピン
Printer.DRAWER_5PIN	ドローアーキックコネクター 5 番ピン
Printer.PARAM_DEFAULT	既定値（ドローアーキックコネクター 2 番ピン）を選択

#### time

ドローアーキック信号のオン時間を指定します。

設定値	説明
Printer.PULSE_100	100 msec の信号
Printer.PULSE_200	200 msec の信号
Printer.PULSE_300	300 msec の信号
Printer.PULSE_400	400 msec の信号
Printer.PULSE_500	500 msec の信号
Printer.PARAM_DEFAULT	既定値（100 msec の信号）を選択

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

□ ページモードでは使用できません。

- ドロアーと外付けオプションブザーは同時に接続する事はできません。
- 短時間でドロアーオープンを繰り返さないでください。ドロアーに負荷がかかり、破損するおそれがあります。
- 以下のプリンターの内蔵ブザー機能付きモデルは、ドロアーキックコネクターへのパルス出力で内蔵ブザーを鳴動できます。  
内蔵ブザー制御の詳細は、各プリンターの詳細取扱説明書を参照してください。
  - ・ TM-T70
  - ・ TM-T70II
  - ・ TM-T88IV
  - ・ TM-T88V
  - ・ TM-T88VI
  - ・ TM-T90II
  - ・ TM-L90

## addSound

ブザーの鳴動設定を命令バッファーに追加します。  
ブザーを設定します。

### 構文

```
public void addSound(int pattern, int repeat, int cycle) throws  
Epos2Exception
```

### パラメーター

#### pattern

ブザーの音色を指定します。

設定値	説明
Printer.PATTERN_NONE	停止（モバイルモデル）
Printer.PATTERN_NONE	停止。ただし鳴動中のブザーは停止しない。 (モバイルモデル以外)
Printer.PATTERN_A	パターン A (外付けオプションブザー)
Printer.PATTERN_B	パターン B (外付けオプションブザー)
Printer.PATTERN_C	パターン C (外付けオプションブザー)
Printer.PATTERN_D	パターン D (外付けオプションブザー)
Printer.PATTERN_E	パターン E (外付けオプションブザー)
Printer.PATTERN_ERROR	エラー鳴動パターン (外付けオプションブザー)
Printer.PATTERN_PAPER_EMPTY	紙無し鳴動パターン (外付けオプションブザー)
Printer.PATTERN_1	パターン 1 (内蔵ブザー)
Printer.PATTERN_2	パターン 2 (内蔵ブザー)
Printer.PATTERN_3	パターン 3 (内蔵ブザー)
Printer.PATTERN_4	パターン 4 (内蔵ブザー)
Printer.PATTERN_5	パターン 5 (内蔵ブザー)
Printer.PATTERN_6	パターン 6 (内蔵ブザー)
Printer.PATTERN_7	パターン 7 (内蔵ブザー)
Printer.PATTERN_8	パターン 8 (内蔵ブザー)
Printer.PATTERN_9	パターン 9 (内蔵ブザー)
Printer.PATTERN_10	パターン 10 (内蔵ブザー)
Printer.PARAM_DEFAULT	既定値 (パターン A) を選択

#### repeat

繰り返し回数を指定します。

設定値	説明
0	無限
1 ~ 255	1 ~ 255 回
Printer.PARAM_DEFAULT	既定値（1回）を選択

### cycle

ブザーを鳴らす周期（msec 単位）を指定します。

設定値	説明
1000 ~ 25500	1000 ~ 25500 msec
Printer.PARAM_DEFAULT	既定値（1000 msec）を選択

パターン1～パターン10の場合のみ有効です。

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- ページモードでは使用できません。
- ドロアーと外付けオプションブザーは同時に接続する事はできません。
- 本 API はプリンターにブザーが付いてなければ使用できません。
- プリンターにより [sendData](#) API のコールバックを受信するタイミングが異なります。  
モバイルモデル：[sendData](#) API 実行後  
モバイルモデル以外：鳴動終了後
- 以下のプリンターの内蔵ブザー機能付きモデルは、[addPulse](#) で内蔵ブザーを鳴動できます。
  - ・ TM-T70
  - ・ TM-T70II
  - ・ TM-T88IV
  - ・ TM-T88V
  - ・ TM-T88VI
  - ・ TM-T90II
  - ・ TM-L90

## addFeedPosition

ラベル紙／ブラックマーク紙の紙送りを命令バッファーに追加します。

### 構文

```
public void addFeedPosition(int position) throws
Epos2Exception
```

### パラメーター

#### position

紙送りする位置を指定します。

設定値	説明
Printer.FEED_PEELING	剥離位置まで紙送り
Printer.FEED_CUTTING	カット位置まで紙送り
Printer.FEED_CURRENT_TOF	現在のラベル頭出し位置まで紙送り
Printer.FEED_NEXT_TOF	次のラベル頭出し位置まで紙送り

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- ラベル紙 / ブラックマーク紙の制御はスタンダードモードで行います。
- ページモードでは、本 API は使用できません。

## addLayout

ラベル紙 / ブラックマーク紙の用紙レイアウト情報を命令バッファーに追加します。

### 構文

```
public void addLayout(int type, int width, int height, int
marginTop, int marginBottom, int offsetCut, int offsetLabel)
throws Epos2Exception
```

### パラメーター

#### type

用紙の種類を指定します。

設定値	説明
Printer.LAYOUT_RECEIPT	レシート紙（ブラックマークなし）
Printer.LAYOUT_RECEIPT_BM	レシート紙（ブラックマークあり）
Printer.LAYOUT_LABEL	ラベル紙（ブラックマークなし）
Printer.LAYOUT_LABEL_BM	ラベル紙（ブラックマークあり）

#### width

用紙幅（0.1 mm 単位）を指定します。

設定値		説明
TM プリンターモデル	POS ターミナルモデル	
1 ~ 10000 の整数	290 ~ 600 の整数	用紙幅（0.1 mm 単位）

#### height

印字基準から次の印字基準までの距離（0.1mm 単位）を指定します。

用紙の種類	設定値		説明
	TM プリンター モデル	POS ターミナ ルモデル	
レシート紙 (ブラックマークなし)	0	0	距離の指定なし（自動）
レシート紙 (ブラックマークあり)			ブラックマーク上端位置から次のブ ラックマーク上端位置までの距離
ラベル紙 (ブラックマークなし)	0 ~ 10000	0 ~ 1550	ラベル上端位置から次のラベル上端 位置までの距離
ラベル紙 (ブラックマークあり)			ブラックマーク下端位置から次のブ ラックマーク下端位置までの距離

## marginTop

印字基準から頭出し位置までの距離（0.1mm 単位）を指定します。

用紙の種類	設定値		説明
	TM プリンター モデル	POS ターミナ ルモデル	
レシート紙 (ブラックマークなし)	0	0	距離の指定なし（自動）
レシート紙 (ブラックマークあり)	-9999～ 10000	-150～1500	ブラックマーク上端位置から頭出し 位置までの距離
ラベル紙 (ブラックマークなし)	0～10000	0～1500	ラベル上端位置から頭出し位置まで の距離
ラベル紙 (ブラックマークあり)	-9999～ 10000	-15～1500	ブラックマーク下端位置から頭出し 位置までの距離

## marginBottom

排出基準から印刷可能領域の下端までの距離（0.1mm 単位）を指定します。

用紙の種類	設定値		説明
	TM プリンター モデル	POS ターミナ ルモデル	
レシート紙 (ブラックマークなし)	0	0	距離の指定なし（自動）
レシート紙 (ブラックマークあり)	0	0	距離の指定なし（自動）
ラベル紙 (ブラックマークなし)	-9999～0	-15～0	ラベル下端位置からの距離 (紙送り方向が正の数)
ラベル紙 (ブラックマークあり)	-9999～ 10000	-15～15	ブラックマーク上端位置からの距離 (紙送り方向が正の数)

## offsetCut

排出基準からカット位置までの距離（0.1mm 単位）を指定します。

用紙の種類	設定値		説明
	TM プリンター モデル	POS ターミナ ルモデル	
レシート紙 (ブラックマークなし)	0	0	距離の指定なし（自動）
レシート紙 (ブラックマークあり)	-9999～ 10000	-290～50	ブラックマーク上端位置からカット 位置までの距離
ラベル紙 (ブラックマークなし)	0～10000	0～50	ラベル下端位置からカット位置まで の距離
ラベル紙 (ブラックマークあり)	0～10000	0～50	ブラックマーク上端位置からカット 位置までの距離

## offsetLabel

排出基準からラベル下端までの距離（0.1mm 単位）を指定します。

用紙の種類	設定値		説明
	TM プリンターモデル	POS ターミナルモデル	
レシート紙 (ブラックマークなし)	0	0	距離の指定なし（自動）
レシート紙 (ブラックマークあり)	0	0	距離の指定なし（自動）
ラベル紙 (ブラックマークなし)	0	0	距離の指定なし（自動）
ラベル紙 (ブラックマークあり)	0 ~ 10000	0 ~ 15	ブラックマーク上端位置からラベル下端位置までの距離

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

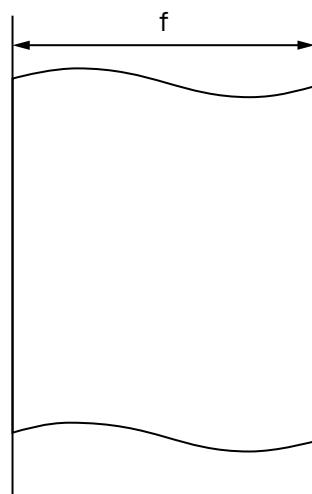
エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

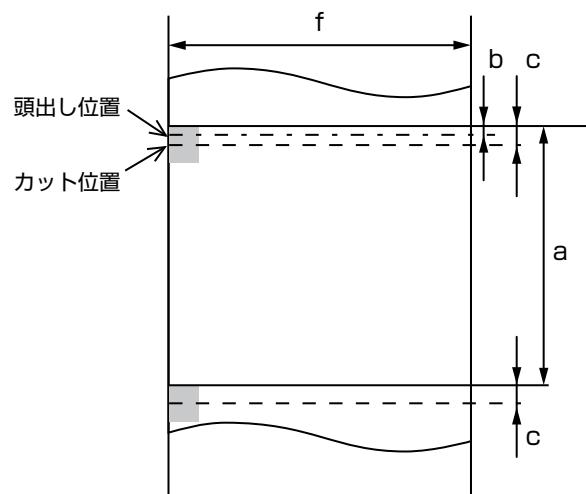
- ページモードでは機能しません。
- TM プリンターモデルに設定可能な値はプリンターによって異なります。詳細は ESC/POS コマンドリファレンスの FS ( L < 機能 33 > ) コマンドを参照してください。  
[https://reference.epson-biz.com/pos/reference\\_ja/](https://reference.epson-biz.com/pos/reference_ja/)
- 用紙の種類に応じた各パラメーターの設定可能値は以下の通りです。  
 表内の値は TM プリンターモデルの設定値です。() 内に POS ターミナルモデルの設定値を記載しています。

パラメーター	レシート紙 (ブラック マークなし)	ラベル紙 (ブラック マークなし)	ラベル紙 (ブラックマー クあり)	レシート紙 (ブラックマーク あり)	図内 記号
width	1 ~ 10000 (290 ~ 600)				f
height	0 (0)	0 ~ 10000 (0 ~ 1550)		a	
marginTop	0 (0)	0 ~ 10000 (0 ~ 1500)	-9999 ~ 10000 (-15 ~ 1500)	-9999 ~ 10000 (-150 ~ 1500)	b
marginBottom	0 (0)	-9999 ~ 0 (-15 ~ 0)	-9999 ~ 10000 (-15 ~ 15)	0 (0)	e
offsetCut	0 (0)	0 ~ 10000 (0 ~ 50)		-9999 ~ 10000 (-290 ~ 50)	c
offsetLabel	0 (0)			0 ~ 10000 (0 ~ 15)	d

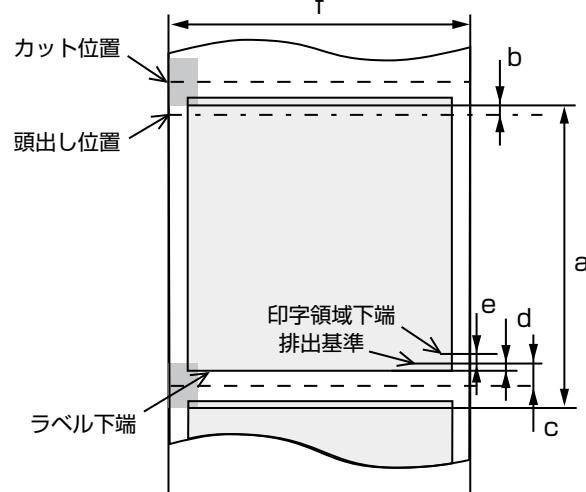
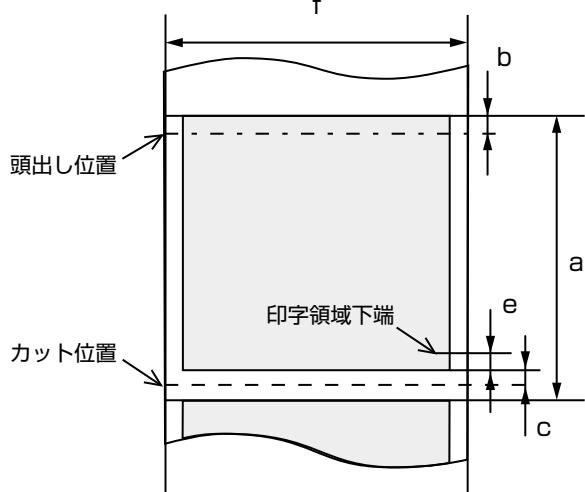
レシート紙



&lt;ブラックマーク付&gt;



ラベル紙



## addCommand

コマンドを命令バッファーに追加します。  
ESC/POS コマンドを送信します。

---

### 構文

```
public void addCommand(byte[] data) throws Epos2Exception
```

---

### パラメーター

#### data

ESC/POS コマンドを指定します。  
バイナリーデータを指定します。

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- ESC/POS コマンドの詳細は以下の URL を参照してください。  
[https://reference.epson-biz.com/modules/ref\\_escpos\\_ja/](https://reference.epson-biz.com/modules/ref_escpos_ja/)
- Epson ePOS SDK は本 API で送信するコマンドをチェックしません。  
Epson ePOS SDK の動作を妨げるようなコマンドを送信した場合、他の API の動作やステータスの値が不正になることがあります。  
ESC/POS コマンドやレシートプリンターの仕様を十分に理解した上で使用してください。

## getMaintenanceCounter

プリンターのメンテナンスカウンターの値を取得します。

本 API で取得する値は、listener パラメーターで指定するリスナー登録メソッドに通知されます。

### 構文

```
public void getMaintenanceCounter(int timeout, int type,
    MaintenanceCounterListener listener) throws Epos2Exception
```

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
5000 ~ 600000 の整数	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

#### type

取得するメンテナンスカウンターの種類を指定します。

設定値	説明
Printer.MaintenanceCounter.PaperFeed	紙送り行数
Printer.MaintenanceCounter.AutoCutter	オートカッターの動作回数

#### listener

本 API の実行結果を返すリスナー登録メソッドを登録します。

```
- (void) onGetMaintenanceCounter(int code, int type, int
value);
```

##### □ code

処理の実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	成功
CODE_ERR_AUTORECOVER	自動復帰エラー発生
CODE_ERR_COVER_OPEN	カバーオープンエラー発生
CODE_ERR_CUTTER	オートカッターエラー発生
CODE_ERR_MECHANICAL	メカニカルエラー発生
CODE_ERR_EMPTY	ロール紙エンド検出器に用紙なし
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_MEMORY	処理に必要なメモリーが確保できなかった。

エラーステータス	説明
CODE_ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
CODE_ERR_PROCESSING	処理が実行できなかった。
CODE_ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。
CODE_ERR_FAILURE	その他のエラーが発生した。

- type  
取得したメンテナンスカウンターの種類が入ります。
- value  
取得したメンテナンスカウンターの値が入ります。

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 本 API は他の API と同時に使用できません。  
本 API 実行後は、コールバックが返るまで他の API を実行しないでください。
- 本 API 実行中はプリンターの電源を切らないでください。
- 本 API は TM プリンターモデルとカスタマーディスプレイモデルのシステム構成で使用できます。  
TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデルのシステム構成では、  
Bluetooth® や USB での接続時に使用できます。
- 本 API は SSL/TLS 通信では使用できません。

## resetMaintenanceCounter

プリンターのメンテナンスカウンターの値をリセットします。

本 API で取得する値は、listener パラメーターで指定するリスナー登録メソッドに通知されます。

### 構文

```
public void resetMaintenanceCounter(int timeout, int type,
                                     MaintenanceCounterListener listener) throws Epos2Exception
```

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
5000 ~ 600000 の整数	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

#### type

リセットするメンテナンスカウンターの種類を指定します。

設定値	説明
Printer.MaintenanceCounter.PaperFeed	紙送り行数
Printer.MaintenanceCounter.AutoCutter	オートカッターの動作回数

#### listener

本 API の実行結果を返すリスナー登録メソッドを登録します。

- (void) onResetMaintenanceCounter(int code, int type);

#### □ code

処理の実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	成功
CODE_ERR_AUTORECOVER	自動復帰エラー発生
CODE_ERR_COVER_OPEN	カバーオープンエラー発生
CODE_ERR_CUTTER	オートカッターエラー発生
CODE_ERR_MECHANICAL	メカニカルエラー発生
CODE_ERR_EMPTY	ロール紙エンド検出器に用紙なし
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_MEMORY	処理に必要なメモリーが確保できなかった。
CODE_ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。

エラーステータス	説明
CODE_ERR_PROCESSING	処理が実行できなかった。
CODE_ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。
CODE_ERR_FAILURE	その他のエラーが発生した。

□ type

リセットしたメンテナンスカウンターの種類が入ります。

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 本 API は他の API と一緒に使用できません。  
本 API 実行後は、コールバックが返るまで他の API を実行しないでください。
- 本 API 実行中はプリンターの電源を切らないでください。
- 本 API の実行は 1 日に 10 回までにしてください。それ以上実行すると、プリンターが使用できなくなることがあります。
- 本 API は TM プリンターモデルとカスタマーディスプレイモデルのシステム構成で使用できます。  
TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデルのシステム構成では、Bluetooth® や USB での接続時に使用できます。
- 本 API は SSL/TLS 通信では使用できません。

## getPrinterSetting

プリンター設定の設定値を取得します。

本 API で取得する値は、listener パラメーターで指定するリスナー登録メソッドに通知されます。

### 構文

```
public void getPrinterSetting(int timeout, int type,
    PrinterSettingListener listener) throws Epos2Exception
```

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
5000 ~ 600000 の整数	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

#### type

取得するプリンター設定の項目を指定します。

設定値	説明
Printer.Settings.PaperWidth	用紙幅
Printer.Settings.PrintDensity	印字濃度
Printer.Settings.PrintSpeed	印字速度

#### listener

本 API の実行結果を返すリスナー登録メソッドを登録します。

- (void) onGetPrinterSetting(int code, int type, int value);

##### □ code

処理の実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	成功
CODE_ERR_AUTORECOVER	自動復帰エラー発生
CODE_ERR_COVER_OPEN	カバーオープンエラー発生
CODE_ERR_CUTTER	オートカッターエラー発生
CODE_ERR_MECHANICAL	メカニカルエラー発生
CODE_ERR_EMPTY	ロール紙エンド検出器に用紙なし
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_MEMORY	処理に必要なメモリーが確保できなかった。

エラーステータス	説明
CODE_ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
CODE_ERR_PROCESSING	処理が実行できなかった。
CODE_ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。
CODE_ERR_FAILURE	その他のエラーが発生した。

□ type

取得したプリンター設定の項目が入ります。

値	説明
Printer.Settings.PaperWidth	用紙幅
Printer.Settings.PrintDensity	印字濃度
Printer.Settings.PrintSpeed	印字速度

□ value

取得したメンテナنسカウンターの値が入ります。

設定値	説明
Printer.Settings.PaperWidth58_0	用紙幅 58mm
Printer.Settings.PaperWidth60_0	用紙幅 60mm
Printer.Settings.PaperWidth80_0	用紙幅 80mm
Printer.Settings.PrintDensityDIP	印字濃度 DIP スイッチ設定
Printer.Settings.PrintDensity70	印字濃度 70%
Printer.Settings.PrintDensity75	印字濃度 75%
Printer.Settings.PrintDensity80	印字濃度 80%
Printer.Settings.PrintDensity85	印字濃度 85%
Printer.Settings.PrintDensity90	印字濃度 90%
Printer.Settings.PrintDensity95	印字濃度 95%
Printer.Settings.PrintDensity100	印字濃度 100%
Printer.Settings.PrintDensity105	印字濃度 105%
Printer.Settings.PrintDensity110	印字濃度 110%
Printer.Settings.PrintDensity115	印字濃度 115%
Printer.Settings.PrintDensity120	印字濃度 120%
Printer.Settings.PrintDensity125	印字濃度 125%
Printer.Settings.PrintDensity130	印字濃度 130%
Printer.Settings.PrintSpeed1	印字速度 レベル 1
Printer.Settings.PrintSpeed2	印字速度 レベル 2
Printer.Settings.PrintSpeed3	印字速度 レベル 3
Printer.Settings.PrintSpeed4	印字速度 レベル 4

設定値	説明
Printer.Setting.PrintSpeed5	印字速度 レベル 5
Printer.Setting.PrintSpeed6	印字速度 レベル 6
Printer.Setting.PrintSpeed7	印字速度 レベル 7
Printer.Setting.PrintSpeed8	印字速度 レベル 8
Printer.Setting.PrintSpeed9	印字速度 レベル 9
Printer.Setting.PrintSpeed10	印字速度 レベル 10
Printer.Setting.PrintSpeed11	印字速度 レベル 11
Printer.Setting.PrintSpeed12	印字速度 レベル 12
Printer.Setting.PrintSpeed13	印字速度 レベル 13
Printer.Setting.PrintSpeed14	印字速度 レベル 14

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 本 API は他の API と同時に使用できません。  
本 API 実行後は、コールバックが返るまで他の API を実行しないでください。
- 本 API 実行中はプリンターの電源を切らないでください。
- 本 API は TM プリンターモデルとカスタマーディスプレイモデルのシステム構成で使用できます。  
TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデルのシステム構成では、  
Bluetooth® や USB での接続時に使用できます。
- 本 API は SSL/TLS 通信では使用できません。

## setPrinterSetting

プリンター設定の設定値を変更します。

本 API で取得する値は、listener パラメーターで指定するリスナー登録メソッドに通知されます。

### 構文

```
public void setPrinterSetting(int timeout,
java.util.Map<java.lang.Integer,?java.lang.Integer> list,
PrinterSettingListener listener) throws
```

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
5000 ~ 600000 の整数	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

#### list

変更するプリンター設定の項目 (type) と設定値 (value) を、組み合わせた List を指定します。

#### type

変更するプリンター設定の項目を指定します。

設定値	説明
Printer.Settings.PaperWidth	用紙幅
Printer.Settings.PrintDensity	印字濃度
Printer.Settings.PrintSpeed	印字速度

#### value

変更するプリンター設定の設定値を指定します。

設定値	説明
Printer.Settings.PaperWidth58_0	用紙幅 58mm
Printer.Settings.PaperWidth60_0	用紙幅 60mm
Printer.Settings.PaperWidth80_0	用紙幅 80mm
Printer.Settings.PrintDensityDIP	印字濃度 DIP スイッチ設定
Printer.Settings.PrintDensity70	印字濃度 70%
Printer.Settings.PrintDensity75	印字濃度 75%
Printer.Settings.PrintDensity80	印字濃度 80%
Printer.Settings.PrintDensity85	印字濃度 85%

設定値	説明
Printer.Setting.PrintDensity90	印字濃度 90%
Printer.Setting.PrintDensity95	印字濃度 95%
Printer.Setting.PrintDensity100	印字濃度 100%
Printer.Setting.PrintDensity105	印字濃度 105%
Printer.Setting.PrintDensity110	印字濃度 110%
Printer.Setting.PrintDensity115	印字濃度 115%
Printer.Setting.PrintDensity120	印字濃度 120%
Printer.Setting.PrintDensity125	印字濃度 125%
Printer.Setting.PrintDensity130	印字濃度 130%
Printer.Setting.PrintSpeed1	印字速度 レベル 1
Printer.Setting.PrintSpeed2	印字速度 レベル 2
Printer.Setting.PrintSpeed3	印字速度 レベル 3
Printer.Setting.PrintSpeed4	印字速度 レベル 4
Printer.Setting.PrintSpeed5	印字速度 レベル 5
Printer.Setting.PrintSpeed6	印字速度 レベル 6
Printer.Setting.PrintSpeed7	印字速度 レベル 7
Printer.Setting.PrintSpeed8	印字速度 レベル 8
Printer.Setting.PrintSpeed9	印字速度 レベル 9
Printer.Setting.PrintSpeed10	印字速度 レベル 10
Printer.Setting.PrintSpeed11	印字速度 レベル 11
Printer.Setting.PrintSpeed12	印字速度 レベル 12
Printer.Setting.PrintSpeed13	印字速度 レベル 13
Printer.Setting.PrintSpeed14	印字速度 レベル 14

## listener

本 API の実行結果を返すリスナー登録メソッドを登録します。

- **(void) onSetPrinterSetting(int code);**

□ code

処理の実行結果が入ります。

設定値	説明
CODE_SUCCESS	成功
CODE_ERR_AUTORECOVER	自動復帰エラー発生
CODE_ERR_COVER_OPEN	カバーオープンエラー発生
CODE_ERR_CUTTER	オートカッターエラー発生
CODE_ERR_MECHANICAL	メカニカルエラー発生

設定値	説明
CODE_ERR_EMPTY	ロール紙エンド検出器に用紙なし
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_MEMORY	処理に必要なメモリーが確保できなかった。
CODE_ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
CODE_ERR_PROCESSING	処理が実行できなかった。
CODE_ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。
CODE_ERR_FAILURE	その他のエラーが発生した。

## 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが指定された。本 API を実行する条件が整っていなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- 本 API は他の API と一緒に使用できません。  
本 API 実行後は、コールバックが返るまで他の API を実行しないでください。
- 本 API 実行中はプリンターの電源を切らないでください。
- 本 API 実行中にプリンターとの通信が切断されると、プリンターが操作できなくなる場合があります。その場合は、プリンターの電源を入れ直してください。  
DHCP を使用しているネットワーク環境では、プリンターの再起動によって IP アドレスが変わることがあります。  
その場合は [Discovery クラス](#) でプリンターを検索して接続しなおしてください。  
USB 接続では、プリンターの再起動によって USB のデバイスノードが変わることがあります。  
その場合は [Discovery クラス](#) でプリンターを検索して接続しなおしてください。
- 本 API を実行するとプリンターは自動で再起動するため、通信が切断されます。必要に応じて接続しなおしてください。  
その場合は [Discovery クラス](#) でプリンターを検索して接続しなおしてください。
- 本 API は TM プリンターモデルとカスタマーディスプレイモデルのシステム構成で使用できます。  
TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデルのシステム構成では、Bluetooth® や USB での接続時に使用できます。
- 本 API は SSL/TLS 通信では使用できません。

## forceRecover

復帰可能エラー状態（例：オートカッターエラー）からの復帰コマンドを強制送信します。

復帰可能エラーの詳細は、各プリンターの詳細取扱説明書を参照してください。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、Printer クラスの [setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void forceRecover(int timeout) throws Epos2Exception
```

---

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
0 ~ 600000	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

---

### 補足説明

- 復帰可能エラーから復帰すると、プリンター本体のバッファーはリセットされます。
- ePOS-Device XML 制御時に使用できます。

## forcePulse

ドローアーキックコマンドを強制送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、Printer クラスの [setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void forcePulse(int drawer, int pulseTime, int timeout)
throws Epos2Exception
```

### パラメーター

#### drawer

ドローアーキックコネクターを指定します。

設定値	説明
Printer.DRAWER_2PIN	ドローアーキックコネクター 2 番ピン
Printer.DRAWER_5PIN	ドローアーキックコネクター 5 番ピン
Printer.PARAM_DEFAULT	既定値（ドローアーキックコネクター 2 番ピン）を選択

#### time

ドローアーキック信号のオン時間を指定します。

設定値	説明
Printer.PULSE_100	100 msec の信号
Printer.PULSE_200	200 msec の信号
Printer.PULSE_300	300 msec の信号
Printer.PULSE_400	400 msec の信号
Printer.PULSE_500	500 msec の信号
Printer.PARAM_DEFAULT	既定値（100 msec の信号）を選択

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
0 ~ 600000	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

---

### 補足説明

- ドロアーと外付けオプションブザーは同時に接続する事はできません。
- ePOS-Device XML 制御時に使用できます。

## forceStopSound

ブザー鳴動停止コマンドを強制送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、Printer クラスの [setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void forceStopSound(int timeout) throws  
Epos2Exception
```

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
0 ~ 600000	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

### 補足説明

- ドロワーと外付けオプションブザーは同時に接続する事はできません。
- 本 API はプリンターにブザーが付いてなければ使用できません。
- ePOS-Device XML 制御時に使用できます。

## forceCommand

ESC/POS コマンドを強制送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、Printer クラスの [setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void (byte[] data, int timeout) throws Epos2Exception
```

### パラメーター

#### data

ESC/POS コマンドを強制送信します。

バイナリーデータを指定します。

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
0 ~ 600000	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

### 補足説明

ePOS-Device XML 制御時に使用できます。

## forceReset

プリンタリセットコマンドを強制送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、Printer クラスの [setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

- `public void forceReset(int timeout) throws Epos2Exception`

---

### パラメーター

#### timeout

コマンド送信完了までのタイムアウト時間を msec 単位で指定します。

設定値	説明
0 ~ 600000	タイムアウト時間 (msec 単位)
Printer.PARAM_DEFAULT	既定値 (10000) を指定

---

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

---

### 補足説明

ePOS-Device XML 制御時に使用できます。

## setStatusChangeListener

ステータス変更イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setStatusChangeListener  
(StatusChangeListener statusChangeListener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface StatusChangeListener extends EventListener
```

### リスナー登録メソッド

```
void onPtrStatusChange(Printer printerObj, int eventType)
```

### パラメーター

#### printerObj

コールバック元のオブジェクトが入ります。

#### eventType

発生したイベントの種類が入ります。

イベントの種類	説明
Printer.EVENT_ONLINE	オンライン
Printer.EVENT_OFFLINE	オフライン
Printer.EVENT_POWER_OFF	電源オフ
Printer.EVENT_COVER_CLOSE	カバークローズ
Printer.EVENT_COVER_OPEN	カバーオープン
Printer.EVENT_PAPER_OK	用紙あり
Printer.EVENT_PAPER_NEAR_END	用紙ニアエンド
Printer.EVENT_PAPER_EMPTY	用紙なし
Printer.EVENT_DRAWER_HIGH	ドローアキックコネクター 3 番ピンの状態 = "H"
Printer.EVENT_DRAWER_LOW	ドローアキックコネクター 3 番ピンの状態 = "L"
Printer.EVENT_BATTERY_ENOUGH	バッテリー残量有り

イベントの種類	説明
Printer.EVENT_BATTERY_EMPTY	バッテリー残量無し

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setReceiveEventListener

印刷完了イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setReceiveEventListener (ReceiveListener  
receiveListener)
```

### パラメーター

#### target

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ReceiveListener extends EventListener
```

### リスナー登録メソッド

```
void onPtrReceive (Printer printerObj, int code,  
PrinterStatusInfo status, String printJobId)
```

### パラメーター

#### printerObj

コーレバック元のオブジェクトが入ります。

#### code

処理の実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	印刷成功
CODE_PRINTING	印刷中
CODE_ERR_AUTORECOVER	自動復帰エラー発生
CODE_ERR_COVER_OPEN	カバーオープンエラー発生
CODE_ERR_CUTTER	オートカッターエラー発生
CODE_ERR_MECHANICAL	メカニカルエラー発生
CODE_ERR_EMPTY	ロール紙エンド検出器に用紙なし
CODE_ERR_UNRECOVERABLE	復帰不可能エラー発生
CODE_ERR_FAILURE	要求ドキュメントの構文に誤りがある
CODE_ERR_NOT_FOUND	デバイス ID で指定したプリンターが存在しない

エラーステータス	説明
CODE_ERR_SYSTEM	印刷システムにエラーが発生
CODE_ERR_PORT	通信ポートに異常を検出
CODE_ERR_TIMEOUT	印刷タイムアウトが発生
CODE_ERR_JOB_NOT_FOUND	指定の印刷ジョブ ID が存在しない
CODE_ERR_SPOOLER	印刷キューに空き容量が無い
CODE_ERR_BATTERY_LOW	バッテリー残量が無い
CODE_ERR_TOO_MANY_REQUESTS	印刷データ送信数が許容量を超えた
CODE_ERR_REQUEST_ENTITY_TOO_LARGE	印刷ジョブのデータサイズが、プリンターの許容量を超えている

## status

処理実行時のステータスが `PrinterStatusInfo` 型で入ります。

## printJobId

- スプーラー機能が有効の場合、印刷ジョブ ID が入ります。
- スプーラー機能が無効の場合、`null` が入ります。

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に `listener` に `null` を指定した場合は、リスナー登録メソッドの登録を解除します。
- ePOS-Device XML 制御で通信異常が発生した場合、通信切断を通知しません。  
切断イベントを通知できるように、`setConnectionEventListener` を実装してください。
- 複数のプリンターを制御する場合は、呼び出し元の `Printer` オブジェクトと `printerObj` パラメーターを比較して、印刷が完了したプリンターを特定してください。

## interval

プリンターステータス (Printer.EVENT\_POWER\_OFF イベント) を更新、通知する間隔を指定します。

### 構文

#### Setter

```
public void setInterval (int interval) throws Epos2Exception
```

#### Getter

```
public int getInterval ()
```

### 設定値

プリンターステータス (Printer.EVENT\_POWER\_OFF イベント) を更新、通知する間隔を指定します。初期値は 3,000。

設定値	説明
1000 ~ 60000 の整数	プリンターステータスを更新する間隔 (msec 単位)
Printer.PARAM_DEFAULT	規定値 (3000) を指定

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。

### 補足説明

Printer.EVENT\_POWER\_OFF イベント以外のイベントは 1 秒未満の間隔で更新、通知されます。更新する間隔は変更できません。

# LineDisplay クラス

## LineDisplay

LineDisplay クラスを初期化します。

### 構文

```
public LineDisplay(int displaySeries, Context context) throws
Epos2Exception
```

### パラメーター

#### displaySeries

対象のカスタマーディスプレイの機種名を指定します。

設定値	説明
LineDisplay.DM_D110	DM-D110
LineDisplay.DM_D30	DM-D30

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが指定された。
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。

OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

カスタマーディスプレイとの通信を開始します。

### 構文

```
Public void connect (String target, int timeout) throws  
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

システム構成によって接続方法が異なります。

- カスタマーディスプレイモデル  
<接続形式>:<識別子>

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCP"	<ul style="list-style-type: none"> <li>• IPv4 形式の IP アドレス</li> <li>• MAC アドレス</li> <li>• プリンターホスト名</li> </ul>	"TCP:192.168.192.168"
Bluetooth®	"BT"	BD アドレス	"BT:00:22:15:7D:70:9C"
USB	"USB"	デバイスノード	"USB:/dev/udev/xxxxxxxxxx"

SSL/TLS 通信でプリンターと接続するには、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCPS"	<ul style="list-style-type: none"> <li>• IPv4 形式の IP アドレス</li> <li>• MAC アドレス</li> <li>• ホスト名</li> </ul>	"TCPS:192.168.192.168[local_display]"

- TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデル

Wi-Fi/Ethernet のネットワーク通信で接続する場合は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCP"	<ul style="list-style-type: none"> <li>• IPv4 形式の IP アドレス</li> <li>• MAC アドレス</li> <li>• ホスト名</li> </ul>	"TCP:192.168.192.168[local_display]"

Bluetooth® または USB で接続する場合は、以下の構文で指定してください。

I/F	接続形式	識別子	例
Bluetooth®	"BT"	BD アドレス	"BT:00:22:15:7D:70:9C"
USB	"USB"	デバイスノード	"USB:/dev/udev/xxxxxxxxxx"

- POS ターミナルモデル

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_display]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

## timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
LineDisplay.PARAM_DEFAULT	既定値 (15000) を指定

---

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。 プリンターがオフライン。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	デバイスが見つからない。
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

---

## 補足説明

- プリンター以外のデバイスは排他的にロックします。

- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- デバイス ID は、周辺機器を TM プリンター（親機）へ登録する際に設定する任意の文字列です。  
詳細は、各プリンターの詳細取扱説明書を参照してください。
- 本 API は TM プリンター（親機）と周辺機器の接続状態に関わらず成功します。

## disconnect

カスタマーディスプレイとの通信を終了します。

### 構文

```
Public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public DisplayStatusInfo getStatus()
```

### 戻り値

現在の状態が DisplayStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
connection : 接続状態	LineDisplay.TRUE	接続中
	LineDisplay.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。
- DM-D30 接続時に FALSE が返った場合は、TM プリンターとの接続が切れています。

## sendData

命令バッファーを送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、[setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

add 系 API (例 [addText](#)) でバッファリングしたデータを送信します。

---

### 構文

```
public void sendData() throw Epos2Exception
```

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理を実行できなかった。
ERR_ILLEGAL	制御コマンドがバッファリングされていない。 通信が開始されていない状態で、本 API が呼び出された。

## clearCommandBuffer

命令バッファーをクリアします。

バッファリングされた命令バッファーは、本 API を実行するまで保存されます。

### 構文

```
public void clearCommandBuffer()
```

## addInitialize

カスタマーディスプレイを初期化します。初期化により以下の状態となります。

- 表示文字が全て消去されて全てのウィンドウが破棄されます。
- カーソルはデフォルト状態に戻り、カスタマーディスプレイ原点に移動します。
- 表示点滅、表示輝度がデフォルト状態に戻ります。

---

### 構文

```
public void addInitialize ()
```

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addCreateWindow

ウィンドウ設定を命令バッファーに追加します。

ウィンドウは最大で 4 つまで設定可能です。

### 構文

```
public int addCreateWindow(int number, int x, int y, int width, int height, int scrollMode) throw Epos2Exception
```

### パラメーター

#### number

ウィンドウ番号を指定します。

設定値	説明
1 ~ 4 の整数	ウィンドウ番号

#### x

ウィンドウ原点のカスタマーディスプレイ上での x 座標を指定します。

設定値	説明
1 ~ 20 の整数	カスタマーディスプレイ上での x 座標

#### y

ウィンドウ原点のカスタマーディスプレイ上での y 座標を指定します。

設定値	説明
1 ~ 2 の整数	カスタマーディスプレイ上での y 座標

#### width

ウィンドウの幅を指定します。

設定値	説明
1 ~ 20 の整数	ウィンドウの幅

#### height

ウィンドウの高さを指定します。

設定値	説明
1 ~ 2 の整数	ウィンドウの高さ

#### scrollMode

ウィンドウのスクロールモードを指定します。

設定値	説明
LineDisplay.SCROLL_OVERWRITE	表示位置が上段右端位置にあるとき文字表示すると、表示位置を下段左端に移動し、下段右端にあるとき文字表示すると表示位置を上段左端に移動します。
LineDisplay.SCROLL_VERTICAL	表示位置が上段右端位置にあるとき文字表示すると、表示位置を下段左端に移動し、下段右端にあるとき文字表示するとすでに表示してある下段の表示文字を上段にスクロールし、下段の表示をクリアします。
LineDisplay.SCROLL_HORIZONTAL	表示位置が右端位置にあるとき文字表示すると、カーソルのある行にすでに表示している全ての表示文字は1文字分左へスクロールし、新たな表示文字を右端に表示します
LineDisplay.PARAM_DEFAULT	既定値(Overwrite)を選択

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

追加でウィンドウを設定する際は、設定済みのウィンドウとカスタマーディスプレイ上の設定範囲が重ならないように注意してください。

## addDestroyWindow

[addCreateWindow](#) で定義したウィンドウの破棄設定を命令バッファーに追加します。

### 構文

```
public void addDestroyWindow(int number) throw  
Epos2Exception
```

### パラメーター

#### number

破棄するウィンドウ番号を指定します。

設定値	説明
1 ~ 4 の整数	破棄するウィンドウ番号

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addSetCurrentWindow

カレントウィンドウの変更設定を命令バッファーに追加します。

[addCreateWindow](#) で定義したウィンドウを指定して、カレントウィンドウに設定します。

カーソルをカレントウィンドウの原点位置に移動させます。

### 構文

```
public void addSetCurrentWindow(int number) throw  
Epos2Exception
```

### パラメーター

#### number

移動先のウィンドウ番号を指定します。

設定値	説明
1 ~ 4 の整数	移動先のウィンドウ番号

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addClearCurrentWindow

カレントウィンドウ内の表示消去を命令バッファーに追加します。

### 構文

```
public void addClearCurrentWindow() throw Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addSetCursorPosition

カーソルの位置設定を命令バッファーに追加します。

### 構文

```
public void addSetCursorPosition(int x, int y) throw
Epos2Exception
```

### パラメーター

x

移動先となる x 座標を指定します。

設定値	説明
1 ~ 20 の整数	移動先となる x 座標

y

移動先となる y 座標を指定します。

設定値	説明
1 ~ 2 の整数	移動先となる y 座標

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

ウィンドウ内を移動したい場合は、[addMoveCursorPosition](#) を使用してください。

## addMoveCursorPosition

カレントウィンドウ内のカーソル位置設定を命令バッファーに追加します。

### 構文

```
public void addMoveCursorPosition(int position) throw
Epos2Exception
```

### パラメーター

#### position

カレントウィンドウ内のカーソルの移動先を指定します。

設定値	説明
LineDisplay.MOVE_TOP_LEFT	移動先を左上端に設定
LineDisplay.MOVE_TOP_RIGHT	移動先を右上端に設定
LineDisplay.MOVE_BOTTOM_LEFT	移動先を左下端に設定
LineDisplay.MOVE_BOTTOM_RIGHT	移動先を右下端に設定
LineDisplay.PARAM_DEFAULT	既定値（左上端）を選択

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addSetCursorType

カーソル表示設定を命令バッファーに追加します。

### 構文

```
public void addSetCursorType(int type) throw Epos2Exception
```

### パラメーター

#### type

カーソル表示方法を指定します。

設定値	説明
LineDisplay.CURSOR_NONE	カーソル表示なし
LineDisplay.CURSOR_UNDERLINE	アンダーライン
LineDisplay.PARAM_DEFAULT	既定値（カーソル表示なし）を選択

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addText

文字列の表示を命令バッファーに追加します。

### 構文

- `public void addText(String data) throws Epos2Exception`
- `public void addText(String data, int lang) throws Epos2Exception`
- `public void addText(String data, int x, int y) throws Epos2Exception`
- `public void addText(String data, int x, int y, int lang) throws Epos2Exception`

---

### パラメーター

#### `data`

表示文字列を指定します。

#### `lang`

文字列の表示言語を指定します。

設定値	説明
<code>LineDisplay.LANG_EN</code> (初期値)	英語
<code>LineDisplay.LANG_JA</code>	日本語
<code>LineDisplay.PARAM_DEFAULT</code>	既定値（英語）を選択

#### `x`

表示位置の `x` を指定します。

設定値	説明
1 ~ 20 の整数	表示位置の <code>x</code> 座標

#### `y`

表示位置の `y` を指定します。

設定値	説明
1 ~ 2 の整数	表示位置の <code>y</code> 座標

---

### 例外

処理に失敗した場合、以下のエラー値の `Epos2Exception` が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

## 補足説明

- パラメーター x と y は、カスタマーディスプレイの左上を原点とした座標を指定します。
- 表示位置の座標指定を省略すると、[addText](#) 実行時のカーソル位置座標が指定されます。

## addReverseText

白黒反転文字列の表示を命令バッファーに追加します。

### 構文

- `public void addReverseText(String data) throws Epos2Exception`
- `public void addReverseText(String data, int lang) throws Epos2Exception`
- `public void addReverseText(String data, int x, int y) throws Epos2Exception`
- `public void addReverseText(String data, int x, int y, int lang) throws Epos2Exception`

---

### パラメーター

#### `data`

表示文字列を指定します。

#### `lang`

文字列の表示言語を指定します。

設定値	説明
<code>LineDisplay.LANG_EN</code> (初期値)	英語
<code>LineDisplay.LANG_JA</code>	日本語
<code>LineDisplay.PARAM_DEFAULT</code>	既定値（英語）を選択

#### `x`

表示位置の `x` を指定します。

設定値	説明
1 ~ 20 の整数	表示位置の <code>x</code> 座標

#### `y`

表示位置の `y` を指定します。

設定値	説明
1 ~ 2 の整数	表示位置の <code>y</code> 座標

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- パラメーター x と y は、カスタマーディスプレイの左上を原点とした座標を指定します。
- 表示位置の座標指定を省略すると、[addText](#) 実行時のカーソル位置座標が指定されます。

## addMarqueeText

マーキー表示を命令バッファーに追加します。

マーキー表示は、表示ウィンドウのスクロールモード設定とは無関係に、1行の水平スクロールモードで表示されます。

### 構文

```
public void addMarqueeText(String data ,int format, int
unitWait, int repeatWait, int repeatCount, int lang) throws
Epos2Exception
```

### パラメーター

#### data

表示文字列を指定します。

#### format

文字列の表示方法を指定します。

設定値	説明
LineDisplay.MARQUEE_WALK	ウィンドウ右端から文字列を表示する
LineDisplay.MARQUEE_PLACE	ウィンドウ左端から文字列を表示する
LineDisplay.PARAM_DEFAULT	既定値 (Walk) を選択

#### unitWait

1 文字ごとの表示間隔 (msec 単位) を指定します。

設定値	説明
1 ~ 2000 の整数	1 文字ごとの表示間隔 (msec 単位)

#### repeatWait

文字列表示の繰り返し間隔 (msec 単位) を指定します。

設定値	説明
1 ~ 2000 の整数	文字列表示の繰り返し間隔 (msec 単位)

#### repeatCount

文字列表示の繰り返し回数を指定します。

設定値	説明
1 ~ 127 の整数	文字列表示の繰り返し回数
0	無制限に繰り返す

## lang

文字列の表示言語を指定します。

設定値	説明
LineDisplay.LANG_EN	英語
LineDisplay.LANG_JA	日本語
LineDisplay.PARAM_DEFAULT	既定値（英語）を選択

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## 補足説明

- DM-D30 の場合、repeatCount には 0 しか指定できません。
- DM-D30 を制御する場合、搭載されている LCD の特性により、文字表示切り替え間隔が短いとマーキー表示が乱れるおそれがあります。  
unitWait と repeatWait には、文字表示切り替え間隔が 600 msec 以上となる値の設定を推奨します。

## addSetBlink

画面の点滅設定を命令バッファーに追加します。

### 構文

```
public void addSetBlink(int interval) throws Epos2Exception
```

---

### パラメーター

#### interval

点滅間隔 (msec 単位) を指定します。設定値は 50 msec 単位で切り上げられます。

設定値	説明
1 ~ 12700 の整数	点滅間隔 (msec 単位)
0	点灯状態になる
LineDisplay.PARAM_DEFAULT	既定値 (点灯状態) を指定

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addSetBrightness

カスタマーディスプレイの輝度設定を命令バッファーに追加します。

### 構文

```
public void addSetBrightness(int brightness) throws
Epos2Exception
```

### パラメーター

#### brightness

カスタマーディスプレイの輝度値を指定します。

設定値	説明
LineDisplay.BRIGHTNESS_20	カスタマーディスプレイの輝度値を 20% に設定する
LineDisplay.BRIGHTNESS_40	カスタマーディスプレイの輝度値を 40% に設定する
LineDisplay.BRIGHTNESS_60	カスタマーディスプレイの輝度値を 60% に設定する
LineDisplay.BRIGHTNESS_100	カスタマーディスプレイの輝度値を 100% に設定する
LineDisplay.PARAM_DEFAULT	既定値（輝度値 100%）を選択

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## addShowClock

時刻の表示を命令バッファーに追加します。

時刻表示を行うと、表示中の文字列は全て削除されます。また、表示中の他の命令を送信すると、時刻表示を終了します。

### 構文

```
public void addShowClock() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

DM-D30 の場合、使用できません。

## addCommand

ESC/POS コマンドを命令バッファーに追加します。

### 構文

```
public void addCommand(byte[] data) throws Epos2Exception
```

---

### パラメーター

#### data

ESC/POS コマンドを指定します。  
バイナリーデータを指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

---

### 補足説明

- ESC/POS コマンドの詳細は以下の URL を参照してください。  
[https://reference.epson-biz.com/modules/ref\\_escpos\\_dm\\_d\\_ja/](https://reference.epson-biz.com/modules/ref_escpos_dm_d_ja/)
- Epson ePOS SDK は本 API で送信するコマンドをチェックしません。  
Epson ePOS SDK の動作を妨げるようなコマンドを送信した場合、他の API の動作やステータスの値が不正になることがあります。  
ESC/POS コマンドやカスタマーディスプレイの仕様を十分に理解した上で使用してください。

## setReceiveEventListener

表示完了イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setReceiveEventListener (ReceiveListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ReceiveListener extends EventListener
```

### リスナー登録メソッド

```
void onDispReceive (LineDisplay displayObj, int code)
```

### パラメーター

#### displayObj

コールバック元のオブジェクトが入ります。

#### code

処理の実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	表示が完了した
CODE_ERR_NOT_FOUND	デバイスが見つからない
CODE_ERR_INVALID_WINDOW	未登録のウィンドウが指定された
CODE_ERR_PORT	デバイスとの内部通信エラーが発生した
CODE_ERR_TIMEOUT	デバイスとの通信でタイムアウトエラーが発生した
CODE_ERR_TOO_MANY_REQUESTS	ディスプレイ表示データ送信数が許容量を超えた
CODE_ERR_FAILURE	その他のエラーが発生した

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

- ePOS-Device XML 制御で通信異常が発生した場合、通信切断を通知しません。  
切断イベントを通知できるように、[setConnectionEventListener](#) を実装してください。

# Keyboard クラス

## Keyboard

Keyboard クラスを初期化します。

### 構文

```
public Keyboard(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。

OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

キーボードとの通信を開始します。

### 構文

```
Public void connect(String target, int timeout) throws
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_keyboard]" "

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
Keyboard.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	デバイスが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、周辺機器を TM プリンター（親機）へ登録する際に設定する任意の文字列です。  
詳細は、各プリンターの詳細取扱説明書を参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- 本 API は TM プリンター（親機）と周辺機器の接続状態に関わらず成功します。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

キー ボードとの通信を終了します。

### 構文

```
Public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public KeyboardStatusInfo getStatus ()
```

### 戻り値

現在の状態が KeyboardStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
connection : 接続状態	Keyboard.TRUE	接続中
	Keyboard.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## setPrefix

文字列入力の接頭辞の一覧を設定します。

キーボード入力を文字列としてまとめて受け取る際、文字列の始まりと判断するためのキーコードを設定します。

### 構文

```
public void setPrefix(int[] prefixes) throws Epos2Exception
```

### パラメーター

#### prefixes

キーコードの配列を指定します。

キーコードは [Key Code 一覧](#) を参照します。

null を指定した場合、setPrefix を無効にします。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

NULL または "" (空文字) を入力した場合、設定が解除されます。

## getPrefix

現在設定中の接頭辞の一覧を取得します。

### 構文

```
public int[] getPrefix()
```

### 戻り値

setPrefix で設定中の接頭辞の一覧が配列で返ります。

## setKeyPressEventListener

キー押下イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setKeyPressEventListener (KeyPressListener  
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface KeyPressListener extends EventListener
```

### リスナー登録メソッド

```
void onKbdKeyPress (Keyboard keyboardObj, int keyCode,  
String ascii)
```

### パラメーター

#### keyboardObj

コールバック元のオブジェクトが入ります。

#### keyCode

押下されたキーコードが入ります。

#### ascii

押下されたキーコードに対応する文字が入ります。キーコードに対応する文字がない場合、"" (空文字) が入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setReadStringEventListener

`setPrefix` で指定した文字列群のいずれかで始まり、Enter キーを検出するまでの間に検出した文字列入力を文字列として受け取るリスナー登録メソッドを登録します。

### 構文

```
public void setReadStringEventListener (ReadStringListener  
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ReadStringListener extends EventListener
```

### リスナー登録メソッド

```
void onKbdReadString (Keyboard keyboardObj, String  
readString, int prefix)
```

### パラメーター

#### keyboardObj

コーレバック元のオブジェクトが入ります。

#### readString

検出した文字列が入ります。

#### prefix

文字列の開始と判断したキーコードが入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に `listener` に `null` を指定した場合は、リスナー登録メソッドの登録を解除します。

# POSKeyboard クラス

## PosKeyboard

PosKeyboard クラスを初期化します。

### 構文

```
public PosKeyboard(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。

OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

POS キーボードとの通信を開始します。

### 構文

```
public void connect(String target, int timeout) throws
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_poskeyboa rd]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
Poskeyboard.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとしました。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	プリンターが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、POS キーボードを TM プリンター（親機）へ登録する際に設定する論理デバイス名です。  
詳細は、TM-DT シリーズ周辺機器制御ガイドを参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

POS キーボードとの通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getstatus

現在のステータス情報を取得します。

### 構文

```
public PosKeyboardStatusInfo getStauts()
```

### 戻り値

現在の状態が PosKeyboardStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
(int)connection : 接続状態	PosKeyboard.TRUE	接続中
	PosKeyboard.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## setKeyPressEventListener

キー押下イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setKeyPressEventListener(KeyPressListener  
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface KeyPressListener extends EventListener
```

### リスナー登録メソッド

```
void onPosKbdKeyPress(PosKeyboard keyboardObj, int  
posKeyCode)
```

### パラメーター

#### keyboardObj

コールバック元のオブジェクトが入ります。

#### posKeyCode

押下されたキーコードが入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します

# BarcodeScanner クラス

## BarcodeScanner

BarcodeScanner クラスを初期化します。

### 構文

```
public BarcodeScanner(Context context) throws  
Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。  
OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

バーコードスキャナーとの通信を開始します。

### 構文

```
Public void connect (String target, int timeout) throws  
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

システム構成によって接続方法が異なります。

- TM-m30 または TM-T88VI+DM-D30+ バーコードスキャナーモデル /  
TM-m30 または TM-T88VI+ バーコードスキャナーモデル

Wi-Fi/Ethernet のネットワーク通信で接続する場合は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	"TCP"	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>ホスト名</li> </ul>	"TCP:192.168.192.168[local_scanner]"

Bluetooth® または USB で接続する場合は、以下の構文で指定してください。

<接続形式>:<識別子>

I/F	接続形式	識別子	例
Bluetooth®	"BT"	BD アドレス	"BT:00:22:15:7D:70:9C"
USB	"USB"	デバイスノード	"USB:/dev/udev/xxxxxxxxxx"

- POS ターミナルモデル

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	<ul style="list-style-type: none"> <li>"TCP"</li> <li>"TCPS"</li> </ul>	<ul style="list-style-type: none"> <li>IPv4 形式の IP アドレス</li> <li>MAC アドレス</li> <li>ホスト名</li> </ul>	"TCP:192.168.192.168[local_scanner]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

## timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
BarcodeScanner.PARAM_DEFAULT	既定値 (15000) を指定

## 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	デバイスが見つからない。
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、周辺機器を TM プリンター（親機）へ登録する際に設定する任意の文字列です。  
詳細は、各プリンターの詳細取扱説明書を参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- 本 API は TM プリンター（親機）と周辺機器の接続状態に関わらず成功します。
- バーコードリーダーのサフィックス（区切り文字）の設定は CR（改行コード）にしてください。これ以外の設定ではデータを取得できません。
- プリンター以外のデバイスは排他的にロックします。スキャナー、およびスキャナーに接続されたプリンターを、同時に複数のアプリケーションから制御しないでください。
- TM-m30またはTM-T88VI+バーコードスキャナーの組み合わせをBluetooth®またはUSBで接続して使用する場合、印刷中に読み取ったバーコードデータは印刷処理終了後に通知されます。

## disconnect

バーコードスキャナーとの通信を終了します。

### 構文

```
Public void disconnect () throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public ScannerStatusInfo getStatus ()
```

### 戻り値

現在の状態が ScannerStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
connection : 接続状態	BarcodeScanner.TRUE	接続中
	BarcodeScanner.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。
- TM-m30またはTM-T88VI+バーコードスキャナーの組み合わせをBluetooth®またはUSBで接続して使用する場合は、TM-m30 または TM-T88VI とバーコードスキャナーの接続状態を取得します。これ以外の接続方法で使用する場合は、スマートデバイスと TM-m30 または TM-T88VI の接続状態を取得します。

## setScanEventListener

バーコードデータ入力イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setScanEventListener (ScanListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ScanListener extends EventListener
```

### リスナー登録メソッド

```
void onScanData (BarcodeScanner scannerObj, String scanData)
```

### パラメーター

#### scannerObj

コールバック元のオブジェクトが入ります。

#### scanData

検出した文字列が入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- バーコードデータ入力イベントの通知タイミングは接続方法によって異なります。  
詳細は [connect](#) API の target パラメーターの記述を参照してください。

# MSR クラス

## Msr

Msr クラスを初期化します。

### 構文

```
public Msr(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。  
OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

MSR との通信を開始します。

### 構文

```
public void connect(String target, int timeout) throws
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_ms]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
Msr.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	プリンターが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、MSR を TM プリンター（親機）へ登録する際に設定します。  
制御方法によって設定値が異なります。

制御方法	デバイス ID	参照
デバイス制御プログラム	論理デバイス名	TM-DT シリーズ周辺機器制御ガイド
デバイス制御スクリプト	任意の文字列	各プリンターの詳細取扱説明書

- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

MSRとの通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getstatus

現在のステータス情報を取得します。

### 構文

```
public MsrStatusInfo getStauts()
```

### 戻り値

現在の状態が MsrStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
(int)connection : 接続状態	Msr.TRUE	接続中
	Msr.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## setDataEventListener

カードデータ検出イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setDataEventListener(DataListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface DataListener extends EventListener
```

### リスナー登録メソッド

```
void onData(Msr msrObj, Data data)
```

### パラメーター

#### msrObj

コールバック元のオブジェクトが入ります。

#### data

読み込んだカードデータを取得します。

Data 構造体に結果が入ります。

プロパティー	説明
String track1	トラック 1 のデータ全体
String track2	トラック 2 のデータ全体
String track4	トラック 4(JIS2 トラック) のデータ全体
String accountNumber	アカウント番号
String expirationData	有効期限(“YYMM” 形式)
String surname	姓
String firstName	名
String middleInitial	ミドルネーム / イニシャル
String title	タイトル
String serviceCode	サービスコード
String track1_dd	トラック 1 の任意データ

プロパティー	説明
String track2_dd	トラック 2 の任意データ

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に `listener` に `null` を指定した場合は、リスナー登録メソッドの登録を解除します。

# SimpleSerial クラス

## SimpleSerial

SimpleSerial クラスを初期化します。

### 構文

```
public SimpleSerial(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。  
OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

シリアル通信デバイスとの通信を開始します。

### 構文

```
Public void connect(String target, int timeout) throws
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_serial]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
SimpleSerial.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	デバイスが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、周辺機器を TM プリンター（親機）へ登録する際に設定する任意の文字列です。  
詳細は、各プリンターの詳細取扱説明書を参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- 本 API は TM プリンター（親機）と周辺機器の接続状態に関わらず成功します。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

シリアル通信デバイスとの通信を終了します。

### 構文

```
Public void disconnect () throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public SerialStatusInfo getStatus ()
```

### 戻り値

現在の状態が SerialStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
connection : 接続状態	SimpleSerial.TRUE	接続中
	SimpleSerial.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## sendCommand

任意コマンドを送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、[setReceiveEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void sendCommand(byte[] data) throws  
Epos2Exception
```

---

### パラメーター

#### data

コマンドを指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

## setReceiveEventListener

デバイスからの受信イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setReceiveEventListener(ReceiveListener listener);
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ReceiveListener extends EventListener
```

### リスナー登録メソッド

```
void onSimpleSerialReceive (SimpleSerial serialObj, byte[] data);
```

### パラメーター

#### serialObj

コールバック元のオブジェクトが入ります。

#### data

受信データを受け取ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- ePOS-Device XML 制御で通信異常が発生した場合、通信切断を通知しません。  
切断イベントを通知できるように、[setConnectionEventListener](#) を実装してください。

# CashChanger クラス (デバイス制御プログラム)

## CashChanger

CashChanger クラスを初期化します。

### 構文

```
public CashChanger(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。  
OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

自動つり銭機との通信を開始します。

### 構文

```
public void connect(String target, int timeout) throws
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_changer]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間 (msec 単位) を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間 (msec 単位)
Cashchanger.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	プリンターが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、自動つり銭機を TM プリンター（親機）へ登録する際に設定する論理デバイス名です。  
詳細は、TM-D シリーズ周辺機器制御ガイドを参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

自動つり銭機との通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public CashChangerStatusInfo getStatus()
```

### 戻り値

現在の状態が CashChangerStatusInf 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

状態	ステータス	説明
(int)connection : 接続状態	CashChanger.TRUE	接続中
	CashChanger.FALSE	切断中

### 補足説明

ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。

## getOposErrorCode

デバイスベンダー定義のエラーが発生した際に、そのエラーコードを取得します。

### 構文

```
public int getOposErrorCode()
```

### 戻り値

発生したデバイスベンダー定義のエラーコードが返ります。

### 補足説明

デバイスベンダー定義のエラーが発生していない場合、0が返ります。

## readCashCount

デバイス内の金種、枚数情報を取得します。

本 API の結果は、[setCashCountEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void readCashCount() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## beginDeposit

入金処理を開始します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void beginDeposit() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## pauseDeposit

入金処理を一時停止します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void pauseDeposit() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## restartDeposit

入金処理を再開します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void restartDeposit() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## endDeposit

入金処理を終了します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void endDeposit(int config) throws Epos2Exception
```

### パラメーター

#### config

終了動作を指定します。

設定値	説明
CashChanger.DEPOSIT_CHANGE	つり銭あり ( 収納部への収納を保留する。 )
CashChanger.DEPOSIT_NOCHANGE	つり銭なし ( 収納部に収納する。 )
CashChanger.DEPOSIT_REPAY	預かり金返却

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## dispenseChange

指定金額で出金します。

本 API の実行結果は、[setDispenseEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void dispenseChange(int cash) throws Epos2Exception
```

### パラメーター

#### cash

出金する金額を指定します。

設定値	説明
0 ~ 99999 の整数	出金する金額を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## dispenseCash

紙幣と硬貨の枚数を指定して出金します。

本 API の実行結果は、[setDispenseEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void dispenseCash(Map<String, Integer> data) throws
Epos2Exception
```

---

### パラメーター

#### data

出金する金種の枚数を指定します。キーに金種の文字列で指定し、枚数を値に指定します。

キー	説明	値
(String)"jpy1"	1 円硬貨	(Integer)
(String)"jpy5"	5 円硬貨	(Integer)
(String)"jpy10"	10 円硬貨	(Integer)
(String)"jpy50"	50 円硬貨	(Integer)
(String)"jpy100"	100 円硬貨	(Integer)
(String)"jpy500"	500 円硬貨	(Integer)
(String)"jpy1000"	1,000 円紙幣	(Integer)
(String)"jpy2000"	2,000 円紙幣	(Integer)
(String)"jpy5000"	5,000 円紙幣	(Integer)
(String)"jpy10000"	10,000 円紙幣	(Integer)

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

---

### 補足説明

data の値は、0 ~ 99 の範囲で指定します。

## sendDirectICommand

OPOS の DirectIO メソッドを送信します。

本 API の実行結果は、[setDirectICommandReplyEventListerner](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void sendDirectICommand(int command, int data,
String string) throws Epos2Exception
```

---

### パラメーター

#### command

DirectIO のコマンド番号を指定します。

#### data

DirectIO の数値データを指定します。

#### string

DirectIO の文字列データを指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

## setCashCountEventListener

[readCashCount](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setCashCountEventListener(CashCountListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface CashCountListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerCashCount(CashChanger cchangerObj, int
code, Map<String, Integer> data)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_DISCREPANCY	金額不一致の可能性あり
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

#### data

自動つり銭機内の金種、枚数情報が入ります。キーに金種の文字列、値に枚数が入ります。

キー	説明	値
(String)"jpy1"	1 円硬貨	(Integer)

キー	説明	値
(String)"jpy5"	5 円硬貨	(Integer)
(String)"jpy10"	10 円硬貨	(Integer)
(String)"jpy50"	50 円硬貨	(Integer)
(String)"jpy100"	100 円硬貨	(Integer)
(String)"jpy500"	500 円硬貨	(Integer)
(String)"jpy1000"	1,000 円紙幣	(Integer)
(String)"jpy2000"	2,000 円紙幣	(Integer)
(String)"jpy5000"	5,000 円紙幣	(Integer)
(String)"jpy10000"	10,000 円紙幣	(Integer)

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- data パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setDepositEventListener

`beginDeposit`、`pauseDeposit`、`restartDeposit`、`endDeposit` の実行結果を通知するイベントのリスナー登録メソッドを登録します。

`beginDeposit` と `restartDeposit` 実行後は、自動つり銭機へ入金される度に `setDepositEventListener` が入金金額を通知します。(使用する OPOS ドライバーが対応している場合のみ)

---

### 構文

```
public void setDepositEventListener(DepositListener listener)
```

---

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

---

### リスナーインターフェイス

```
public interface DepositListener extends EventListener
```

---

### リスナー登録メソッド

```
void onCChangerDeposit(CashChanger cchangerObj, int code,
int status, int amount, Map<String, Integer> data)
```

---

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cashchanger.Epos2CallbackCode.CODE_ERR_OPOSCODE	デバイスベンダー定義のエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

## status

処理実行時のステータスが入ります。

エラーステータス	説明
CashChanger.STATUS_BUSY	他コマンド要求による動作中
CashChanger.STATUS_PAUSE	入金一時停止状態
CashChanger.STATUS_END	入金停止状態
CashChanger.STATUS_ERR	エラー状態

## amount

入金金額が入ります。

## data

自動つり銭機内の金種、枚数情報が入ります。キーに金種の文字列、値に枚数が入ります。

キー	説明	値
(String)"jpy1"	1 円硬貨	(Integer)
(String)"jpy5"	5 円硬貨	(Integer)
(String)"jpy10"	10 円硬貨	(Integer)
(String)"jpy50"	50 円硬貨	(Integer)
(String)"jpy100"	100 円硬貨	(Integer)
(String)"jpy500"	500 円硬貨	(Integer)
(String)"jpy1000"	1,000 円紙幣	(Integer)
(String)"jpy2000"	2,000 円紙幣	(Integer)
(String)"jpy5000"	5,000 円紙幣	(Integer)
(String)"jpy10000"	10,000 円紙幣	(Integer)

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- [beginDeposit](#) と [restartDeposit](#) が成功した場合、status に CashChanger.STATUS\_BUSY が返ります。
- [pauseDeposit](#) が成功した場合、status に CashChanger.STATUS\_PAUSE が返ります。
- [endDeposit](#) メソッドが成功した場合、status に CashChanger.STATUS\_END が返ります。
- data パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setDispenseEventListener

[dispenseChange](#)、[dispenseCash](#)の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setDispenseEventListener(DispenseListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface DispenseListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerDispense(CashChanger cchangerObj, int
code)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cashchanger.Epos2CallbackCode.CODE_ERR_PARAM	パラメーター不正エラー
cashchanger.Epos2CallbackCode.CODE_ERR_OPOSCODE	デバイスベンダー定義のエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

---

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setDirectIOCommandReplyEventListener

[sendDirectIOCommand](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setDirectIOCommandReplyEventListener(DirectIOCommand
ReplyListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface DirectIOCommandReplyListener extends
EventLister
```

### リスナー登録メソッド

```
void onCChangerDirectIOCommandReply(CashChanger
cchangerObj, int code, int command, int data, String string)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_OPOSCODE	デバイスベンダー定義のエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

#### command

DirectIO のコマンド番号が入ります。

**data**

DirectIO の数値データが入ります。

**string**

DirectIO の文字列データが入ります。

---

**補足説明**

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setDirectIOEventListener

OPOS ドライバーの DirectIOEvent を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setDirectIOEventListener(DirectIOListener  
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface DirectIOListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerDirectIO(CashChanger cchangerObj, int  
eventNumber, int data, String string)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### eventNumber

DirectIOEvent のコマンド番号が入ります。

#### data

DirectIOEvent の数値データが入ります。

#### string

DirectIOEvent の文字列データが入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setStatusUpdateEventListener

OPOS ドライバーの StatusUpdateEvent を通知するイベントのリスナー登録メソッドを登録します。 StatusUpdateEvent は、自動つり銭機の収納庫ステータスや機器の状態が変化した際に発生します。 詳細は、OPOS ドライバーのマニュアルを参照してください。

---

### 構文

```
public void
setStatusUpdateEventListener(StatusUpdateListener listener)
```

---

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

---

### リスナーインターフェイス

```
public interface StatusUpdateListener extends EventListener
```

---

### リスナー登録メソッド

```
void onCChangerStatusUpdate(CashChanger cchangerObj, int
status)
```

---

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### status

StatusUpdateEvent の値が入ります。

ステータス	説明
CashChanger.SUE_POWER_ONLINE	オンライン状態
CashChanger.SUE_POWER_OFF	電源が切れているか、プリンターと接続されていない
CashChanger.SUE_POWER_OFFLINE	オフライン状態
CashChanger.SUE_POWER_OFF_OFFLINE	<ul style="list-style-type: none"> <li>電源が切れているか、プリンターと接続されていない</li> <li>オフライン状態</li> </ul>
CashChanger.SUE_STATUS_EMPTY	空状態の収納庫あり
CashChanger.SUE_STATUS_NEAREMPTY	空に近い状態の収納庫あり

ステータス	説明
CashChanger.SUE_STATUS_EMPTYOK	現金が補充され、SUE_STATUS_EMPTY、SUE_STATUS_NEAREMPTY ステータスが解除された
CashChanger.SUE_STATUS_FULL	満杯状態の収納庫あり
CashChanger.SUE_STATUS_NEARFULL	満杯に近い状態の収納庫あり
CashChanger.SUE_STATUS_FULLOK	現金が回収され、SUE_STATUS_FULL、SUE_STATUS_NEARFULL ステータスが解除された
CashChanger.SUE_STATUS_JAM	機器障害が発生
CashChanger.SUE_STATUS_JAMOK	機器障害から復帰

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

# CashChanger クラス（デバイス制御スクリプト）

## CashChanger

CashChanger クラスを初期化します。

### 構文

```
public CashChanger(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。  
OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

自動つり銭機との通信を開始します。

### 構文

```
public void connect(String target, int timeout) throws
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_changer]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間 (msec 単位) を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間 (msec 単位)
Cashchanger.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	プリンターが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、自動つり銭機を TM プリンター（親機）へ登録する際に設定する論理デバイス名です。  
詳細は、TM-DT シリーズ周辺機器制御ガイドを参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

自動つり銭機との通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public CashChangerStatusInfo getStatus()
```

### 戻り値

現在の状態が CashChangerStatusInf 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

状態	ステータス	説明
(int)connection : 接続状態	CashChanger.TRUE	接続中
	CashChanger.FALSE	切断中

### 補足説明

ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。

## setConfigCountMode

自動つり銭機の計数モードを変更します。

本 API の実行結果は、[setConfigChangeEventLister](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void setConfigCountMode(int countMode) throws
Epos2Exception
```

### パラメーター

#### countMode

計数モードを指定します。

設定値	説明
CashChanger.COUNT_MODE_MANUAL_INPUT	手入力モード
CashChanger.COUNT_MODE_AUTO_COUNT	自動計数モード

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## setConfigLeftCash

自動つり銭機の残置金額を変更します。

本 API の実行結果は、[setConfigChangeEventLister](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void setConfigLeftCash(int coins, int bills) throws  
Epos2Exception
```

### パラメーター

#### coins

硬貨の残置金額を指定します。

設定値	説明
0 ~ 99999 の整数	10 円単位で指定。

#### bills

紙幣の残置金額を指定します。

設定値	説明
0 ~ 999 の整数	1,000 円単位で指定。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## readCashCount

デバイス内の金種、枚数情報を取得します。

本 API の結果は、[setCashCountEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void readCashCount() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## beginDeposit

入金処理を開始します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void beginDeposit() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## pauseDeposit

入金処理を一時停止します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void pauseDeposit() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## restartDeposit

入金処理を再開します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void restartDeposit() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## endDeposit

入金処理を終了します。

本 API の実行結果は、[setDepositEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void endDeposit(int config) throws Epos2Exception
```

---

### パラメーター

#### config

終了動作を指定します。

設定値	説明
CashChanger.DEPOSIT_CHANGE	つり銭あり（収納部への収納を保留する。）
CashChanger.DEPOSIT_NOCHANGE	つり銭なし（収納部に収納する。）
CashChanger.DEPOSIT_REPAY	預かり金返却

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## dispenseChange

指定金額で出金します。

本 API の実行結果は、[setDispenseEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void dispenseChange(int cash) throws Epos2Exception
```

### パラメーター

#### cash

出金する金額を指定します。

設定値	説明
0 ~ 99999 の整数	出金する金額を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## dispenseCash

紙幣と硬貨の枚数を指定して出金します。

本 API の実行結果は、[setDispenseEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void dispenseCash(Map<String, Integer> data) throws
Epos2Exception
```

---

### パラメーター

#### data

出金する金種の枚数を指定します。キーに金種の文字列で指定し、枚数を値に指定します。

キー	説明	値
(String)"jpy1"	1 円硬貨	(Integer)
(String)"jpy5"	5 円硬貨	(Integer)
(String)"jpy10"	10 円硬貨	(Integer)
(String)"jpy50"	50 円硬貨	(Integer)
(String)"jpy100"	100 円硬貨	(Integer)
(String)"jpy500"	500 円硬貨	(Integer)
(String)"jpy1000"	1,000 円紙幣	(Integer)
(String)"jpy2000"	2,000 円紙幣	(Integer)
(String)"jpy5000"	5,000 円紙幣	(Integer)
(String)"jpy10000"	10,000 円紙幣	(Integer)

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

---

### 補足説明

data の値は、0 ~ 99 の範囲で指定します。

## collectCash

デバイス内の現金を回収します。

本 API の実行結果は、[setCollectEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void collectCash(int type) throws Epos2Exception
```

### パラメーター

#### type

回収方法を指定します。

設定値	説明
CashChanger.COLLECT_ALL_CASH	全額回収
CashChanger.COLLECT_PART_OF_CASH	残置回収

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## openDrawer

キャッシュドロワーを開きます。

本 API の実行結果は、[setStatusChangeEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void openDrawer() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった

## sendCommand

任意コマンドを送信します。

本 API の実行結果は、[setCommandReplyEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void sendCommand(byte[] data) throws
Epos2Exception
```

---

### パラメーター

#### data

コマンドを指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

## setCashCountEventListener

[readCashCount](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setCashCountEventListener(CashCountListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface CashCountListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerCashCount(CashChanger cchangerObj, int
code, Map<String, Integer> data)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_DISCREPANCY	金額不一致の可能性あり
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

#### data

自動つり銭機内の金種、枚数情報が入ります。キーに金種の文字列、値に枚数が入ります。

キー	説明	値
(String)"jpy1"	1 円硬貨	(Integer)

キー	説明	値
(String)"jpy5"	5 円硬貨	(Integer)
(String)"jpy10"	10 円硬貨	(Integer)
(String)"jpy50"	50 円硬貨	(Integer)
(String)"jpy100"	100 円硬貨	(Integer)
(String)"jpy500"	500 円硬貨	(Integer)
(String)"jpy1000"	1,000 円紙幣	(Integer)
(String)"jpy2000"	2,000 円紙幣	(Integer)
(String)"jpy5000"	5,000 円紙幣	(Integer)
(String)"jpy10000"	10,000 円紙幣	(Integer)
(String)"jpy1000cassete"	回収部の 1,000 円紙幣	(Integer)
(String)"jpy2000cassete"	回収部の 2,000 円紙幣	(Integer)
(String)"jpy5000cassete"	回収部の 5,000 円紙幣	(Integer)
(String)"jpy10000cassete"	回収部の 10,000 円紙幣	(Integer)

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- data パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setDepositEventListener

`beginDeposit`、`pauseDeposit`、`restartDeposit`、`endDeposit` の実行結果を通知するイベントのリスナー登録メソッドを登録します。

`beginDeposit` と `restartDeposit` 実行後は、自動つり銭機へ入金される度に `setDepositEventListener` が入金金額を通知します。

---

### 構文

```
public void setDepositEventListener(DepositListener listener)
```

---

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

---

### リスナーインターフェイス

```
public interface DepositListener extends EventListener
```

---

### リスナー登録メソッド

```
void onCChangerDeposit(CashChanger cchangerObj, int code,
int status, int amount, Map<String, Integer> data)
```

---

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cashchanger.Epos2CallbackCode.CODE_ERR_OPOSCODE	デバイスベンダー定義のエラー
cashchanger.Epos2CallbackCode.CODE_ERR_CASH_IN_TRAY	現金抜き取り待ち状態
cashchanger.Epos2CallbackCode.CODE_ERR_REJECT_UNIT	リジェクトユニット容量オーバーエラー

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

### status

処理実行時のステータスが入ります。

エラーステータス	説明
CashChanger.STATUS_BUSY	他コマンド要求による動作中
CashChanger.STATUS_PAUSE	入金一時停止状態
CashChanger.STATUS_END	入金停止状態
CashChanger.STATUS_ERR	エラー状態

### amount

入金金額が入ります。

### data

自動つり銭機内の金種、枚数情報が入ります。キーに金種の文字列、値に枚数が入ります。

キー	説明	値
(String)"jpy1"	1 円硬貨	(Integer)
(String)"jpy5"	5 円硬貨	(Integer)
(String)"jpy10"	10 円硬貨	(Integer)
(String)"jpy50"	50 円硬貨	(Integer)
(String)"jpy100"	100 円硬貨	(Integer)
(String)"jpy500"	500 円硬貨	(Integer)
(String)"jpy1000"	1,000 円紙幣	(Integer)
(String)"jpy2000"	2,000 円紙幣	(Integer)
(String)"jpy5000"	5,000 円紙幣	(Integer)
(String)"jpy10000"	10,000 円紙幣	(Integer)

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- [beginDeposit](#) と [restartDeposit](#) が成功した場合、status に CashChanger.STATUS\_BUSY が返ります。
- [pauseDeposit](#) が成功した場合、status に CashChanger.STATUS\_PAUSE が返ります。
- [endDeposit](#) メソッドが成功した場合、status に CashChanger.STATUS\_END が返ります。
- data パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setDispenseEventListener

[dispenseChange](#)、[dispenseCash](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setDispenseEventListener(DispenseListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface DispenseListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerDispense(CashChanger cchangerObj, int
code)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_SHORTAGE	現金不足による失敗
cashchanger.Epos2CallbackCode.CODE_ERR_CASH_IN_TRAY	現金抜き取り待ち状態
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cashchanger.Epos2CallbackCode.CODE_ERR_PARAM	パラメーター不正エラー
cashchanger.Epos2CallbackCode.CODE_ERR_OPOSCODE	デバイスベンダー定義のエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

---

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setConfigChangeEventLister

`setConfigCountMode`、`setConfigLeftCash` の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setConfigChangeEventLister(ConfigChangeListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ConfigChangeListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerConfigChange(CashChanger cchangerObj,
int code)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_CASH_IN_TRAY	現金抜き取り待ち状態
cashchanger.Epos2CallbackCode.CODE_ERR_UNSUPPORTED	デバイス制御プログラムで制御する自動つり銭機に処理を実行した。
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_ERR_PARAM	パラメーター不正エラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setCollectEventListener

[collectCash](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setCollectEventListener(CollectListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface CollectListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerCollect(CashChanger cchangerObj, int code)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_SHORTAGE	出金に成功したが、収納庫内の現金が空、もしくは空に近い状態
cashchanger.Epos2CallbackCode.CODE_ERR_CASH_IN_TRAY	現金抜き取り待ち状態
cashchanger.Epos2CallbackCode.CODE_ERR_UNSUPPRESSED	デバイス制御プログラムで制御する自動つり銭機に処理を実行した。
cashchanger.Epos2CallbackCode.CODE_ERR_COMMAND	送信コマンドエラー
cashchanger.Epos2CallbackCode.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cashchanger.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

---

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setCommandReplyEventListener

[sendCommand](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setCommandReplyEventListener(CommandReplyListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface CommandReplyListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerCommandReply(CashChanger cchangerObj,
int code, byte[] data)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_BUSY	動作中のため処理できない。
cashchanger.Epos2CallbackCode.CODE_ERR_PARAM	パラメーター不正エラー
cashchanger.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

#### data

受信コマンドが入ります。

### 補足説明

□ 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。

- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

## setStatusChangeListener

収納庫ステータスの変化を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setStatusChangeListener(StatusChangeListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface StatusChangeListener extends EventListener
```

### リスナー登録メソッド

```
void onCChangerStatusChange(CashChanger cchangerObj, int
code, Map<String, Integer> status)
```

### パラメーター

#### cchangerObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cashchanger.Epos2CallbackCode.CODE_SUCCESS	成功
cashchanger.Epos2CallbackCode.CODE_DISCREPANCY	金額不一致の可能性あり

#### status

収納庫ステータスが入ります。キーに金種の文字列、値にステータスが入ります。

キー	説明	値
(String)"st1"	1 円硬貨の状態	(Integer)
(String)"st5"	5 円硬貨の状態	(Integer)
(String)"st10"	10 円硬貨の状態	(Integer)
(String)"st50"	50 円硬貨の状態	(Integer)

キー	説明	値
(String)"st100"	100 円硬貨の状態	(Integer)
(String)"st500"	500 円硬貨の状態	(Integer)
(String)"st1000"	1,000 円紙幣の状態	(Integer)
(String)"st2000"	2,000 円紙幣の状態	(Integer)
(String)"st5000"	5,000 円紙幣の状態	(Integer)
(String)"st10000"	10,000 円紙幣の状態	(Integer)

各ステータスには以下の値が入ります。

エラーステータス	説明
ST_EMPTY	空
ST_NEAR_EMPTY	空に近い状態
ST_OK	適量
ST_NEAR_FULL	満杯に近い状態
ST_FULL	満杯

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- API 実施時、listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

# CAT クラス

## Cat

Cat クラスを初期化します。

### 構文

```
public Cat(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

## connect

信用照会端末との通信を開始します。

### 構文

```
public void connect(String target, int timeout) throws  
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_cat]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
Cat.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	プリンターが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、信用照会端末を TM プリンター（親機）へ登録する際に設定する論理デバイス名です。  
詳細は、TM-DT シリーズ周辺機器制御ガイドを参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

信用照会端末との通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getstatus

現在のステータス情報を取得します。

### 構文

```
public CatStatusInfo getStauts()
```

### 戻り値

現在の状態が CatStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
(int)connection : 接続状態	Cat.TRUE	接続中
	Cat.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## getOposErrorCode

デバイスベンダー定義のエラーが発生した際に、そのエラーコードを取得します。

### 構文

```
public int getOposErrorCode()
```

### 戻り値

発生したデバイスベンダー定義のエラーコードが返ります。

### 補足説明

- デバイスベンダー定義のエラーが発生していない場合、0が返ります。

## authorizeSales

売上を実行します。

本 API の実行結果は、[setAuthorizeSalesEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void authorizeSales(int service, int totalAmount, int sequence) throws Epos2Exception
```

---

### パラメーター

#### service

決済メディアの種類を指定します。

設定値	説明
SERVICE_CREDIT	クレジット
SERVICE_DEBIT	デビット
SERVICE_UNIONPAY	銀聯
SERVICE_EDY	楽天 Edy
SERVICE_ID	iD
SERVICE_NANACO	nanaco
SERVICE_QUICPAY	QUICPay
SERVICE_SUICA	交通系 IC (suica)
SERVICE_WAON	WAON

#### totalAmount

売上金額合計（税込）を指定します。

設定値	説明
1 ~ 9999999	クレジット、デビット、銀聯、iD の場合指定可能
1 ~ 99999	楽天 Edy、交通系 IC の場合指定可能
1 ~ 999999	nanaco, QUICPay, WAON の場合指定可能

#### sequence

信用照会端末が管理するシーケンス番号 4 桁を指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

## authorizeVoid

取消を実行します。

本 API の実行結果は、[setAuthorizeVoidEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void authorizeVoid(int service, int totalAmount, int sequence) throws Epos2Exception
```

### パラメーター

#### service

決済メディアの種類を指定します。

設定値	説明
SERVICE_CREDIT	クレジット
SERVICE_DEBIT	デビット
SERVICE_UNIONPAY	銀聯
SERVICE_ID	iD
SERVICE_QUICPAY	QUICPay
SERVICE_SUICA	交通系 IC (suica)
SERVICE_WAON	WAON

#### totalAmount

売上金額合計（税込）を指定します。

設定値	説明
1 ~ 9999999	クレジット、デビット、銀聯、iD の場合指定可能
1 ~ 99999	交通系 IC の場合指定可能
1 ~ 999999	QUICPay, WAON の場合指定可能

#### sequence

信用照会端末が管理するシーケンス番号 4 枠を指定します。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

## authorizeRefund

返品を実行します。

本 API の実行結果は、[setAuthorizeRefundEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void authorizeRefund(int service, int totalAmount, int sequence) throws Epos2Exception
```

---

### パラメーター

#### service

決済メディアの種類を指定します。

設定値	説明
SERVICE_CREDIT	クレジット
SERVICE_DEBIT	デビット
SERVICE_UNIONPAY	銀聯
SERVICE_ID	iD
SERVICE_WAON	WAON

#### totalAmount

売上金額合計（税込）を指定します。

設定値	説明
1 ~ 9999999	クレジット、デビット、銀聯、iD の場合指定可能
1 ~ 999999	WAON の場合指定可能

#### sequence

信用照会端末が管理するシーケンス番号 4 枠を指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

## authorizeCompletion

承認後売上を実行します。

本 API の実行結果は、[setAuthorizeCompletionEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void authorizeCompletion(int service, int totalAmount,
int sequence) throws Epos2Exception
```

---

### パラメーター

#### service

決済メディアの種類を指定します。

設定値	説明
SERVICE_CREDIT	クレジット

#### totalAmount

売上金額合計（税込）を指定します。

設定値	説明
1 ~ 9999999	クレジットの場合指定可能

#### sequence

信用照会端末が管理するシーケンス番号 4 衔を指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

## accessDailyLog

日計処理を実行します。

本 API の実行結果は、[setAccessDailyLogEventListener API](#) で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void accessDailyLog(int service, int sequence) throws
Epos2Exception
```

---

### パラメーター

#### service

決済メディアの種類を指定します。

設定値	説明
SERVICE_CREDIT	クレジット
SERVICE_DEBIT	デビット
SERVICE_UNIONPAY	銀聯
SERVICE_EDY	楽天 Edy
SERVICE_ID	iD
SERVICE_NANACO	nanaco
SERVICE_QUICPAY	QUICPay
SERVICE_SUICA	交通系 IC (suica)
SERVICE_WAON	WAON

#### sequence

信用照会端末が管理するシーケンス番号 4 桁を指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

## sendDirectIOCommand

OPOS の DirectIO メソッドを送信します。

本 API の実行結果は、[setDirectIOCommandReplyEventListener](#) API で設定したリスナー登録メソッドへ通知されます。

### 構文

```
public void sendDirectIOCommand(int command, int data,
java.lang.String string, int service) throws Epos2Exception
```

### パラメーター

#### command

DirectIO のコマンド番号を指定します。

#### data

DirectIO の数値データを指定します。

#### string

DirectIO の文字列データを指定します。

#### service

決済メディアの種類を指定します。

設定値	説明
SERVICE_CREDIT	クレジット
SERVICE_DEBIT	デビット
SERVICE_UNIONPAY	銀聯
SERVICE_EDY	楽天 Edy
SERVICE_ID	iD
SERVICE_NANACO	nanaco
SERVICE_QUICPAY	QUICPay
SERVICE_SUICA	交通系 IC (suica)
SERVICE_WAON	WAON
SERVICE_POINT	ポイント
SERVICE_COMMON	端末共通

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。

## setTimeout

以下のメソッドに、決済メディア読み取りまでのタイムアウト時間を指定します。

- [authorizeSales](#)
- [authorizeVoid](#)
- [authorizeRefund](#)
- [authorizeCompletion](#)

### 構文

```
public void setTimeout(int timeout) throws Epos2Exception
```

### パラメーター

#### timeout

初期値：0

設定値	説明
1 ~ 999000 の整数	タイムアウト時間 (msec 単位)
0	タイムアウトなし

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。

## getTimeout

[setTimeout](#) で指定したタイムアウト時間取得し、以下のメソッドに渡します。

- [authorizeSales](#)
- [authorizeVoid](#)
- [authorizeRefund](#)
- [authorizeCompletion](#)

---

### 構文

```
public int getTimeout()
```

---

### 戻り値

[setTimeout](#) で指定したタイムアウト時間を返します。

## setTrainingMode

以下のメソッドの動作モードを指定します。

- [authorizeSales](#)
- [authorizeVoid](#)
- [authorizeRefund](#)
- [authorizeCompletion](#)
- [accessDailyLog](#)

---

### 構文

```
public void setTrainingMode(int mode) throws  
Epos2Exception
```

---

### パラメーター

#### mode

初期値 : Cat.FALSE

設定値	説明
Cat.TRUE	練習モード
Cat.FALSE	通常モード

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。

## getTrainingMode

`setTrainingMode` で指定した動作モードを取得し、以下のメソッドに渡します。

- [authorizeSales](#)
- [authorizeVoid](#)
- [authorizeRefund](#)
- [authorizeCompletion](#)
- [accessDailyLog](#)

---

### 構文

```
public int getTrainingMode()
```

---

### 戻り値

`setTrainingMode` で指定した動作モードを返します。

## setAuthorizeSalesEventListener

[authorizeSales](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setAuthorizeSalesEventListener(AuthorizeSalesListener
listener)
```

---

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

---

### リスナーインターフェイス

```
public interface AuthorizeSalesListener extends EventListener
```

---

### リスナー登録メソッド

```
void onCatAuthorizeSales(Cat catObj, int code, int sequence,
int service, AuthorizeResult result)
```

---

### パラメーター

#### catObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	売上処理に成功。
CODE_BUSY	動作中のため実行できない。
CODE_EXCEEDING_LIMIT *	残高不足による失敗。
CODE_DISAGREEMENT *	取引不一致（金額、IC 取扱通番不一致）
CODE_INVALID_CARD *	無効なカード
CODE_RESET *	端末で操作をキャンセルした。
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_DEVICE	デバイスエラー
CODE_ERR_SYSTEM	システムエラー

エラーステータス	説明
CODE_ERR_CENTER *	センター処理でエラーが発生した
CODE_ERR_TIMEOUT	決済メディア読み取り待ちタイムアウト
CODE_ERR_OPOS_CODE	デバイスベンダー定義のエラーコード
CODE_ERR_FAILURE	その他のエラーが発生した。

\*:TM-DT ソフトウェア Ver.4.10 以降では発生しない。

### sequence

信用照会端末管理のシーケンス番号 4 衍を指定します。

### service

[authorizeSales](#) の service パラメーターで指定した決済メディアの種類が入ります。

### result

処理の実行結果が AuthorizeResult 構造体に入ります。

プロパティー	説明
String accountNumber	会員番号またはカード番号の下 4 衍
int settledAmount	売上合計金額（税込）
String slipNumber	伝票番号 5 衍
String kid	KID3 衍
String approvalCode	承認番号 7 衍
String transactionNumber	処理通番 6 衍
int paymentCondition	支払い方法
String voidSlipNumber	取消 / 返品対象の伝票番号 5 衍
int balance	残高

paymentCondition には、支払方法に応じて以下の定数値が入ります。

定数値	説明
PAYMENT_CONDITION_LUMP_SUM	一括払い
PAYMENT_CONDITION_BONUS_1	ボーナス払い パターン 1
PAYMENT_CONDITION_BONUS_2	ボーナス払い パターン 2
PAYMENT_CONDITION_BONUS_3	ボーナス払い パターン 3
PAYMENT_CONDITION_INSTALLMENT_1	分割 パターン 1
PAYMENT_CONDITION_INSTALLMENT_2	分割 パターン 2
PAYMENT_CONDITION_REVOLVING	リボ払い
PAYMENT_CONDITION_COMBINATION_1	ボーナス併用 パターン 1
PAYMENT_CONDITION_COMBINATION_2	ボーナス併用 パターン 2
PAYMENT_CONDITION_DEBIT	決済メディアの種類がデビット

定数値	説明
PAYMENT_CONDITION ELECTRONIC MONEY	決済メディアの種類が電子マネー
PAYMENT_CONDITION OTHER	その他

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- result パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setAuthorizeVoidEventListener

[authorizeVoid](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setAuthorizeVoidEventListener(AuthorizeVoidListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface AuthorizeVoidListener extends EventListener
```

### リスナー登録メソッド

```
void onCatAuthorizeVoid(Cat catObj, int code, int sequence,
int service, AuthorizeResult result)
```

### パラメーター

#### catObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	売上処理に成功。
CODE_BUSY	動作中のため実行できない。
CODE_EXCEEDING_LIMIT *	残高不足による失敗。
CODE_DISAGREEMENT *	取引不一致（金額、IC 取扱通番不一致）
CODE_INVALID_CARD *	無効なカード
CODE_RESET *	端末で操作をキャンセルした。
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_DEVICE	デバイスエラー
CODE_ERR_SYSTEM	システムエラー

エラーステータス	説明
CODE_ERR_CENTER *	センター処理でエラーが発生した
CODE_ERR_TIMEOUT	決済メディア読み取り待ちタイムアウト
CODE_ERR_OPOS_CODE	デバイスベンダー定義のエラーコード
CODE_ERR_FAILURE	その他のエラーが発生した。

\*:TM-DT ソフトウェア Ver.4.10 以降では発生しない。

### sequence

信用照会端末管理のシーケンス番号 4 衍を指定します。

### service

[authorizeVoid](#) の service パラメーターで指定した決済メディアの種類が入ります。

### result

処理の実行結果が AuthorizeResult 構造体に入ります。

プロパティー	説明
String accountNumber	会員番号またはカード番号の下 4 衍
int settledAmount	売上合計金額（税込）
String slipNumber	伝票番号 5 衍
String kid	KID3 衍
String approvalCode	承認番号 7 衍
String transactionNumber	処理通番 6 衍
int paymentCondition	支払い方法
String voidSlipNumber	取消 / 返品対象の伝票番号 5 衍
int balance	残高

paymentCondition には、支払方法に応じて以下の定数値が入ります。

定数値	説明
PAYMENT_CONDITION_LUMP_SUM	一括払い
PAYMENT_CONDITION_BONUS_1	ボーナス払い パターン 1
PAYMENT_CONDITION_BONUS_2	ボーナス払い パターン 2
PAYMENT_CONDITION_BONUS_3	ボーナス払い パターン 3
PAYMENT_CONDITION_INSTALLMENT_1	分割 パターン 1
PAYMENT_CONDITION_INSTALLMENT_2	分割 パターン 2
PAYMENT_CONDITION_REVOLVING	リボ払い
PAYMENT_CONDITION_COMBINATION_1	ボーナス併用 パターン 1
PAYMENT_CONDITION_COMBINATION_2	ボーナス併用 パターン 2
PAYMENT_CONDITION_DEBIT	決済メディアの種類がデビット

定数値	説明
PAYMENT_CONDITION ELECTRONIC MONEY	決済メディアの種類が電子マネー
PAYMENT_CONDITION OTHER	その他

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- result パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setAuthorizeRefundEventListener

[authorizeRefund](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setAuthorizeRefundEventListener(AuthorizeRefundListener
listener)
```

---

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

---

### リスナーインターフェイス

```
public interface AuthorizeRefundListener extends
EventListener
```

---

### リスナー登録メソッド

```
void onCatAuthorizeRefund(Cat catObj, int code, int
sequence, int service, AuthorizeResult result)
```

---

### パラメーター

#### catObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	売上処理に成功。
CODE_BUSY	動作中のため実行できない。
CODE_EXCEEDING_LIMIT *	残高不足による失敗。
CODE_DISAGREEMENT *	取引不一致（金額、IC 取扱通番不一致）
CODE_INVALID_CARD *	無効なカード
CODE_RESET *	端末で操作をキャンセルした。
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_DEVICE	デバイスエラー
CODE_ERR_SYSTEM	システムエラー

エラーステータス	説明
CODE_ERR_CENTER *	センター処理でエラーが発生した
CODE_ERR_TIMEOUT	決済メディア読み取り待ちタイムアウト
CODE_ERR_OPOS_CODE	デバイスベンダー定義のエラーコード
CODE_ERR_FAILURE	その他のエラーが発生した。

\*:TM-DT ソフトウェア Ver.4.10 以降では発生しない。

### sequence

信用照会端末管理のシーケンス番号 4 衍を指定します。

### service

[authorizeRefund](#) の service パラメーターで指定した決済メディアの種類が入ります。

### result

処理の実行結果が AuthorizeResult 構造体に入ります。

プロパティー	説明
String accountNumber	会員番号またはカード番号の下 4 衍
int settledAmount	売上合計金額（税込）
String slipNumber	伝票番号 5 衍
String kid	KID3 衍
String approvalCode	承認番号 7 衍
String transactionNumber	処理通番 6 衍
int paymentCondition	支払い方法
String voidSlipNumber	取消 / 返品対象の伝票番号 5 衍
int balance	残高

paymentCondition には、支払方法に応じて以下の定数値が入ります。

定数値	説明
PAYMENT_CONDITION_LUMP_SUM	一括払い
PAYMENT_CONDITION_BONUS_1	ボーナス払い パターン 1
PAYMENT_CONDITION_BONUS_2	ボーナス払い パターン 2
PAYMENT_CONDITION_BONUS_3	ボーナス払い パターン 3
PAYMENT_CONDITION_INSTALLMENT_1	分割 パターン 1
PAYMENT_CONDITION_INSTALLMENT_2	分割 パターン 2
PAYMENT_CONDITION_REVOLVING	リボ払い
PAYMENT_CONDITION_COMBINATION_1	ボーナス併用 パターン 1
PAYMENT_CONDITION_COMBINATION_2	ボーナス併用 パターン 2
PAYMENT_CONDITION_DEBIT	決済メディアの種類がデビット

定数値	説明
PAYMENT_CONDITION ELECTRONIC MONEY	決済メディアの種類が電子マネー
PAYMENT_CONDITION OTHER	その他

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- result パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setAuthorizeCompletionEventListener

[authorizeCompletion](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setAuthorizeCompletionEventListener(AuthorizeCompletionL
istener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface AuthorizeCompletionListener extends
EventListener
```

### リスナー登録メソッド

```
void onCatAuthorizeCompletion(Cat catObj, int code, int
sequence, int service, AuthorizeResult result)
```

### パラメーター

#### catObj

コーレバッケ元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	売上処理に成功。
CODE_BUSY	動作中のため実行できない。
CODE_EXCEEDING_LIMIT *	残高不足による失敗。
CODE_DISAGREEMENT *	取引不一致（金額、IC 取扱通番不一致）
CODE_INVALID_CARD *	無効なカード
CODE_RESET *	端末で操作をキャンセルした。
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_DEVICE	デバイスエラー
CODE_ERR_SYSTEM	システムエラー

エラーステータス	説明
CODE_ERR_CENTER *	センター処理でエラーが発生した
CODE_ERR_TIMEOUT	決済メディア読み取り待ちタイムアウト
CODE_ERR_OPOS_CODE	デバイスベンダー定義のエラーコード
CODE_ERR_FAILURE	その他のエラーが発生した。

\*:TM-DT ソフトウェア Ver.4.10 以降では発生しない。

### sequence

信用照会端末管理のシーケンス番号 4 衍を指定します。

### service

[authorizeCompletion](#) の service パラメーターで指定した決済メディアの種類が入ります。

### result

処理の実行結果が AuthorizeResult 構造体に入ります。

プロパティー	説明
String accountNumber	会員番号またはカード番号の下 4 衍
int settledAmount	売上合計金額（税込）
String slipNumber	伝票番号 5 衍
String kid	KID3 衍
String approvalCode	承認番号 7 衍
String transactionNumber	処理通番 6 衍
int paymentCondition	支払い方法
String voidSlipNumber	取消 / 返品対象の伝票番号 5 衍
int balance	残高

paymentCondition には、支払方法に応じて以下の定数値が入ります。

定数値	説明
PAYMENT_CONDITION_LUMP_SUM	一括払い
PAYMENT_CONDITION_BONUS_1	ボーナス払い パターン 1
PAYMENT_CONDITION_BONUS_2	ボーナス払い パターン 2
PAYMENT_CONDITION_BONUS_3	ボーナス払い パターン 3
PAYMENT_CONDITION_INSTALLMENT_1	分割 パターン 1
PAYMENT_CONDITION_INSTALLMENT_2	分割 パターン 2
PAYMENT_CONDITION_REVOLVING	リボ払い
PAYMENT_CONDITION_COMBINATION_1	ボーナス併用 パターン 1
PAYMENT_CONDITION_COMBINATION_2	ボーナス併用 パターン 2
PAYMENT_CONDITION_DEBIT	決済メディアの種類がデビット

定数値	説明
PAYMENT_CONDITION ELECTRONIC MONEY	決済メディアの種類が電子マネー
PAYMENT_CONDITION OTHER	その他

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- result パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setAccessDailyLogEventListener

[accessDailyLog](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setAccessDailyLogEventListener(AccessDailyLogListener
listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface AccessDailyLogListener extends EventListener
```

### リスナー登録メソッド

```
void onCatAccessDailyLog(Cat catObj, int code, int sequence,
int service, ArrayList<DailyLog> dailyLog)
```

### パラメーター

#### catObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
CODE_SUCCESS	売上処理に成功。
CODE_BUSY	動作中のため実行できない。
CODE_ERR_PARAM	不正なパラメーターが渡された。
CODE_ERR_DEVICE	デバイスエラー
CODE_ERR_SYSTEM	システムエラー
CODE_ERR_CENTER *	センター処理でエラーが発生した
CODE_ERR_OPOS_CODE	デバイスベンダー定義のエラーコード
CODE_ERR_FAILURE	その他のエラーが発生した。

\*:TM-DT ソフトウェア Ver.4.10 以降では発生しない。

**sequence**

信用照会端末管理のシーケンス番号 4 衍を指定します。

**service**

[accessDailyLog](#) の service パラメーターで指定した決済メディアの種類が入ります。

**dailyLog**

日計処理の結果が、配列として返ります。

設定値	説明
String kid	KID3 衍文字列
long salesCount	売上データ件数 0 ~ 9999999999
long salesAmount	売上データ金額 0 ~ 9999999999
long voidCount	取消、返品データ件数 0 ~ 9999999999
long voidAmount	取消、返品データ金額 0 ~ 9999999999

**補足説明**

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- dailyLog パラメーターは null になる可能性があります。null チェックしてから使用してください。

## setDirectIOCommandReplyEventListener

[sendDirectIOCommand](#) の実行結果を通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void
setDirectIOCommandReplyEventListener(DirectIOCommand
ReplyListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface DirectIOCommandReplyListener extends
EventListener
```

### リスナー登録メソッド

```
void onCatDirectIOCommandReply(Cat catObj, int code, int
command, int data, java.lang.String string, int sequence, int
service, DirectIOResult result)
```

### パラメーター

#### catObj

コールバック元のオブジェクトが入ります。

#### code

実行結果が入ります。

エラーステータス	説明
cat.Epos2CallbackCode.CODE_SUCCESS	成功。
cat.Epos2CallbackCode.CODE_BUSY	動作中のため実行できない。
cat.Epos2CallbackCode.CODE_ERR_PARAM	パラメーターエラー
cat.Epos2CallbackCode.CODE_ERR_SYSTEM	システムエラー
cat.Epos2CallbackCode.CODE_ERR_DEVICE	デバイスエラー
cat.Epos2CallbackCode.CODE_ERR_TIMEOUT	タイムアウトエラー
cat.Epos2CallbackCode.CODE_ERR_OPOSCODE	デバイスベンダー定義のエラー
cat.Epos2CallbackCode.CODE_ERR_FAILURE	その他のエラー

**command**

DirectIO のコマンド番号が入ります。

**data**

DirectIO の数値データが入ります。

**string**

DirectIO の文字列データが入ります。

**sequence**

信用照会端末管理のシーケンス番号 4 衔を指定します。

**service**

[sendDirectIOCommand](#) の service パラメーターで指定した決済メディアの種類が入ります。

**result**

処理の実行結果が DirectIOPResult 構造体に入ります。

プロパティ	説明
String accountNumber	会員番号またはカード番号の下 4 衔
int settledAmount	売上合計金額（税込）
String slipNumber	伝票番号 5 衔
String transactionNumber	処理通番 6 衔
int paymentCondition	支払い方法
int balance	残高
String additionalSecurityInformation	拡張結果情報

paymentCondition には、支払方法に応じて以下の定数値が入ります。

定数値	説明
PAYMENT_CONDITION_LUMP_SUM	一括払い
PAYMENT_CONDITION_BONUS_1	ボーナス払い パターン 1
PAYMENT_CONDITION_BONUS_2	ボーナス払い パターン 2
PAYMENT_CONDITION_BONUS_3	ボーナス払い パターン 3
PAYMENT_CONDITION_INSTALLMENT_1	分割 パターン 1
PAYMENT_CONDITION_INSTALLMENT_2	分割 パターン 2
PAYMENT_CONDITION_REVOLVING	リボ払い
PAYMENT_CONDITION_COMBINATION_1	ボーナス併用 パターン 1
PAYMENT_CONDITION_COMBINATION_2	ボーナス併用 パターン 2
PAYMENT_CONDITION_DEBIT	決済メディアの種類がデビット
PAYMENT_CONDITION ELECTRONIC MONEY	決済メディアの種類が電子マネー
PAYMENT_CONDITION_OTHER	その他

---

## 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に `listener` に `null` を指定した場合は、リスナー登録メソッドの登録を解除します。
- `result` パラメーターは `null` になる可能性があります。`null` チェックしてから使用してください。

## setStatusUpdateEventListener

OPOS ドライバーの StatusUpdateEvent を通知するイベントのリスナー登録メソッドを登録します。 StatusUpdateEvent は、信用照会端末の状態が変化した際に発生します。 詳細は、OPOS ドライバーのマニュアルを参照してください。

### 構文

```
public void
setStatusUpdateEventListener(StatusUpdateListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface StatusUpdateListener extends EventListener
```

### リスナー登録メソッド

```
void onCatStatusUpdate(Cat catObj, int status)
```

### パラメーター

#### catObj

コールバック元のオブジェクトが入ります。

#### status

StatusUpdateEvent の値が入ります。

ステータス	説明
cat.SUE_LOGSTATUS_OK	取引ログは充分な空き容量があります。
cat.SUE_LOGSTATUS_NEARFULL	取引ログはフルに近い状態です。
cat.SUE_LOGSTATUS_FULL	取引ログは一杯です。
cat.SUE_POWER_ONLINE	デバイスは電源オンでかつレディ状態です。
cat.SUE_POWER_OFF_OFFLINE	デバイスは電源オフ、またはオフライン状態です。

※ デバイスベンダー固定の定義がある場合、上記以外の値が通知されます。

### 補足説明

□ 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。

- 本 API 実施時に `listener` に `null` を指定した場合は、リスナー登録メソッドの登録を解除します。

# OtherPeripheral クラス

## OtherPeripheral

OtherPeripheral クラスを初期化します。

### 構文

```
public OtherPeripheral(Context context) throws  
Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

## connect

周辺機器との通信を開始します。

### 構文

```
public void connect(String target, int timeout) throws  
Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<デバイス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[local_other]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
Otherperipheral.PARAM_DEFAULT	既定値 (15000) を指定

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	オープンに失敗した。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_ILLEGAL	既に通信されている状態で、再度通信開始しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_NOT_FOUND	デバイスが見つからない。

エラーステータス	説明
ERR_IN_USE	デバイスが使用中。
ERR_TYPE_INVALID	デバイスの種類が違う。

## 補足説明

- デバイス ID は、周辺機器を TM プリンター（親機）へ登録する際に設定する論理デバイス名です。  
詳細は、TM-DT シリーズ周辺機器制御ガイドを参照してください。
- マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [disconnect](#) を実行できません。本 API の終了を待ってから実行してください。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

周辺機器との通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	通信されていない状態で、通信を終了しようとした。
ERR_MEMORY	必要なメモリーを確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_DISCONNECT	デバイスの切断に失敗した。

### 補足説明

マルチスレッドで複数のプリンターや周辺機器と接続して運用する場合、本 API の処理が終わるまで他のプリンターや周辺機器に本 API や [connect](#) を実行できません。本 API の終了を待ってから実行してください。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public OtherPeripheralStatusInfo getStauts()
```

### 戻り値

現在の状態が OtherPeripheralStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
(int)connection : 接続状態	OtherPeripheral.TRUE	接続中
	OtherPeripheral.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## sendData

周辺機器へデータを送信します。

デバイス制御プログラム側の任意関数を指定し、プロパティ名と値のセットからなる JSON データを渡します。

詳細は、TM-DT シリーズ周辺機器制御ガイドを参照してください。

本 API の実行結果は、[setReceiveEventLister](#) API で設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void sendData(String methodName, String data) throws
Epos2Exception
```

---

### パラメーター

#### methodName

実行する関数名を指定します。

#### data

関数へ渡すパラメーターを JSON 文字列で指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

## setReceiveEventListener

周辺機器からの受信データを通知するイベントのリスナー登録メソッドを登録します。

### 構文

```
public void setReceiveEventListener(ReceiveListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ReceiveListener extends EventListener
```

### リスナー登録メソッド

```
void onOtherReceive(OtherPeripheral otherObj,  
java.lang.String eventName, String data)
```

### パラメーター

#### otherObj

コールバック元のオブジェクトが入ります。

#### eventName

イベント名が入ります。

#### data

受信データが JSON 文字列で入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。

# CommBox クラス

## CommBox

CommBox クラスを初期化します。

### 構文

```
public CommBox(Context context) throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。  
ログ出力機能を使用しない場合は null を指定できます。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_MEMORY	必要なメモリーを確保できなかった。

### 補足説明

繰り返して処理を実行する場合、インスタンスの生成と破棄は繰り返し処理の外で行い、短い間隔で繰り返さないでください。  
OS によるメモリの解放処理が Epson ePOS SDK の処理に影響し、不正な動作となる可能性があります。

## connect

コミュニケーションボックスとの通信を開始します。

### 構文

```
public void connect(String target, int timeout, String myId)
throws Epos2Exception
```

### パラメーター

#### target

プリンターとの接続方法を文字列で指定します。

接続方法は、以下の構文で指定してください。

<接続形式>:<識別子>[<ボックス ID>]

I/F	接続形式	識別子	例
Wi-Fi/ Ethernet	• "TCP" • "TCPS"	• IPv4 形式の IP アドレス • MAC アドレス • ホスト名	"TCP:192.168.192.168[Box1]"

SSL/TLS 通信でプリンターと接続するには、接続形式に TCPS を指定してください。

#### timeout

プリンターと通信確立するための最大待ち時間（msec 単位）を指定します。

設定値	説明
1000 ~ 300000 の整数	エラーを返すまでの最大待ち時間（msec 単位）
CommBox.PARAM_DEFAULT	既定値 (15000) を指定

#### myId

コミュニケーションボックス内で自身を識別するためのメンバー ID を指定します。

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	デバイスをオープンできなかった。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

エラーステータス	説明
ERR_ALREADY_OPENED	既にコミュニケーションボックスがオープンしている。
ERR_ALREADY_USED	指定したメンバー ID は既に使用されている。
ERR_BOX_COUNT_OVER	コミュニケーションボックスを作成可能な最大数を超えている。
ERR_BOX_CLIENT_OVER	コミュニケーションボックスに所属可能な最大数を超えている。

### 補足説明

- 本 API は TM プリンター（親機）と周辺機器の接続状態に関わらず成功します。
- プリンター以外のデバイスは排他的にロックします。

## disconnect

コミュニケーションボックスとの通信を終了します。

### 構文

```
public void disconnect() throws Epos2Exception
```

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

## getStatus

現在のステータス情報を取得します。

### 構文

```
public CommBoxStatusInfo getStatus ()
```

### 戻り値

現在の状態が CommBoxStatusInfo 型として返ります。

オブジェクトにはプロパティーとして以下のステータス情報が格納されます。

格納されているステータス情報は、各プロパティーのゲッターメソッドから取得します。

状態	ステータス	説明
connection : 接続状態	CommBox.TRUE	接続中
	CommBox.FALSE	切断中

### 補足説明

- ステータス用オブジェクトは、本 API 実行時のステータスが入り、更新は行いません。
- コールバック関数内で本関数を実行した場合、null が返ります。

## getCommHistory

コミュニケーションボックスの送信履歴を取得します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void getCommHistory(GetCommHistoryCallback  
callback) throws Epos2Exception
```

---

### パラメーター

#### callback

通知先メソッドを持つオブジェクトを指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_CONNECT	通信できなかった。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。

---

### リスナーインターフェイス

```
public interface GetCommHistoryCallback extends  
EventListerner
```

---

### リスナー登録メソッドの定義

```
void onGetCommHistory(CommBox commBoxObj, int code,  
ArrayList<HashMap<String, String>> historyList;
```

---

### パラメーター

#### commBoxObj

コールバック元のオブジェクトが入ります。

**code**

エラーステータス	説明
SUCCESS	処理に成功した。
CODE_ERR_FAILURE	その他のエラーが発生した。

**historyList**

コミュニケーションボックスの通信履歴が入ります。

historyList[index]	
キー	説明
String senderId	送信元のメンバー ID が入ります。 connect メソッド実行時に、メンバー ID の指定がない場合、” ”（空文字）がセットされます。
String receiverId	送信先のメンバー ID が入ります。 コミュニケーションボックス内のすべてのメンバーに送信した場合、” ”（空文字）がセットされます。
String message	受信したデータがセットされます。

## sendMessage

メッセージをコミュニケーションボックスに送信します。

本 API のエラーステータスは、デバイスに対するデータ送信の結果です。

本 API によるデバイスの処理結果は、設定したリスナー登録メソッドへ通知されます。

---

### 構文

```
public void sendMessage (String message, String
targetId,SendMessageCallback callback) throws
Epos2Exception
```

---

### パラメーター

#### message

コミュニケーションボックスに送信するデータを指定します。

#### targetId

送信先のメンバー ID を指定します。

null または空文字を指定した場合、送信者を除くコミュニケーションボックス内の全てのメンバーに送信します。

#### callback

通知先メソッドを持つオブジェクトを指定します。

---

### 例外

処理に失敗した場合、以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_MEMORY	処理に必要なメモリーが確保できなかった。
ERR_FAILURE	その他のエラーが発生した。
ERR_PROCESSING	処理が実行できなかった。
ERR_ILLEGAL	通信が開始されていない状態で、本 API が呼び出された。

---

## リスナーインターフェイス

```
public interface SendMessageCallback extends EventListener
```

---

### リスナー登録メソッドの定義

```
void onCommBoxSendMessage (CommBox commBoxObj,int code, int count)
```

---

### パラメーター

#### commBoxObj

コールバック元のオブジェクトが入ります。

#### code

エラーステータス	説明
CODE_SUCCESS	処理に成功した。
CODE_ERR_NOT_FOUND	指定したメンバー ID が見つからない。
CODE_ERR_FAILURE	その他のエラーが発生した。

#### count

メッセージの送信を試みたクライアント数が入ります。

## setReceiveEventListener

コミュニケーションボックスのメッセージ受信イベントのリスナー登録メソッドを登録します。

### 構文

```
public void setReceiveEventListener (ReceiveListener listener)
```

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

### リスナーインターフェイス

```
public interface ReceiveListener extends EventListener
```

### リスナー登録メソッド

```
void onCommBoxReceive(CommBox commBoxObj, String
senderId, String receiverId String message;
```

### パラメーター

#### commBoxObj

コールバック元のオブジェクトが入ります。

#### senderId

送信元のメンバー ID が入ります。

送信元のメンバーが connect メソッド実行時に、メンバー ID の指定がない場合、" " (空文字) がセットされます。

#### receiverId

送信先のメンバー ID が入ります。

コミュニケーションボックス内のすべてのメンバーに送信した場合、" " (空文字) がセットされます。

#### message

受信したメッセージが入ります。

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- ePOS-Device XML 制御で通信異常が発生した場合、通信切断を通知しません。  
切断イベントを通知できるように、[setConnectionEventListenner](#) を実装してください。

# クラス共通 API

## getAdmin

プリンターに設定された管理者名を取得します。

### 構文

```
public String getAdmin()
```

### 戻り値

管理者名の文字列。

### 補足説明

- 設定されていない場合、TM プリンターの場合は” ”（空文字）が返ります。
- 管理者名は EPSON TMNet WebConfig で設定します。
- 本 API は Bluetooth<sup>®</sup> 通信では使用できません。

## getLocation

プリンターに設定された設置場所情報を取得します。

### 構文

```
public String getLocation()
```

### 戻り値

設置場所の文字列。

### 補足説明

- 設定されていない場合、TM プリンターの場合は " " (空文字) が返ります。
- 設置場所情報は EPSON TMNet WebConfig で設定します。
- 本 API は Bluetooth<sup>®</sup> 通信では使用できません。

## setConnectionEventListener

再接続処理が開始されたことを受け取るリスナー登録メソッドを登録します。

setConnectionEventListener が発生した際に、再接続が開始された旨を示すメッセージを表示するなどの処理を行ってください。

---

### 構文

```
public void setConnectionEventListener (ConnectionListener  
listener)
```

---

### パラメーター

#### listener

通知先メソッドを持つオブジェクトを指定します。

---

### リスナーインターフェイス

```
public interface ConnectionListener extends EventListener
```

---

### リスナー登録メソッド

```
void onConnection(Object deviceObj, int eventType)
```

---

### パラメーター

#### deviceObj

コーレバッケ元のオブジェクトが入ります。

#### eventType

発生したイベントの種類が入ります。

イベントの種類	説明
xxx.EVENT_RECONNECTING	再接続開始イベント
xxx.EVENT_RECONNECT	再接続完了イベント
xxx.EVENT_DISCONNECT	切断イベント

---

### 補足説明

- 本 API を複数回実施した場合は、後に指定されたリスナー登録メソッドで上書きされます。
- 本 API 実施時に listener に null を指定した場合は、リスナー登録メソッドの登録を解除します。
- 切断発生後、切断イベントが発生するまで約 30 秒かかります。
- xxx.EVENT\_RECONNECTING 発生時は何も呼び出さないでください。  
xxx.EVENT\_RECONNECT もしくは xxx.EVENT\_DISCONNECT が発生するまで待ってください。

- xxx.EVENT\_DISCONNECT は、以下の場合に発生します。
  - disconnect の実行時
  - xxx.EVENT\_RECONNECTING 発生後の接続処理失敗時
- xxx.EVENT\_DISCONNECT 発生時、disconnect の呼び出しは不要です。
  - disconnect を呼び出した場合、例外に ERR\_ILLEGAL を格納した Epos2Exception が発生します。

# Discovery クラス

## start

指定されたデバイス種別の検索を開始します。  
検出デバイスごとにイベントが通知されます。

### 構文

```
public static synchronized void start(Context context,
FilterOption filterOption,DiscoveryListener listener) throws
Epos2Exception
```

### パラメーター

#### context

呼び元の Context クラスのインスタンスを設定します。  
例) Activity 内で getBaseContext() で得た Context を設定します。

#### filterOption

検索結果をフィルタリングする場合、FilterOption 型にフィルタオプションを設定し、パラメーターに指定します。null を指定した場合、既定値の設定で検索されます。

FilterOption 型は以下の情報が格納されます。

portType

検索対象のポートを選択します。

設定値	説明
Discovery.PORTTYPE_ALL (既定値)	TCP、Bluetooth® 経由で接続可能な全てのデバイスを検索します。
Discovery.PORTTYPE_TCP	ネットワークに接続されたデバイスを検索します。
Discovery.PORTTYPE_BLUETOOTH	Bluetooth® 接続可能なデバイスを検索します。
Discovery.PORTTYPE_USB	USB 接続可能なデバイスを検索します。

broadcast

TCP 検索する際の Broadcast Address を文字列で指定します。

設定値	説明
” 255.255.255.255 ” (既定値)	-

deviceModel

検索するデバイスモデルを指定します。

設定値	説明
Discovery.MODEL_ALL	全てのモデルを検索します。

□ `epsonFilter`

エプソン製プリンターで検索をかけます。

設定値	説明
<code>Discovery.FILTER_NAME</code> (既定値)	エプソン製プリンターでフィルタリングをします。
<code>Discovery.FILTER_NONE</code>	フィルタリングしません。

□ `deviceType`

検索するデバイスの種類を指定します。

設定値	説明
<code>Discovery.TYPE_ALL</code> (既定値)	全てのデバイスを検索します。
<code>Discovery.TYPE_PRINTER</code>	プリンターを検索します。
<code>Discovery.TYPE_DISPLAY</code>	カスタマーディスプレイを検索します。
<code>Discovery.TYPE_KEYBOARD</code>	キーボードを検索します。
<code>Discovery.TYPE_SCANNER</code>	バーコードスキャナーを検索します。
<code>Discovery.TYPE_SERIAL</code>	シリアル通信デバイスを検索します。
<code>Discovery.TYPE_CCHANGER</code>	自動つり銭機を検索します。
<code>Discovery.TYPE_POS_KEYBOARD</code>	POS キーボードを検索します。
<code>Discovery.TYPE_CAT</code>	信用照会端末を検索します。
<code>Discovery.TYPE_MSR</code>	MSR を検索します。
<code>Discovery.TYPE_OTHER_PERIPHERAL</code>	その他の周辺機器を検索します。

□ `bondedDevices`

Bluetooth® 接続可能なデバイスを検索する場合の検索対象を指定します。

設定値	説明
<code>Discovery.TRUE</code>	接続可能なデバイスと、以前に接続したことがあるデバイスを検索します。
<code>Discovery.FALSE</code> (既定値)	接続可能なデバイスを検索します。

## listener

通知先メソッドを持つオブジェクトを指定します。

## 例外

処理に失敗した場合は以下のエラー値の `Epos2Exception` が発生します。

エラーステータス	説明
<code>ERR_PARAM</code>	不正なパラメーターが渡された。
<code>ERR_ILLEGAL</code>	既に検索している状態で、再度検索を開始しようとした。
<code>ERR_MEMORY</code>	処理に必要なメモリーが確保できなかった。
<code>ERR_FAILURE</code>	その他のエラーが発生した。

エラーステータス	説明
ERR_PROCESSING	処理が実行できなかった。

## リスナーインターフェイス

public interface DiscoveryListener extends EventListener

### リスナー登録メソッド

void onDiscovery(DeviceInfo deviceInfo);

### パラメーター

#### DeviceInfo

発見したデバイスの デバイス情報が DeviceInfo 型として格納されます。

以下のデバイス情報が格納されます。

□ deviceType

デバイスの種類が格納されます。

デバイスの種類	ステータス
プリンターの場合	Discovery.TYPE_PRINTER
カスタマーディスプレイの場合	Discovery.TYPE_DISPLAY
キーボードの場合	Discovery.TYPE_KEYBOARD
バーコードスキャナーの場合	Discovery.TYPE_SCANNER
シリアル通信デバイスの場合	Discovery.TYPE_SERIAL
自動つり銭機の場合	Discovery.TYPE_CCHANGER
POS キーボードの場合	Discovery.TYPE_POS_KEYBOARD
信用照会端末の場合	Discovery.TYPE_CAT
MSR の場合	Discovery.TYPE_MSR
他の周辺機器の場合	Discovery.TYPE_OTHER_PERIPHERAL

□ target

connectAPI で指定可能なデバイスの接続先が文字列で格納されます。

例)

"TCP:192.168.192.168"  
 "BT:00:22:15:7D:70:9C"  
 "USB:/dev/udev/xxxxxxxxxx"  
 "TCP:192.168.192.168[local\_printer]"  
 "TCP:192.168.192.168[local\_display]"  
 "TCP:12:34:56:78:56:78"

□ deviceName

デバイスに設定されている名前が格納されます。

取得できない場合、""（空文字）が格納されます。

例)

プリンターの場合："TM-T88V"

- ipAddress
  - IP アドレスが格納されます。
  - 取得できない場合、""（空文字）が格納されます。
  - 例)
    - TCP の場合："192.168.192.168"
    - BT の場合："（空文字）
- macAddress
  - MAC アドレスが格納されます。
  - 取得できない場合、""（空文字）が格納されます。
  - 例)
    - TCP の場合："12:34:56:78:56:78"
    - BT の場合："（空文字）
- bdAddress
  - BD アドレスが格納されます。
  - 取得できない場合、""（空文字）が格納されます。
  - 例)
    - TCP の場合："（空文字）
    - BT の場合："12:34:56:78:56:78"

---

## 補足説明

- 本 API の検索結果は、検出したデバイスごとにコールバック関数へ通知します。  
オープン済みデバイスも通知されます。ただし、すでにオープン済みの USB デバイス、Bluetooth<sup>®</sup> デバイスは通知されません。
- TCP デバイスの検索において、同一 IP アドレスのデバイスが複数存在する場合、デバイス情報は一つのデバイスとして扱われます。また、そのデバイス情報の一貫性は保証しません。
- 検索を開始し処理が終了したら、必ず検索を終了してください。
- Android Studio の Simulator では検索できません。

## stop

検索を終了します。

### 構文

```
public static synchronized void stop() throws Epos2Exception
```

### 例外

処理に失敗した場合は以下のエラー値の Epos2Exception が発生します。

エラーステータス	説明
ERR_ILLEGAL	検索していない状態で、検索終了を実行した。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

検索を開始し処理が終了したら、必ず検索を終了してください。

# Epos2Exception クラス

## getErrorHandler

例外からエラーステータスを取得します。

### 構文

```
public int getErrorHandler()
```

### 戻り値

例外発生元の API で設定されたエラーステータスが返ります。

[Error Status と対処方法を参照してください。](#)

# Log クラス

## setLogSettings

ログ出力機能を設定します。

### 構文

```
public static void setLogSettings(Context context, int period,
int output, String ipAddress, int port, int logSize, int logLevel)
throws Epos2Exception
```

### パラメーター

#### context

アプリケーションのコンテキストを指定します。

#### period

ログ出力機能の設定が一時的か / 恒久的かを指定します。

一時的の場合は設定ファイルを作成せず、恒久的な場合に設定ファイルを作成します。

設定値	説明
PERIOD_TEMPORARY	設定を一時的に反映する
PERIOD_PERMANENT	設定を恒久的に反映する

#### output

ログ出力機能の有効 / 無効、ログの出力先を指定します。

設定値	説明
OUTPUT_DISABLE	ログ出力機能を無効
OUTPUT_STORAGE	端末のストレージに出力
OUTPUT_TCP	TCP で出力

#### ipAddress

TCP 通信用の IP アドレスを指定します。

IPv4 形式の IP アドレスを指定します。

ログ出力機能を無効にする場合と、ログ出力先を端末ストレージにする場合は null でもよいです。

#### port

TCP 通信用のポート番号を指定します。

ログ出力機能を無効にする場合と、ログ出力先を端末ストレージにする場合も、範囲内の任意の値を指定します。

設定値	説明
0 ~ 65535 の整数	ポート番号

### logSize

ストレージへ保存するログの最大容量を指定します。

ログ出力機能を無効にする場合と、ログ出力先を TCP にする場合も、範囲内の任意の値を指定します。

設定値	説明
1 ~ 50 の整数	保存するログの最大容量 (MB 単位)

### logLevel

ログの出力レベルを指定します。

設定値	説明
LOGLEVEL_LOW	低レベル

### 例外

処理に失敗した場合、以下のエラーステータスの Epos2Exception が発生します。

エラーステータス	説明
ERR_PARAM	不正なパラメーターが渡された。
ERR_FAILURE	その他のエラーが発生した。

### 補足説明

- period に PERIOD\_PERMANENT を指定する場合、アプリケーションにストレージアクセスのパーミッションを設定してください。
- output に OUTPUT\_STORAGE を指定する場合、アプリケーションにストレージアクセスのパーミッションを設定してください。
- output に OUTPUT\_TCP を指定する場合、アプリケーションにネットワークアクセスの パーミッションを設定してください。

## SdkVersion

Epson ePOS SDK のバージョンを取得します。

### 構文

#### Getter

```
public static String getSdkVersion ()
```

### 補足説明

使用している SDK のバージョンが文字列で取得できます。

例：“Ver.2.0.0”

# EasySelect クラス

## parseNFC

NFC タグのデータを解析します。

### 構文

```
public ArrayList<EasySelectInfo>parseNFC(Tag tag, int
timeout)
```

### パラメーター

#### tag

Android の API を使用して読み取った NFC タグデータを指定します。

#### timeout

NFC タグ解析時の待ち時間（msec 単位）を指定します。

設定値	説明
0 ~ 60000 の整数	解析待ち時（msec 単位）
PARSE_NFC_TIMEOUT_DEFAULT	規定値（500）を指定

### 戻り値

NFC タグの解析結果が EasySelectInfo クラスのメンバー変数に返ります。解析に失敗した場合、null が返ります。

### 補足説明

- TM プリンターに設定されている NFC タグ情報は以下のとおりです。

種類	説明
子機プリンター（有線・無線）	エプソン独自データ（製品名、シリアル番号など）
Bluetooth® 対応プリンター	Bluetooth® 用 NFC 標準規格データ（BTSSP）

- 子機プリンターの場合、parseNFC API の中でブロードキャスト通信を行い、MAC アドレスを設定します。timeout 時間に子機プリンターから応答がないと戻り値に null が返ります。parseNFC 実行時には子機プリンターは通信可能な状態に設定してください。

## parseQR

QR コードのデータを解析します。

### 構文

```
public EasySelectInfo parseQR(String data)
```

### パラメーター

#### data

Android の API を使って読み取った QR コードのデータを指定します。

### 戻り値

QR コードの解析結果が EasySelectInfo クラスのメンバー変数に返ります。

解析に失敗した場合、null が返ります。

### 補足説明

QR コードには、createQR API で作成したものを印刷して使用してください。

QR コードをステータスシートに印刷する製品もあります。詳細は各製品の詳細取扱説明書を参照してください。

## createQR

parseQR で解析可能な QR コード用の印刷データを作成します。

### 構文

```
public String createQR(String printerName, int deviceType,
String macAddress)
```

### パラメーター

#### printerName

プリンターナー名を指定します。

#### deviceType

プリンターの種類を指定します。

設定値	説明
EasySelectDeviceType.TCP	子機プリンター（有線・無線）
EasySelectDeviceType.Bluetooth	Bluetooth® 対応プリンター

#### macAddress

MAC アドレス（子機プリンター）または BD アドレス（Bluetooth® 対応プリンター）を指定します。アドレスの指定方法は以下のとおりです。

フォーマット	説明
00:11:22:33:44:55	“：“コロン区切り
00-11-22-33-44-55	“-”ハイフン区切り
001122334455	区切りなし

### 戻り値

QR コードの印刷用データが返されます。このデータを addSymbol に指定することで QR コードを印刷できます。

印刷用データの作成に失敗した場合、null が返されます。

### 補足説明

設定するプリンターナー名、MAC アドレス、BD アドレスはステータスシートを印刷すれば確認できます。  
ステータスシートの印刷方法は各製品の詳細取扱説明書を参照してください。

# EasySelectInfo クラス

## deviceType

プリンターの種類が格納されます。

格納されるデータ	説明
EasySelectDeviceType.TCP	子機プリンター
EasySelectDeviceType.Bluetooth	Bluetooth® 対応プリンター

## 構文

```
int deviceType
```

## printerName

プリンターナー名が格納されます。

## 構文

```
String printerName
```

## 補足説明

UB-R04/UB-E04（インターフェイスボード）が装着された TM プリンターの場合、NFC タグの printerName は “” になります。

## macAddress

MAC アドレスまたは BD アドレスが格納されます。

## 構文

```
String macAddress
```

# Status一覧

## Error Status と対処方法

エラーステータス	要因	対処方法
ERR_PARAM	不正なパラメーターが渡された。 <例> サポート範囲外の数値が指定された。	パラメーターの指定値を確認してください。
ERR_CONNECT	デバイスとの通信に失敗した。 <例> プリンターとの接続に失敗した。	各クラスの disconnect 実行後に connect を実行し、通信を復帰させてください。 Bluetooth® で接続する場合、Android OS が自動で再接続を試みます。20 秒以上経過しても本エラーステータスが発生する場合に、上記方法で通信を復帰させてください。
ERR_TIMEOUT	指定時間内にデバイスと通信できなかつた。	タイムアウト時間を確認してください。 タイムアウト時間は、印刷所要時間以上に設定してください。
ERR_MEMORY	処理に必要なメモリーが確保できなかつた。	不要なアプリケーションソフトを終了してください。
ERR_ILLEGAL	不適切な方法で使用された。 <例> 既に通信されている状態で、再度通信開始しようとした。	API を適切な方法で使用してください。
ERR_PROCESSING	処理を実行できなかつた。 <例> 同様の処理を他のスレッドで実行中のため、処理が実行できなかつた。	処理のタイミングが重なっています。エラーが起きた API を実行しなおしてください。
ERR_UNSUPPORTED	サポートしていない機種名または言語仕様が指定された。	サポートしていない機種では使用できません。
ERR_NOT_FOUND	指定対象を発見できなかつた。 <例> 指定した接続先のプリンターが存在しない。	接続形式や IP アドレス等が間違っていないか、確認してください。
ERR_IN_USE	指定されたデバイスが使用中。 <例> 他のアプリケーションソフトで使用されている。	他のアプリケーションソフトからの使用を終了してください。

エラーステータス	要因	対処方法
ERR_TYPE_INVALID	デバイスのクラスが違う (Printer、LineDisplay)。	接続しているデバイスの種類を確認し、適切なデバイスクラスで接続してください。
	サポートしていない接続方法が指定されています。	システム構成を確認し、適切な接続方法を指定してください。
ERR_DISCONNECT	デバイスとの切断処理に失敗した。	デバイスとの通信状態を確認してください。
ERR_ALREADY_OPENED	既にコミュニケーションボックスがオープンしている。	コミュニケーションボックスとの通信を終了してください。
ERR_ALREADY_USED	指定したメンバー ID は既に使用されている。	異なるメンバー ID を指定してください。
ERR_BOX_COUNT_OVER	コミュニケーションボックスを作成可能な最大数を超えている。	不要なコミュニケーションボックスを削除してください。
ERR_BOX_CLIENT_OVER	コミュニケーションボックスに所属可能な最大数を超えている。	コミュニケーションボックスを使用していないメンバーを除名してください。
ERR_FAILURE	その他のエラーが発生した。	実行環境に問題がないか確認してください。

## Callback Code と対処方法

コード	要因	対処方法
CODE_SUCCESS	処理に成功した。	-
CODE_ERR_TIMEOUT	処理の実行時間が、指定したタイムアウト時間を超過した。	タイムアウト時間を調整してください。
CODE_ERR_NOT_FOUND	<ul style="list-style-type: none"> <li>接続形式や IP アドレスなどの指定が間違っている。</li> <li>指定したデバイスが接続されていない。</li> </ul>	<ul style="list-style-type: none"> <li>接続形式や IP アドレスなどの指定値を確認してください。</li> <li>デバイスの接続状況を確認してください。</li> </ul>
CODE_ERR_AUTORECOVER	<ul style="list-style-type: none"> <li>ヘッド高温エラーが発生した。</li> <li>モータードライバー IC 高温エラーが発生した。</li> <li>バッテリー高温エラーが発生した。</li> </ul>	温度が下がってから処理を実行してください。
	ラベル紙の印刷基準が見つからない。	カバーを開閉してください。
CODE_ERR_COVER_OPEN	カバーが開いている。	プリンターのカバーを閉じてください。
CODE_ERR_CUTTER	オートカッターエラーが発生した。	エラー原因を取り除き、プリンターの電源を入れ直してください。
CODE_ERR_MECHANICAL	メカニカルエラーが発生した。	エラー原因を取り除き、プリンターの電源を入れ直してください。
CODE_ERR_EMPTY	用紙なし。	用紙を補充してください。
CODE_ERR_UNRECOVERABLE	復帰不可能エラーが発生した。	プリンターの電源を入れ直してください。 再発する場合は、購入先かサービスセンターに連絡してください。
CODE_ERR_SYSTEM	TM-i フームウェア、TM-DT ソフトウェアに異常が発生した。	TM-i シリーズ /TM-DT シリーズの電源を入れ直し、OS を再起動してください。
CODE_ERR_PORT	オンライン状態で強制送信処理を実行した。	オフライン状態で強制送信処理を実行してください。
CODE_ERR_INVALID_WINDOW	未登録のウィンドウを指定した。	ウィンドウの指定先を確認してください。
CODE_ERR_JOB_NOT_FOUND	存在しない印刷ジョブ ID が指定された。	印刷ジョブ ID の指定先を確認してください。
CODE_PRINTING	印刷中の印刷ジョブ ID を指定して、 <a href="#">requestPrintJobStatus</a> が実行された。	印刷完了後に処理を実行してください。
CODE_ERR_SPOOLER	スプーラー容量を超えて、印刷データが送信された。	プリンターとの通信が切れていないか、接続状態を確認してください。
CODE_ERR_BATTERY_LOW	バッテリー残量がなくなった。	バッテリーを交換するか、AC アダプターを接続してください。

コード	要因	対処方法
CODE_ERR_FAILURE	その他のエラーが発生した。	実行環境に問題がないか確認してください。
CODE_ERR_TOO_MANY_REQUESTS	プリンターファームウェアの許容量を超えて、印刷データもしくはディスプレイ表示データが送信された。	プリンターの処理がある程度進むまで時間を空けてから、再度データを送信してください。
CODE_ERR_REQUEST_ENTITY_TOO_LARGE	プリンターファームウェアの許容量を超えるデータサイズの印刷ジョブが送信された。	印刷ジョブの内容を確認し、データサイズを下げて再度送信してください。
CODE_ERR_OPOS_CODE	デバイスベンダー定義のエラーが発生した。	OPOS ドライバーのマニュアルで、ResultCodeExtended プロパティーを参照してください。

## Printer Status と対処方法

プリンターステータス	要因	対処方法
status.connection== Printer.FALSE	<ul style="list-style-type: none"> <li>・プリンターの電源が入っていない。</li> <li>・プリンターと接続されていない。</li> </ul>	プリンターの電源や通信状態を確認してください。
status.online==Printer.FALSE	オフライン状態。	カバーオープンや用紙切れなど、オフラインとなる要因を取り除いてください。
status.online==Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.coverOpen== Printer.TRUE	カバーが開いている。	プリンターのカバーを閉じてください。
status.coverOpen== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.paper== Printer.PAPER_NEAR_END	用紙が残り少ない。	用紙を交換してください。
status.paper== Printer.PAPER_EMPTY	用紙なし。	用紙を補充してください。
status.paper==Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.paperFeed==Printer.TRUE	紙送りボタンによる紙送り中。	紙送り停止後に処理を実行してください。
status.paperFeed== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.panelSwitch== Printer.SWITCH_ON	パネルスイッチ操作中。	操作完了後に処理を実行してください。
status.panelSwitch== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.errorStatus== Printer.MECHANICAL_ERR	メカニカルエラーが発生した。	エラー原因を取り除き、プリンターの電源を入れ直してください。
status.errorStatus== Printer.AUTOCUTTER_ERR	オートカッターエラーが発生した。	エラー原因を取り除き、プリンターの電源を入れ直してください。
status.errorStatus== Printer.UNRECOVER_ERR	復帰不可能エラーが発生した。	プリンターの電源を入れ直してください。 再発する場合は、購入先かサービスセンターに連絡してください。

プリンターステータス	要因	対処方法
status.errorStatus== Printer.AUTORECOVER_ERR	• ヘッド高温エラーが発生した。 • モータードライバー IC 高温エラーが発生した。 • バッテリー高温エラーが発生した。	温度が下がってから処理を実行してください。
	ラベル紙の印刷基準が見つからない。	カバーを開閉してください。
status.errorStatus== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.autoRecoverError== Printer.HEAD_OVERHEAT	ヘッド高温エラーが発生した。	温度が下がってから処理を実施してください。
status.autoRecoverError== Printer.MOTOR_OVERHEAT	モーターのドライバー IC 高温エラーが発生した。	温度が下がってから処理を実施してください。
status.autoRecoverError== Printer.BATTERY_OVERHEAT	バッテリー高温エラーが発生した。	温度が下がってから処理を実施してください。
status.autoRecoverError== Printer.WRONG_PAPER	ラベル紙の印刷基準が見つからない。	カバーを開閉してください。
status.autoRecoverError== Printer.COVER_OPEN	カバーが開いている。	プリンターのカバーを閉じてください。
status.autoRecoverError== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.buzzer==Printer.TRUE	ブザー鳴動中。	各プリンターのブザー鳴動条件を確認し、ブザー鳴動原因を取り除いてから処理を実行してください。
status.buzzer== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.adapter== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。
status.batteryLevel== Printer.BATTERY_LEVEL_1	バッテリー残量少。	バッテリーを充電するか、AC アダプターに接続してください。
status.batteryLevel== Printer.BATTERY_LEVEL_0	バッテリー残量なし。	バッテリーを交換するか、AC アダプターに接続してください。
status.batteryLevel== Printer.UNKNOWN	プリンターと接続されていない。	プリンターとの通信状態を確認してください。

# Key Code一覧

定数	コード	定数	コード
Keyboard.VK_BACK	0x08	Keyboard.VK_L	0x4C
Keyboard.VK_TAB	0x09	Keyboard.VK_M	0x4D
Keyboard.VK_RETURN	0x0D	Keyboard.VK_N	0x4E
Keyboard.VK_SHIFT	0x10	Keyboard.VK_O	0x4F
Keyboard.VK_CONTROL	0x11	Keyboard.VK_P	0x50
Keyboard.VK_MENU	0x12	Keyboard.VK_Q	0x51
Keyboard.VK_CAPITAL	0x14	Keyboard.VK_R	0x52
Keyboard.VK_ESCAPE	0x1B	Keyboard.VK_S	0x53
Keyboard.VK_CONVERT	0x1C	Keyboard.VK_T	0x54
Keyboard.VK_NONCONVERT	0x1D	Keyboard.VK_U	0x55
Keyboard.VK_SPACE	0x20	Keyboard.VK_V	0x56
Keyboard.VK_PRIOR	0x21	Keyboard.VK_W	0x57
Keyboard.VK_NEXT	0x22	Keyboard.VK_X	0x58
Keyboard.VK_END	0x23	Keyboard.VK_Y	0x59
Keyboard.VK_HOME	0x24	Keyboard.VK_Z	0x5A
Keyboard.VK_LEFT	0x25	Keyboard.VK_MULTIPLY	0x6A
Keyboard.VK_UP	0x26	Keyboard.VK_ADD	0x6B
Keyboard.VK_RIGHT	0x27	Keyboard.VK_SUBTRACT	0x6D
Keyboard.VK_DOWN	0x28	Keyboard.VK_F1	0x70
Keyboard.VK_INSERT	0x2D	Keyboard.VK_F2	0x71
Keyboard.VK_DELETE	0x2E	Keyboard.VK_F3	0x72
Keyboard.VK_0	0x30	Keyboard.VK_F4	0x73
Keyboard.VK_1	0x31	Keyboard.VK_F5	0x74
Keyboard.VK_2	0x32	Keyboard.VK_F6	0x75
Keyboard.VK_3	0x33	Keyboard.VK_F7	0x76
Keyboard.VK_4	0x34	Keyboard.VK_F8	0x77
Keyboard.VK_5	0x35	Keyboard.VK_F9	0x78
Keyboard.VK_6	0x36	Keyboard.VK_F10	0x79
Keyboard.VK_7	0x37	Keyboard.VK_F11	0x7A
Keyboard.VK_8	0x38	Keyboard.VK_F12	0x7B
Keyboard.VK_9	0x39	Keyboard.VK_OEM_1	0xBA
Keyboard.VK_A	0x41	Keyboard.VK_OEM_PLUS	0xBB
Keyboard.VK_B	0x42	Keyboard.VK_OEM_COMMA	0xBC

定数	コード	定数	コード
Keyboard.VK_C	0x43	Keyboard.VK_OEM_MINUS	0xBD
Keyboard.VK_D	0x44	Keyboard.VK_OEM_PERIOD	0xBE
Keyboard.VK_E	0x45	Keyboard.VK_OEM_2	0xBF
Keyboard.VK_F	0x46	Keyboard.VK_OEM_3	0xC0
Keyboard.VK_G	0x47	Keyboard.VK_OEM_4	0xDB
Keyboard.VK_H	0x48	Keyboard.VK_OEM_5	0xDC
Keyboard.VK_I	0x49	Keyboard.VK_OEM_6	0xDD
Keyboard.VK_J	0x4A	Keyboard.VK_OEM_7	0xDE
Keyboard.VK_K	0x4B	Keyboard.VK_OEM_ATTN	0xF0

# 機器仕様

プリンターや周辺機器などの機器側に依存する API 使用制限や、パラメーターの設定値についての情報を掲載します。



機器ごとの詳細仕様は、各プリンターの詳細取扱説明書を参照してください。

## サポートクラス一覧

プリンターごとの対応クラスは、以下の通りです。

	Printer クラス *1	LineDisplay クラス *1	Keyboard クラス	POSKeyboard クラス	BarcodeScanner クラス	MSR クラス	SimpleSerial クラス	CashChanger クラス	CAT クラス	OtherPeripheral クラス	CommBox クラス	クラス共通 API	Discovery クラス	Epos2Exception クラス	Log クラス	EasySelect クラス *1	EasySelectInfo クラス
TM-m10	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-m30	✓	✓	-	-	✓*3	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T20	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T20II	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T20III	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T70	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T70II	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T88V	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T88VI	✓	✓	-	-	✓*4	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T90II	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-P20	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-P60II	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-P80	✓	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	✓	✓	✓
TM-T70-i	✓	✓	✓	-	✓*5	✓*5	✓	✓*5	-	-	✓	✓	✓	✓	✓	-	-
TM-T88V-i	✓	✓	✓	-	✓*5	✓*5	✓	✓*5	-	-	✓	✓	✓	✓	✓	-	-
TM-T70II-DT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	-
TM-T70II-DT2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	-
TM-T88V-DT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	-
TM-T88VI-DT2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	-
TM-T88IV*2	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	Printer クラス <sup>*1</sup>	LineDisplay クラス <sup>*1</sup>	Keyboard クラス	POSKeyboard クラス	BarcodeScanner クラス	MSR クラス	SimpleSerial クラス	CashChanger クラス	CAT クラス	OtherPeripheral クラス	CommBox クラス	クラス共通 API	Discovery クラス	Epos2Exception クラス	Log クラス	EasySelect クラス <sup>*1</sup>	EasySelectInfo クラス
TM-T90 <sup>*2</sup>	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
TM-T90KP <sup>*2</sup>	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
TM-L90 <sup>*2</sup>	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	

\*1: プリンターごとに API の使用可否や、使用制限があります。[サポート API 一覧](#)を参照してください。

\*2: 子機プリンターでの制御にのみ対応しています。

\*3: TM-m30 フームウェア Ver.1.40ESC/POS 以降で対応しています。

\*4: TM-T88VI フームウェア Ver.40.50ESC/POS 以降で対応しています。

\*5: デバイス制御プログラムでの制御には対応していません。

# 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。

表内記号の意味は、以下の通りです。

- ✓ : 対応している。
- ✓\* : 対応しているが、指定できるパラメーター設定値に制限がある。
- : 対応していない。

「✓\*」の制限については、プリンター別サポート情報を参照してください。

## Printer クラス

### 単機能モデル / モバイルモデル

API	TM-m10	TM-m30	TM-T20	TM-T20II	TM-T20III	TM-T70	TM-T70II	TM-T88V	TM-T88VI	TM-T90II	TM-P20	TM-P60II (Peeler)	TM-P80
Printer	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*
connect	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
disconnect	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
startMonitor	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
stopMonitor	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
getStatus	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
sendData	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
beginTransaction	✓	✓	-	-	-	-	-	-	✓	-	-	-	-
endTransaction	✓	✓	-	-	-	-	-	-	✓	-	-	-	-
requestPrintJobStatus	-	-	-	-	-	-	-	-	-	-	-	-	-
clearCommandBuffer	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addTextAlign	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addLineSpace	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addTextRotate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addText	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addTextLang	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addTextFont	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*
addTextSmooth	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addTextSize	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

API	TM-m10	TM-m30	TM-T20	TM-T20II	TM-T20III	TM-T70	TM-T70II	TM-T88V	TM-T88VI	TM-T90II	TM-P20	TM-P60II (Peeler)	TM-P80
addTextStyle	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*
addHPosition	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addFeedUnit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addFeedLine	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addImage	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*
addLogo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addBarcode	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓	✓*	✓*
addSymbol	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*
addHLine	-	-	-	-	-	-	-	-	-	-	-	-	-
addVLineBegin	-	-	-	-	-	-	-	-	-	-	-	-	-
addVLineEnd	-	-	-	-	-	-	-	-	-	-	-	-	-
addPageBegin	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addPageEnd	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addPageArea	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addPageDirection	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addPagePosition	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
addPageLine	✓*	✓*	-	-	-	-	-	-	✓*	-	✓*	✓*	✓*
addPageRectangle	✓*	✓*	-	-	-	-	-	-	✓*	-	✓*	✓*	✓*
addCut	✓*	✓	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*	✓*
addPulse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	-
addSound	✓*	✓*	✓*	✓*	✓*	✓*	-	✓*	✓*	✓*	-	✓*	✓*
addFeedPosition	-	-	-	-	-	-	-	-	-	-	✓*	✓	✓*
addLayout	-	-	-	-	-	-	-	-	-	-	✓*	✓	✓*
addCommand	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
getMaintenanceCounter	-	✓	-	✓	-	-	-	-	✓	-	✓*	-	-
resetMaintenanceCounter	-	✓	-	✓	-	-	-	-	✓	-	✓*	-	-
getPrinterSetting	-	✓*	-	✓*	-	-	-	-	✓*	-	✓*	-	-
setPrinterSetting	-	✓*	-	✓*	-	-	-	-	✓*	-	✓*	-	-
forceRecover	-	-	-	-	-	-	-	-	-	-	-	-	-
forcePulse	-	-	-	-	-	-	-	-	-	-	-	-	-
forceStopSound	-	-	-	-	-	-	-	-	-	-	-	-	-

API	TM-m10	TM-m30	TM-T20	TM-T20II	TM-T20III	TM-T70	TM-T70II	TM-T88V	TM-T88VI	TM-T90II	TM-P20	TM-P60II (Peeler)	TM-P80
forceCommand	-	-	-	-	-	-	-	-	-	-	-	-	-
forceReset	-	-	-	-	-	-	-	-	-	-	-	-	-
setStatusChangeEventLister	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
setReceiveEventLister	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
interval	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

## TM-iシリーズ / TM-DT シリーズ

API	TM-T70-I	TM-T88V-i	TM-T70II-DT	TM-T70II-DT2	TM-T88V-DT	TM-T88VI-DT2
Printer	✓*	✓*	✓*	✓*	✓*	✓*
connect	✓	✓	✓	✓	✓	✓
disconnect	✓	✓	✓	✓	✓	✓
startMonitor	✓	✓	✓	✓	✓	✓
stopMonitor	✓	✓	✓	✓	✓	✓
getStatus	✓	✓	✓	✓	✓	✓
sendData	✓	✓	✓	✓	✓	✓
beginTransaction	-	-	-	-	-	-
endTransaction	-	-	-	-	-	-
requestPrintJobStatus	✓	✓	✓	✓	✓	✓
clearCommandBuffer	✓	✓	✓	✓	✓	✓
addTextAlign	✓	✓	✓	✓	✓	✓
addLineSpace	✓	✓	✓	✓	✓	✓
addTextRotate	✓	✓	✓	✓	✓	✓
addText	✓	✓	✓	✓	✓	✓
addTextLang	✓	✓	✓	✓	✓	✓
addTextFont	✓*	✓*	✓*	✓*	✓*	✓*
addTextSmooth	✓	✓	✓	✓	✓	✓
addTextSize	✓	✓	✓	✓	✓	✓

API	TM-T70-i	TM-T88V-i	TM-T70II-DT	TM-T70II-DT2	TM-T88V-DT	TM-T88VI-DT2
addTextStyle	✓*	✓*	✓*	✓*	✓*	✓*
addHPosition	✓	✓	✓	✓	✓	✓
addFeedUnit	✓	✓	✓	✓	✓	✓
addFeedLine	✓	✓	✓	✓	✓	✓
addImage	✓*	✓*	✓*	✓*	✓*	✓*
addLogo	✓	✓	✓	✓	✓	✓
addBarcode	✓*	✓*	✓*	✓*	✓*	✓*
addSymbol	✓*	✓*	✓*	✓*	✓*	✓*
addHLine	-	-	-	-	-	-
addVLineBegin	-	-	-	-	-	-
addVLineEnd	-	-	-	-	-	-
addPageBegin	✓	✓	✓	✓	✓	✓
addPageEnd	✓	✓	✓	✓	✓	✓
addPageArea	✓	✓	✓	✓	✓	✓
addPageDirection	✓	✓	✓	✓	✓	✓
addPagePosition	✓	✓	✓	✓	✓	✓
addPageLine	-	-	-	-	-	-
addPageRectangle	-	-	-	-	-	-
addCut	✓*	✓*	✓*	✓*	✓*	✓*
addPulse	✓	✓	✓	✓	✓	✓
addSound	-	✓*	✓*	✓*	✓*	✓*
addFeedPosition	-	-	-	-	-	-
addLayout	-	-	-	-	-	-
addCommand	✓	✓	✓	✓	✓	✓
getMaintenanceCounter	-	-	-	-	-	-
resetMaintenanceCounter	-	-	-	-	-	-
getPrinterSetting	-	-	-	-	-	-
setPrinterSetting	-	-	-	-	-	-
forceRecover	✓	✓	✓	✓	✓	✓
forcePulse	✓	✓	✓	✓	✓	✓
forceStopSound	-	✓	✓	✓	✓	✓

API	TM-T70-i	TM-T88V-i	TM-T70II-DT	TM-T70II-DT2	TM-T88V-DT	TM-T88VI-DT2
forceCommand	✓	✓	✓	✓	✓	✓
forceReset	✓	✓	✓	✓	✓	✓
setStatusChangeEventListerner	✓	✓	✓	✓	✓	✓
setReceiveEventListerner	✓	✓	✓	✓	✓	✓
interval	✓	✓	✓	✓	✓	✓

## 子機プリンター

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
Printer	✓*	✓*	✓*	✓*
connect	✓	✓	✓	✓
disconnect	✓	✓	✓	✓
startMonitor	✓	✓	✓	✓
stopMonitor	✓	✓	✓	✓
getStatus	✓	✓	✓	✓
sendData	✓	✓	✓	✓
beginTransaction	-	-	-	-
endTransaction	-	-	-	-
requestPrintJobStatus	-	-	-	-
clearCommandBuffer	✓	✓	✓	✓
addTextAlign	✓	✓	✓	✓
addLineSpace	✓	✓	✓	✓
addTextRotate	✓	✓	✓	✓
addText	✓	✓	✓	✓
addTextLang	✓	✓	✓	✓
addTextFont	✓*	✓*	✓*	✓*
addTextSmooth	✓	✓	✓	✓
addTextSize	✓	✓	✓	✓
addTextStyle	✓*	✓*	✓*	✓*
addHPosition	✓	✓	✓	✓

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
addFeedUnit	✓	✓	✓	✓
addFeedLine	✓	✓	✓	✓
addImage	✓*	✓*	✓*	✓*
addLogo	✓	✓	✓	✓
addBarcode	✓*	✓*	✓*	✓*
addSymbol	✓*	✓*	✓*	✓*
addHLine	-	-	✓	-
addVLineBegin	-	-	✓	-
addVLineEnd	-	-	✓	-
addPageBegin	✓	✓	✓	✓
addPageEnd	✓	✓	✓	✓
addPageArea	✓	✓	✓	✓
addPageDirection	✓	✓	✓	✓
addPagePosition	✓	✓	✓	✓
addPageLine	-	-	✓	-
addPageRectangle	-	-	✓	-
addCut	✓*	✓*	✓	✓*
addPulse	✓	✓	✓	✓
addSound	-	-	✓*	-
addFeedPosition	-	-	-	✓
addLayout	-	-	-	-
addCommand	✓	✓	✓	✓
getMaintenanceCounter	-	-	-	-
resetMaintenanceCounter	-	-	-	-
getPrinterSetting	-	-	-	-
setPrinterSetting	-	-	-	-
forceRecover	-	-	-	-
forcePulse	-	-	-	-
forceStopSound	-	-	-	-
forceCommand	-	-	-	-
forceReset	-	-	-	-
setStatusChangeEventListener	✓	✓	✓	✓
setReceiveEventListener	✓	✓	✓	✓

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
interval	✓	✓	✓	✓

## LineDisplay クラス

単機能モデル / モバイルモデル

API	TM-m10	TM-m30	TM-T20	TM-T20II	TM-T20III	TM-T70	TM-T70II	TM-T88V	TM-T88VI	TM-T90II	TM-P20	TM-P60II (Peeler)	TM-P80
LineDisplay	-	✓*	-	-	-	-	-	-	✓	-	-	-	-
connect	-	✓	-	-	-	-	-	-	✓	-	-	-	-
disconnect	-	✓	-	-	-	-	-	-	✓	-	-	-	-
getStatus	-	✓	-	-	-	-	-	-	✓	-	-	-	-
sendData	-	✓	-	-	-	-	-	-	✓	-	-	-	-
clearCommandBuffer	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addInitialize	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addCreateWindow	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addDestroyWindow	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addSetCurrentWindow	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addClearCurrentWindow	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addSetCursorPosition	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addMoveCursorPosition	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addSetCursorType	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addText	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addReverseText	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addMarqueeText	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addSetBlink	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addSetBrightness	-	✓	-	-	-	-	-	-	✓	-	-	-	-
addShowClock	-	-	-	-	-	-	-	-	✓	-	-	-	-
addCommand	-	✓	-	-	-	-	-	-	✓	-	-	-	-
setReceiveEventListener	-	✓	-	-	-	-	-	-	✓	-	-	-	-

## TM-iシリーズ / TM-DTシリーズ

API	TM-T70-I	TM-T88VI	TM-T70II-DT	TM-T70II-DT2	TM-T88V-DT	TM-T88VI-DT2
LineDisplay	✓*	✓*	✓	✓	✓	✓
connect	✓	✓	✓	✓	✓	✓
disconnect	✓	✓	✓	✓	✓	✓
getStatus	✓	✓	✓	✓	✓	✓
sendData	✓	✓	✓	✓	✓	✓
clearCommandBuffer	✓	✓	✓	✓	✓	✓
addInitialize	✓	✓	✓	✓	✓	✓
addCreateWindow	✓	✓	✓	✓	✓	✓
addDestroyWindow	✓	✓	✓	✓	✓	✓
addSetCurrentWindow	✓	✓	✓	✓	✓	✓
addClearCurrentWindow	✓	✓	✓	✓	✓	✓
addSetCursorPosition	✓	✓	✓	✓	✓	✓
addMoveCursorPosition	✓	✓	✓	✓	✓	✓
addSetCursorType	✓	✓	✓	✓	✓	✓
addText	✓	✓	✓	✓	✓	✓
addReverseText	✓	✓	✓	✓	✓	✓
addMarqueeText	✓	✓	✓	✓	✓	✓
addSetBlink	✓	✓	✓	✓	✓	✓
addSetBrightness	✓	✓	✓	✓	✓	✓
addShowClock	✓	✓	✓	✓	✓	✓
addCommand	✓	✓	✓	✓	✓	✓
setReceiveEventListerner	✓	✓	✓	✓	✓	✓

## 子機プリンター

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
LineDisplay	-	-	-	-
connect	-	-	-	-
disconnect	-	-	-	-

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
getStatus	-	-	-	-
sendData	-	-	-	-
clearCommandBuffer	-	-	-	-
addInitialize	-	-	-	-
addCreateWindow	-	-	-	-
addDestroyWindow	-	-	-	-
addSetCurrentWindow	-	-	-	-
addClearCurrentWindow	-	-	-	-
addSetCursorPosition	-	-	-	-
addMoveCursorPosition	-	-	-	-
addSetCursorType	-	-	-	-
addText	-	-	-	-
addReverseText	-	-	-	-
addMarqueeText	-	-	-	-
addSetBlink	-	-	-	-
addSetBrightness	-	-	-	-
addShowClock	-	-	-	-
addCommand	-	-	-	-
setReceiveEventListener	-	-	-	-

## クラス共通 API

単機能モデル / モバイルモデル

API	TM-m10	TM-m30	TM-T20	TM-T20II	TM-T20III	TM-T70II	TM-T88V	TM-T88VI	TM-T90II	TM-P20	TM-P60II (Peerer)	TM-P80
getAdmin	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
getLocation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
setConnectionEventListener	-	-	-	-	-	-	-	-	-	-	-	-

TM-i シリーズ / TM-DT シリーズ

API	TM-T70-i	TM-T88V-i	TM-T70II-DT	TM-T70II-DT2	TM-T88V-DT	TM-T88VI-DT2
getAdmin	✓	✓	✓	✓	✓	✓
getLocation	✓	✓	✓	✓	✓	✓
setConnectionEventListener	✓	✓	✓	✓	✓	✓

子機プリンター

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
getAdmin	-	-	-	-
getLocation	-	-	-	-
setConnectionEventListener	-	-	-	-

## EasySelect クラス

単機能モデル / モバイルモデル

API	TM-m10	TM-m30	TM-T20	TM-T20II	TM-T20III	TM-T70II	TM-T88V	TM-T88VI	TM-T90II	TM-P20	TM-P60II (Peeler)	TM-P80
parseNFC	✓	✓	-	-	-	-	-	✓	-	✓	✓	✓
parseQR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
createQR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

TM-i シリーズ / TM-DT シリーズ

API	TM-T70-i	TM-T88V-i	TM-T70II-DT	TM-T70II-DT2	TM-T88V-DT	TM-T88VI-DT2
parseNFC	-	-	-	-	-	-
parseQR	-	-	-	-	-	-
createQR	-	-	-	-	-	-

子機プリンター

API	TM-T88IV	TM-T90	TM-T90KP	TM-L90
parseNFC	-	-	-	-
parseQR	-	-	-	-
createQR	-	-	-	-

# プリンター別サポート情報

## TM-m10

### プリンター仕様

項目	仕様
初期改行量	30 ドット
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
Font A のベースライン	文字の上端から 21 ドット目
Font B のベースライン	文字の上端から 21 ドット目
Font C のベースライン	文字の上端から 16 ドット目

### パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	<ul style="list-style-type: none"> <li>Printer.TM_M10</li> </ul>
addTextFont	font	<ul style="list-style-type: none"> <li>Printer.FONT_A (初期値)</li> <li>Printer.FONT_B</li> <li>Printer.FONT_C</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addTextStyle	color	<ul style="list-style-type: none"> <li>Printer.COLOR_1 (初期値)</li> <li>Printer.PARAM_UNSPECIFIED</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addImage	color	<ul style="list-style-type: none"> <li>Printer.COLOR_1</li> <li>Printer.PARAM_DEFAULT</li> </ul>
	mode	<ul style="list-style-type: none"> <li>Printer.MODE_MONO</li> <li>Printer.MODE_MONO_HIGH_DENSITY</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addBarcode	font	<ul style="list-style-type: none"> <li>Printer.FONT_A (初期値)</li> <li>Printer.FONT_B</li> <li>Printer.FONT_C</li> <li>Printer.PARAM_DEFAULT</li> <li>Printer.PARAM_UNSPECIFIED</li> </ul>

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addPageLine	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addPageRectangle	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
addSound	pattern	<ul style="list-style-type: none"><li>Printer.PATTERN_NONE</li><li>Printer.PATTERN_A</li><li>Printer.PATTERN_B</li><li>Printer.PATTERN_C</li><li>Printer.PATTERN_D</li><li>Printer.PATTERN_E</li><li>Printer.PATTERN_ERROR</li><li>Printer.PATTERN_PAPER_EMPTY</li><li>Printer.PARAM_DEFAULT</li></ul>
	repeat	<ul style="list-style-type: none"><li>1 ~ 255</li><li>Printer.PARAM_DEFAULT</li></ul>

## TM-m30

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
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
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 21 ドット目	
Font C のベースライン	文字の上端から 16 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	<ul style="list-style-type: none"> <li>Printer.TM_M30</li> </ul>
addTextFont	font	<ul style="list-style-type: none"> <li>Printer.FONT_A (初期値)</li> <li>Printer.FONT_B</li> <li>Printer.FONT_C</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addTextStyle	color	<ul style="list-style-type: none"> <li>Printer.COLOR_1 (初期値)</li> <li>Printer.PARAM_UNSPECIFIED</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addImage	color	<ul style="list-style-type: none"> <li>Printer.COLOR_1</li> <li>Printer.PARAM_DEFAULT</li> </ul>
	mode	<ul style="list-style-type: none"> <li>Printer.MODE_MONO</li> <li>Printer.MODE_MONO_HIGH_DENSITY</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addBarcode	font	<ul style="list-style-type: none"> <li>Printer.FONT_A (初期値)</li> <li>Printer.FONT_B</li> <li>Printer.FONT_C</li> <li>Printer.PARAM_DEFAULT</li> <li>Printer.PARAM_UNSPECIFIED</li> </ul>

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_ STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_1 2</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_1 6</li> </ul>
addPageLine	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addPageRectangle	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
setPrinterSetting	value	<ul style="list-style-type: none"> <li>• Printer.Setting.PaperWidth58_0</li> <li>• Printer.Setting.PaperWidth80_0</li> <li>• Printer.Setting.PrintDensity70</li> <li>• Printer.Setting.PrintDensity75</li> <li>• Printer.Setting.PrintDensity80</li> <li>• Printer.Setting.PrintDensity85</li> <li>• Printer.Setting.PrintDensity90</li> <li>• Printer.Setting.PrintDensity95</li> <li>• Printer.Setting.PrintDensity100</li> <li>• Printer.Setting.PrintDensity105</li> <li>• Printer.Setting.PrintDensity110</li> <li>• Printer.Setting.PrintDensity115</li> <li>• Printer.Setting.PrintDensity120</li> <li>• Printer.Setting.PrintDensity125</li> <li>• Printer.Setting.PrintDensity130</li> <li>• Printer.Setting.PrintSpeed1</li> <li>• Printer.Setting.PrintSpeed2</li> <li>• Printer.Setting.PrintSpeed3</li> <li>• Printer.Setting.PrintSpeed4</li> <li>• Printer.Setting.PrintSpeed5</li> <li>• Printer.Setting.PrintSpeed6</li> <li>• Printer.Setting.PrintSpeed7</li> <li>• Printer.Setting.PrintSpeed8</li> <li>• Printer.Setting.PrintSpeed9</li> <li>• Printer.Setting.PrintSpeed10</li> </ul>
Linedisplay	displaySeries	<ul style="list-style-type: none"> <li>• Linedisplay.DM_D30</li> </ul>

## TM-T20

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	420 x 831 ドット	576 x 831 ドット
ページモード最大領域	420 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 16 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T20
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-T20II

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	420 x 831 ドット	576 x 831 ドット
ページモード最大領域	420 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 16 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T20
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
getPrinterSetting	type	<ul style="list-style-type: none"><li>• Printer.Setting.PrintDensity</li><li>• Printer.Setting.PrintSpeed</li></ul>
	type	<ul style="list-style-type: none"><li>• Printer.Setting.PrintDensity</li><li>• Printer.Setting.PrintSpeed</li></ul>
setPrinterSetting	value	<ul style="list-style-type: none"><li>• Printer.Setting.PrintDensity85</li><li>• Printer.Setting.PrintDensity90</li><li>• Printer.Setting.PrintDensity95</li><li>• Printer.Setting.PrintDensity100</li><li>• Printer.Setting.PrintDensity105</li><li>• Printer.Setting.PrintDensity110</li><li>• Printer.Setting.PrintDensity115</li><li>• Printer.Setting.PrintSpeed1</li><li>• Printer.Setting.PrintSpeed2</li><li>• Printer.Setting.PrintSpeed3</li><li>• Printer.Setting.PrintSpeed4</li><li>• Printer.Setting.PrintSpeed5</li><li>• Printer.Setting.PrintSpeed6</li><li>• Printer.Setting.PrintSpeed7</li><li>• Printer.Setting.PrintSpeed8</li><li>• Printer.Setting.PrintSpeed9</li><li>• Printer.Setting.PrintSpeed10</li><li>• Printer.Setting.PrintSpeed11</li><li>• Printer.Setting.PrintSpeed12</li><li>• Printer.Setting.PrintSpeed13</li></ul>

## TM-T20III

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	通常モード	420 x 1662 ドット
	42 衍モード	378 x 1662 ドット
ページモード最大領域	通常モード	464 x 3324 ドット
	42 衍モード	639 x 3324 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 16 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T20
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_0_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-T70

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	416 x 1662 ドット	576 x 1662 ドット
ページモード最大領域	416 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 15 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T70
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	type	• Printer.BARCODE_UPC_A • Printer.BARCODE_UPC_E • Printer.BARCODE_EAN13 • Printer.BARCODE_JAN13 • Printer.BARCODE_EAN8 • Printer.BARCODE_JAN8 • Printer.BARCODE_CODE39 • Printer.BARCODE_ITF • Printer.BARCODE_CODABAR • Printer.BARCODE_CODE93 • Printer.BARCODE_CODE128
	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"><li>Printer.SYMBOL_QRCODE_MODEL_1</li><li>Printer.SYMBOL_QRCODE_MODEL_2</li></ul>
	level	<ul style="list-style-type: none"><li>Printer.LEVEL_L</li><li>Printer.LEVEL_M</li><li>Printer.LEVEL_Q</li><li>Printer.LEVEL_H</li><li>Printer.PARAM_DEFAULT</li><li>Printer.PARAM_UNSPECIFIED</li></ul>
addCut	type	<ul style="list-style-type: none"><li>Printer.CUT_FEED</li><li>Printer.CUT_NO_FEED</li><li>Printer.PARAM_DEFAULT</li></ul>

## TM-T70II

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	416 x 1662 ドット	576 x 1662 ドット
ページモード最大領域	416 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 15 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T70
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

**TM-T88V****プリンター仕様**

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	360 x 831 ドット	512 x 831 ドット
ページモード最大領域	360 x 1662 ドット	512 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	ANK	文字の上端から 16 ドット目
	漢字	文字の上端から 15 ドット目

**パラメーター制限**

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T88
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

**TM-T88VI****プリンター仕様**

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	360 x 831 ドット	512 x 831 ドット
ページモード最大領域	360 x 1662 ドット	512 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	ANK	文字の上端から 16 ドット目
	漢字	文字の上端から 15 ドット目

**パラメーター制限**

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T88
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addPageLine	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addPageRectangle	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>• 1 ~ 255</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
setPrinterSetting	value	<ul style="list-style-type: none"> <li>• Printer.Setting.PaperWidth58_0</li> <li>• Printer.Setting.PaperWidth80_0</li> <li>• Printer.Setting.PrintDensityDIP</li> <li>• Printer.Setting.PrintDensity70</li> <li>• Printer.Setting.PrintDensity75</li> <li>• Printer.Setting.PrintDensity80</li> <li>• Printer.Setting.PrintDensity85</li> <li>• Printer.Setting.PrintDensity90</li> <li>• Printer.Setting.PrintDensity95</li> <li>• Printer.Setting.PrintDensity100</li> <li>• Printer.Setting.PrintDensity105</li> <li>• Printer.Setting.PrintDensity110</li> <li>• Printer.Setting.PrintDensity115</li> <li>• Printer.Setting.PrintDensity120</li> <li>• Printer.Setting.PrintDensity125</li> <li>• Printer.Setting.PrintDensity130</li> <li>• Printer.Setting.PrintSpeed1</li> <li>• Printer.Setting.PrintSpeed2</li> <li>• Printer.Setting.PrintSpeed3</li> <li>• Printer.Setting.PrintSpeed4</li> <li>• Printer.Setting.PrintSpeed5</li> <li>• Printer.Setting.PrintSpeed6</li> <li>• Printer.Setting.PrintSpeed7</li> <li>• Printer.Setting.PrintSpeed8</li> <li>• Printer.Setting.PrintSpeed9</li> <li>• Printer.Setting.PrintSpeed10</li> <li>• Printer.Setting.PrintSpeed11</li> <li>• Printer.Setting.PrintSpeed12</li> <li>• Printer.Setting.PrintSpeed13</li> <li>• Printer.Setting.PrintSpeed14</li> </ul>

## TM-T90II

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	モノクロ印字	420 x 738 ドット
	2 色印字	420 x 369 ドット
ページモード最大領域	モノクロ印字	420 x 1476 ドット
	2 色印字	420 x 738 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 21 ドット目	
Font C のベースライン	文字の上端から 15 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T90
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.FONT_C • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.COLOR_2 • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.FONT_C • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-P20

## プリンター仕様

項目	仕様
初期改行量	30 ドット
ページモード初期領域	384 x 2400 ドット
ページモード最大領域	384 x 2400 ドット
Font A のベースライン	文字の上端から 21 ドット目
Font B のベースライン	文字の上端から 21 ドット目
Font C のベースライン	文字の上端から 16 ドット目
Font D のベースライン	文字の上端から 21 ドット目
Font E のベースライン	文字の上端から 15 ドット目

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_P20
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addPageLine	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addPageRectangle	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
addSound	pattern	<ul style="list-style-type: none"> <li>Printer.PATTERN_NONE</li> <li>Printer.PATTERN_1</li> <li>Printer.PATTERN_2</li> <li>Printer.PATTERN_3</li> <li>Printer.PATTERN_4</li> <li>Printer.PATTERN_5</li> <li>Printer.PATTERN_6</li> <li>Printer.PATTERN_7</li> <li>Printer.PATTERN_8</li> <li>Printer.PATTERN_9</li> <li>Printer.PATTERN_10</li> <li>Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>1 ~ 255</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addFeedPosition	position	<ul style="list-style-type: none"> <li>Printer.FEED_CUTTING</li> <li>Printer.FEED_CURRENT_TOF</li> <li>Printer.FEED_NEXT_TOF</li> </ul>
addLayout	type	<ul style="list-style-type: none"> <li>Printer.LAYOUT_RECEIPT</li> <li>Printer.LAYOUT_RECEIPT_BM</li> </ul>
	height	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	marginTop	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	marginBottom	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	offsetCut	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	offsetLabel	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
getMaintenanceCounter	type	<ul style="list-style-type: none"> <li>Printer.MaintenanceCounter.PaperFeed</li> </ul>
resetMaintenanceCounter	type	<ul style="list-style-type: none"> <li>Printer.MaintenanceCounter.PaperFeed</li> </ul>

API	パラメーター	指定可能な設定値
getPrinterSetting	type	<ul style="list-style-type: none"> <li>• Printer.Setting.PrintDensity</li> <li>• Printer.Setting.PrintSpeed</li> </ul>
	type	<ul style="list-style-type: none"> <li>• Printer.Setting.PrintDensity</li> <li>• Printer.Setting.PrintSpeed</li> </ul>
setPrinterSetting	value	<ul style="list-style-type: none"> <li>• Printer.Setting.PrintDensity70</li> <li>• Printer.Setting.PrintDensity75</li> <li>• Printer.Setting.PrintDensity80</li> <li>• Printer.Setting.PrintDensity85</li> <li>• Printer.Setting.PrintDensity90</li> <li>• Printer.Setting.PrintDensity95</li> <li>• Printer.Setting.PrintDensity100</li> <li>• Printer.Setting.PrintDensity105</li> <li>• Printer.Setting.PrintDensity110</li> <li>• Printer.Setting.PrintDensity115</li> <li>• Printer.Setting.PrintDensity120</li> <li>• Printer.Setting.PrintDensity125</li> <li>• Printer.Setting.PrintDensity130</li> <li>• Printer.Setting.PrintSpeed1</li> <li>• Printer.Setting.PrintSpeed2</li> <li>• Printer.Setting.PrintSpeed3</li> <li>• Printer.Setting.PrintSpeed4</li> <li>• Printer.Setting.PrintSpeed5</li> <li>• Printer.Setting.PrintSpeed6</li> <li>• Printer.Setting.PrintSpeed7</li> <li>• Printer.Setting.PrintSpeed8</li> <li>• Printer.Setting.PrintSpeed9</li> <li>• Printer.Setting.PrintSpeed10</li> </ul>

## TM-P60II (Peeler)

### プリンター仕様

項目	レシート仕様	ダイカットラベル仕様
初期改行量	30 ドット	
ページモード初期領域	432 x 1624 ドット	400 x 1624 ドット
ページモード最大領域	432 x 1624 ドット	400 x 1624 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 16 ドット目	
Font C のベースライン	文字の上端から 15 ドット目	

### パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	<ul style="list-style-type: none"> <li>• Printer.TM_P60II</li> </ul>
addTextFont	font	<ul style="list-style-type: none"> <li>• Printer.FONT_A (初期値)</li> <li>• Printer.FONT_B</li> <li>• Printer.FONT_C</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addTextStyle	color	<ul style="list-style-type: none"> <li>• Printer.COLOR_1 (初期値)</li> <li>• Printer.PARAM_UNSPECIFIED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addImage	color	<ul style="list-style-type: none"> <li>• Printer.COLOR_1</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
	mode	<ul style="list-style-type: none"> <li>• Printer.MODE_MONO</li> <li>• Printer.MODE_MONO_HIGH_DENSITY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addBarcode	font	<ul style="list-style-type: none"> <li>• Printer.FONT_A (初期値)</li> <li>• Printer.FONT_B</li> <li>• Printer.FONT_C</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addPageLine	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addPageRectangle	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
addSound	pattern	<ul style="list-style-type: none"><li>• Printer.PATTERN_NONE</li><li>• Printer.PATTERN_1</li><li>• Printer.PATTERN_2</li><li>• Printer.PATTERN_3</li><li>• Printer.PATTERN_4</li><li>• Printer.PATTERN_5</li><li>• Printer.PATTERN_6</li><li>• Printer.PATTERN_7</li><li>• Printer.PATTERN_8</li><li>• Printer.PATTERN_9</li><li>• Printer.PATTERN_10</li><li>• Printer.PARAM_DEFAULT</li></ul>
	repeat	<ul style="list-style-type: none"><li>• 1 ~ 255</li><li>• Printer.PARAM_DEFAULT</li></ul>

## TM-P80

## プリンター仕様

項目	仕様
初期改行量	30 ドット
ページモード初期領域	576 x 1662 ドット
ページモード最大領域	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目
Font B のベースライン	文字の上端から 21 ドット目
Font C のベースライン	文字の上端から 15 ドット目

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_P80
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.FONT_C • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.FONT_C • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addPageLine	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addPageRectangle	lineStyle	<ul style="list-style-type: none"> <li>• Printer.LINE_THIN</li> <li>• Printer.LINE_MEDIUM</li> <li>• Printer.LINE_THICK</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
addSound	pattern	<ul style="list-style-type: none"> <li>Printer.PATTERN_NONE</li> <li>Printer.PATTERN_1</li> <li>Printer.PATTERN_2</li> <li>Printer.PATTERN_3</li> <li>Printer.PATTERN_4</li> <li>Printer.PATTERN_5</li> <li>Printer.PATTERN_6</li> <li>Printer.PATTERN_7</li> <li>Printer.PATTERN_8</li> <li>Printer.PATTERN_9</li> <li>Printer.PATTERN_10</li> <li>Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>1 ~ 255</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addFeedPosition	position	<ul style="list-style-type: none"> <li>Printer.FEED_CUTTING</li> <li>Printer.FEED_CURRENT_TOF</li> <li>Printer.FEED_NEXT_TOF</li> </ul>
addLayout	type	<ul style="list-style-type: none"> <li>Printer.LAYOUT_RECEIPT</li> <li>Printer.LAYOUT_RECEIPT_BM</li> </ul>
	height	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	marginTop	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	marginBottom	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	offsetCut	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>
	offsetLabel	<ul style="list-style-type: none"> <li>レシート紙 (ブラックマークなし)</li> <li>レシート紙 (ブラックマークあり)</li> </ul>

## TM-T70-i

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	416 x 1662 ドット	576 x 1662 ドット
ページモード最大領域	416 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 15 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T70
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
	mode	• Printer.MODE_MONO • Printer.MODE_MONO_HIGH_DENSITY • Printer.PARAM_DEFAULT
addBarcode	type	• Printer.BARCODE_UPC_A • Printer.BARCODE_UPC_E • Printer.BARCODE_EAN13 • Printer.BARCODE_JAN13 • Printer.BARCODE_EAN8 • Printer.BARCODE_JAN8 • Printer.BARCODE_CODE39 • Printer.BARCODE_ITF • Printer.BARCODE_CODABAR • Printer.BARCODE_CODE93 • Printer.BARCODE_CODE128
	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"><li>Printer.SYMBOL_QRCODE_MODEL_1</li><li>Printer.SYMBOL_QRCODE_MODEL_2</li></ul>
	level	<ul style="list-style-type: none"><li>Printer.LEVEL_L</li><li>Printer.LEVEL_M</li><li>Printer.LEVEL_Q</li><li>Printer.LEVEL_H</li><li>Printer.PARAM_DEFAULT</li><li>Printer.PARAM_UNSPECIFIED</li></ul>
addCut	type	<ul style="list-style-type: none"><li>Printer.CUT_FEED</li><li>Printer.CUT_NO_FEED</li><li>Printer.PARAM_DEFAULT</li></ul>
LineDisplay	displaySeries	<ul style="list-style-type: none"><li>LineDisplay.DM_D110</li></ul>

## TM-T88V-i

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	360 x 831 ドット	512 x 831 ドット
ページモード最大領域	360 x 1662 ドット	512 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	ANK	文字の上端から 16 ドット目
	漢字	文字の上端から 15 ドット目

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T88
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
LineDisplay	displaySeries	<ul style="list-style-type: none"> <li>• LineDisplay.DM_D110</li> </ul>

**TM-T70II-DT****プリンター仕様**

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	416 x 1662 ドット	576 x 1662 ドット
ページモード最大領域	416 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 15 ドット目	

**パラメーター制限**

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T70
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

**TM-T70II-DT2****プリンター仕様**

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	416 x 1662 ドット	576 x 1662 ドット
ページモード最大領域	416 x 1662 ドット	576 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 15 ドット目	

**パラメーター制限**

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T70
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-T88V-DT

### プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	360 x 831 ドット	512 x 831 ドット
ページモード最大領域	360 x 1662 ドット	512 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	ANK	文字の上端から 16 ドット目
	漢字	文字の上端から 15 ドット目

### パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T88
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_O_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-T88VI-DT2

### プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	360 x 831 ドット	512 x 831 ドット
ページモード最大領域	360 x 1662 ドット	512 x 1662 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	ANK	文字の上端から 16 ドット目
	漢字	文字の上端から 15 ドット目

### パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T88
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.PARAM_DEFAULT
addBarcode	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED</li> <li>• Printer.SYMBOL_GS1_DATABAR_STACKED_0_MNIDIRECTIONAL</li> <li>• Printer.SYMBOL_GS1_DATABAR_EXPANDED_STACKED</li> <li>• Printer.SYMBOL_AZTECCODE_FULLRANGE</li> <li>• Printer.SYMBOL_AZTECCODE_COMPACT</li> <li>• Printer.SYMBOL_DATAMATRIX_SQUARE</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_8</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_12</li> <li>• Printer.SYMBOL_DATAMATRIX_RECTANGLE_16</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>• Printer.PATTERN_NONE</li> <li>• Printer.PATTERN_A</li> <li>• Printer.PATTERN_B</li> <li>• Printer.PATTERN_C</li> <li>• Printer.PATTERN_D</li> <li>• Printer.PATTERN_E</li> <li>• Printer.PATTERN_ERROR</li> <li>• Printer.PATTERN_PAPER_EMPTY</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-T88IV

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	モノクロ印字	360 x 831 ドット
	2色印字	360 x 415 ドット
ページモード最大領域	モノクロ印字	360 x 1662 ドット
	2色印字	360 x 831 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 16 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T88
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.COLOR_2 • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.COLOR_2 • Printer.PARAM_DEFAULT
addBarcode	type	• Printer.BARCODE_UPC_A • Printer.BARCODE_UPC_E • Printer.BARCODE_EAN13 • Printer.BARCODE_JAN13 • Printer.BARCODE_EAN8 • Printer.BARCODE_JAN8 • Printer.BARCODE_CODE39 • Printer.BARCODE_ITF • Printer.BARCODE_CODABAR • Printer.BARCODE_CODE93 • Printer.BARCODE_CODE128
	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.PARAM_DEFAULT • Printer.PARAM_UNSPECIFIED

API	パラメーター	指定可能な設定値
addSymbol	type	<ul style="list-style-type: none"><li>Printer.SYMBOL_QRCODE_MODEL_1</li><li>Printer.SYMBOL_QRCODE_MODEL_2</li></ul>
	level	<ul style="list-style-type: none"><li>Printer.LEVEL_L</li><li>Printer.LEVEL_M</li><li>Printer.LEVEL_Q</li><li>Printer.LEVEL_H</li><li>Printer.PARAM_DEFAULT</li><li>Printer.PARAM_UNSPECIFIED</li></ul>
addCut	type	<ul style="list-style-type: none"><li>Printer.CUT_FEED</li><li>Printer.CUT_NO_FEED</li><li>Printer.PARAM_DEFAULT</li></ul>

## TM-T90

## プリンター仕様

項目	58mm 仕様	80mm 仕様
初期改行量	30 ドット	
ページモード初期領域	モノクロ印字	420 x 738 ドット
	2 色印字	420 x 369 ドット
ページモード最大領域	モノクロ印字	420 x 1476 ドット
	2 色印字	420 x 738 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 21 ドット目	
Font C のベースライン	文字の上端から 15 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_T90
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.FONT_C • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.COLOR_2 • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.COLOR_2 • Printer.PARAM_DEFAULT

API	パラメーター	指定可能な設定値
addBarcode	type	<ul style="list-style-type: none"> <li>• Printer.BARCODE_UPC_A</li> <li>• Printer.BARCODE_UPC_E</li> <li>• Printer.BARCODE_EAN13</li> <li>• Printer.BARCODE_JAN13</li> <li>• Printer.BARCODE_EAN8</li> <li>• Printer.BARCODE_JAN8</li> <li>• Printer.BARCODE_CODE39</li> <li>• Printer.BARCODE_ITF</li> <li>• Printer.BARCODE_CODABAR</li> <li>• Printer.BARCODE_CODE93</li> <li>• Printer.BARCODE_CODE128</li> </ul>
	font	<ul style="list-style-type: none"> <li>• Printer.FONT_A (初期値)</li> <li>• Printer.FONT_B</li> <li>• Printer.FONT_C</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## TM-T90KP

## プリンター仕様

項目	仕様
初期改行量	30 ドット
ページモード初期領域	576 x 738 ドット
ページモード最大領域	576 x 1476 ドット
Font A のベースライン	文字の上端から 21 ドット目
Font B のベースライン	文字の上端から 21 ドット目
Font C のベースライン	文字の上端から 15 ドット目

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	<ul style="list-style-type: none"> <li>Printer.TM_T90KP</li> </ul>
addTextFont	font	<ul style="list-style-type: none"> <li>Printer.FONT_A (初期値)</li> <li>Printer.FONT_B</li> <li>Printer.FONT_C</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addTextStyle	color	<ul style="list-style-type: none"> <li>Printer.COLOR_1 (初期値)</li> <li>Printer.PARAM_UNSPECIFIED</li> <li>Printer.PARAM_DEFAULT</li> </ul>
addImage	color	<ul style="list-style-type: none"> <li>Printer.COLOR_1</li> <li>Printer.PARAM_DEFAULT</li> </ul>
	mode	<ul style="list-style-type: none"> <li>Printer.MODE_MONO</li> <li>Printer.MODE_MONO_HIGH_DENSITY</li> <li>Printer.PARAM_DEFAULT</li> </ul>

API	パラメーター	指定可能な設定値
addBarcode	type	<ul style="list-style-type: none"> <li>Printer.BARCODE_UPC_A</li> <li>Printer.BARCODE_UPC_E</li> <li>Printer.BARCODE_EAN13</li> <li>Printer.BARCODE_JAN13</li> <li>Printer.BARCODE_EAN8</li> <li>Printer.BARCODE_JAN8</li> <li>Printer.BARCODE_CODE39</li> <li>Printer.BARCODE_ITF</li> <li>Printer.BARCODE_CODABAR</li> <li>Printer.BARCODE_CODE93</li> <li>Printer.BARCODE_CODE128</li> </ul>
	font	<ul style="list-style-type: none"> <li>Printer.FONT_A (初期値)</li> <li>Printer.FONT_B</li> <li>Printer.FONT_C</li> <li>Printer.PARAM_DEFAULT</li> <li>Printer.PARAM_UNSPECIFIED</li> </ul>
addSymbol	type	<ul style="list-style-type: none"> <li>Printer.SYMBOL_PDF417_STANDARD</li> <li>Printer.SYMBOL_PDF417_TRUNCATED</li> </ul>
	level	<ul style="list-style-type: none"> <li>Printer.LEVEL_0</li> <li>Printer.LEVEL_1</li> <li>Printer.LEVEL_2</li> <li>Printer.LEVEL_3</li> <li>Printer.LEVEL_4</li> <li>Printer.LEVEL_5</li> <li>Printer.LEVEL_6</li> <li>Printer.LEVEL_7</li> <li>Printer.LEVEL_8</li> <li>Printer.PARAM_DEFAULT</li> <li>Printer.PARAM_UNSPECIFIED</li> </ul>
addSound	pattern	<ul style="list-style-type: none"> <li>Printer.PATTERN_NONE</li> <li>Printer.PATTERN_A</li> <li>Printer.PATTERN_B</li> <li>Printer.PATTERN_C</li> <li>Printer.PATTERN_D</li> <li>Printer.PATTERN_E</li> <li>Printer.PATTERN_ERROR</li> <li>Printer.PATTERN_PAPER_EMPTY</li> <li>Printer.PARAM_DEFAULT</li> </ul>
	repeat	<ul style="list-style-type: none"> <li>1 ~ 255</li> <li>Printer.PARAM_DEFAULT</li> </ul>

## TM-L90

## プリンター仕様

項目	レシート仕様	ダイカットラベル仕様
初期改行量	30 ドット	
ページモード初期領域	モノクロ印字	576 x 738 ドット
	2 色印字	576 x 369 ドット
ページモード最大領域	モノクロ印字	576 x 1476 ドット
	2 色印字	576 x 738 ドット
Font A のベースライン	文字の上端から 21 ドット目	
Font B のベースライン	文字の上端から 21 ドット目	
Font C のベースライン	文字の上端から 15 ドット目	

## パラメーター制限

API	パラメーター	指定可能な設定値
Printer	printerSeries	• Printer.TM_L90
addTextFont	font	• Printer.FONT_A (初期値) • Printer.FONT_B • Printer.FONT_C • Printer.PARAM_DEFAULT
addTextStyle	color	• Printer.COLOR_1 (初期値) • Printer.COLOR_2 • Printer.PARAM_UNSPECIFIED • Printer.PARAM_DEFAULT
addImage	color	• Printer.COLOR_1 • Printer.COLOR_2 • Printer.PARAM_DEFAULT

API	パラメーター	指定可能な設定値
addBarcode	type	<ul style="list-style-type: none"> <li>• Printer.BARCODE_UPC_A</li> <li>• Printer.BARCODE_UPC_E</li> <li>• Printer.BARCODE_EAN13</li> <li>• Printer.BARCODE_JAN13</li> <li>• Printer.BARCODE_EAN8</li> <li>• Printer.BARCODE_JAN8</li> <li>• Printer.BARCODE_CODE39</li> <li>• Printer.BARCODE_ITF</li> <li>• Printer.BARCODE_CODABAR</li> <li>• Printer.BARCODE_CODE93</li> <li>• Printer.BARCODE_CODE128</li> </ul>
	font	<ul style="list-style-type: none"> <li>• Printer.FONT_A (初期値)</li> <li>• Printer.FONT_B</li> <li>• Printer.FONT_C</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addSymbol	type	<ul style="list-style-type: none"> <li>• Printer.SYMBOL_PDF417_STANDARD</li> <li>• Printer.SYMBOL_PDF417_TRUNCATED</li> <li>• Printer.SYMBOL_QRCODE_MODEL_1</li> <li>• Printer.SYMBOL_QRCODE_MODEL_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_2</li> <li>• Printer.SYMBOL_MAXICODE_MODE_3</li> <li>• Printer.SYMBOL_MAXICODE_MODE_4</li> <li>• Printer.SYMBOL_MAXICODE_MODE_5</li> <li>• Printer.SYMBOL_MAXICODE_MODE_6</li> </ul>
	level	<ul style="list-style-type: none"> <li>• Printer.LEVEL_0</li> <li>• Printer.LEVEL_1</li> <li>• Printer.LEVEL_2</li> <li>• Printer.LEVEL_3</li> <li>• Printer.LEVEL_4</li> <li>• Printer.LEVEL_5</li> <li>• Printer.LEVEL_6</li> <li>• Printer.LEVEL_7</li> <li>• Printer.LEVEL_8</li> <li>• Printer.LEVEL_L</li> <li>• Printer.LEVEL_M</li> <li>• Printer.LEVEL_Q</li> <li>• Printer.LEVEL_H</li> <li>• Printer.PARAM_DEFAULT</li> <li>• Printer.PARAM_UNSPECIFIED</li> </ul>
addCut	type	<ul style="list-style-type: none"> <li>• Printer.CUT_FEED</li> <li>• Printer.CUT_NO_FEED</li> <li>• Printer.PARAM_DEFAULT</li> </ul>

## ファームウェアバージョンによる使用制限

プリンターに搭載されているファームウェアやソフトウェアのバージョンによって、使用できる API やパラメーターなどに制限があります。

ファームウェアやソフトウェアのバージョン確認、バージョンアップ方法は、各プリンターの詳細取扱説明書を参照してください。

### TM-i シリーズ / TM-DT シリーズ

- TM-DT ソフトウェア Ver.2.5、TM-i ファームウェア Ver.4.0 以降で対応

クラス	API	パラメーター	値
CommBox クラス	-	-	-

- TM-DT ソフトウェア Ver.2.5、TM-i ファームウェア Ver.4.1 以降で対応

クラス	API	パラメーター	値
Printer クラス	requestPrintJobStatus	-	-
	addSymbol	data	QR Code Micro
	addSymbol	type	Printer.SymbolType.QRCodeMicro
	setReceiveEventListener	args	PrintJobId

- TM-DT ソフトウェア Ver.3.02、TM-i ファームウェア Ver.4.4 以降で対応

クラス	API	パラメーター	値
Discovery クラス	-	-	-

- TM-DT ソフトウェア Ver.4.0 以降対応

クラス	API	パラメーター	値
POSKeyboard クラス	-	-	-
MSR クラス	-	-	-
CashChanger クラス	-	-	-
CAT クラス	-	-	-
OtherPeripheral クラス	-	-	-
Discovery クラス	start	deviceType	Discovery.TYPE_CCHANGER
			Discovery.TYPE_POS_KEYBOARD
			Discovery.TYPE_CAT
			Discovery.TYPE_MSR
			Discovery.TYPE_OTHER_PERIPHERAL

- TM-DT ソフトウェア Ver.4.1 以降対応

クラス	API	パラメーター	値
CAT クラス	getOposErrorCode	-	-
	sendDirectIOWCommand	-	-
	setDirectIOWCommandReplyEventListener	-	-
	setStatusUpdateEventListener	-	-
	setAuthorizeSalesEventListener	code	CODE_ERR_OPOSCODE
	setAuthorizeVoidEventListener	code	CODE_ERR_OPOSCODE
	setAuthorizeRefundEventListener	code	CODE_ERR_OPOSCODE
	setAuthorizeCompletionEventListener	code	CODE_ERR_OPOSCODE
	setAccessDailyLogEventListener	code	CODE_ERR_OPOSCODE

## TM-m30

- TM-m30 Firmware Ver.1.40 ESC/POS 以降で対応

クラス	API	パラメーター	値
BarcodeScanner クラス	-	-	-

- TM-m30 Firmware Ver.1.44 ESC/POS 以降で対応

クラス	API	パラメーター	値
Printer クラス	addCut	type	Printer.CUT_RESERVE

## TM-T88VI

- TM-T88VI Firmware Ver.40.50 ESC/POS 以降で対応

クラス	API	パラメーター	値
BarcodeScanner クラス	-	-	-

# サンプルプログラム

Epson ePOS SDK for Android が提供するサンプルプログラムは、Android アプリケーションソフト開発者向けの実装サンプルです。

Java ソースファイルを含む Android Studio 用 Android アプリケーションプロジェクトを提供します。



提供するサンプルプログラムは、サンプル API Level 14 で作成されています。

## 機能

サンプルプログラムは、以下の機能を実装しています。

### ePOS2Printer

- プリンターの検索、指定
  - 接続形式の指定
  - 検索結果一覧の表示
  - 検索結果から選択された接続に必要な target 情報を取得
- サンプルレシートの印刷
- サンプルクーポンの印刷
- プリンター状態確認
  - 印刷前にプリンターのステータスを確認する。
  - プリンターが印刷不可状態の場合、印刷不可原因の対処メッセージを通知する。
- 印刷結果、プリンターステータスの表示
  - 印刷終了後に、印刷結果を通知する。
  - 印刷結果通知時のプリンターステータスに対する対処メッセージを通知する。
  - プリンターの検索中、サンプルレシートの印刷中、サンプルクーポンの印刷中に API のエラーが発生した場合、エラーの発生をユーザーに通知する。
- ログ出力機能の設定
  - メイン画面起動時、端末ストレージへのログ出力を有効に設定する。

### ePOS2Discovery

- Filter Option の設定
- プリンター、周辺機器の検索
- 検索の開始
- 検索結果の表示
- 検索の停止
- エラーの通知
  - 検索の開始時、検索中、検索の停止時に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2LineDisplay

- 指定したテキストをカスタマーディスプレイに表示
- 点滅表示の ON/OFF 切り替え
- エラーの通知
  - カスタマーディスプレイとの接続に失敗した場合、接続失敗をユーザーに通知する。
  - カスタマーディスプレイへの表示データ送信時に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2Keyboard

- プリンターとの接続開始と終了
  - 指定されたキーボードへ接続する。
  - キーボードとの接続に失敗した場合、接続失敗をユーザーに通知する。
- キーボードで入力した文字をアプリケーションソフトのテキストボックスに表示
- エラーの通知
  - キーボードで文字入力中に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2POSKeyboard

- TM-DT との接続開始と終了
  - Target テキストボックスで指定した POS キーボードと接続を開始する。
  - 接続中の POS キーボードと接続を終了する。
- POS キーボードで入力したキーコードをアプリケーションソフトのテキストボックスに表示
- エラーの通知
  - POS キーボードで入力中に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2BarcodeScanner

- プリンターとの接続開始と終了
  - 指定されたバーコードスキャナーへ接続する。
  - バーコードスキャナーとの接続に失敗した場合、接続失敗をユーザーに通知する。
- バーコードスキャナーで読み取ったデータを、アプリケーションソフトのテキストボックスに表示
- エラーの通知
  - バーコードスキャナーでの読み取り中に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2MSR

- TM-DTとの接続開始と終了
  - Targetテキストボックスで指定したMSRと接続を開始する。
  - 接続中のMSRと接続を終了する。
- MSRで読み取ったデータを、アプリケーションソフトのテキストボックスに表示
- エラーの通知
  - MSRでの読み取り中にAPIのエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2SimpleSerial

- プリンターとの接続開始と終了
  - 指定されたシリアル通信デバイスへ接続する。
  - シリアル通信デバイスとの接続に失敗した場合、接続失敗をユーザーに通知する。
- アプリケーションソフトで指定したデータを、シリアル通信デバイスへ送信
- シリアル通信デバイスからの応答コマンドを、アプリケーションソフトに表示
- エラーの通知
  - シリアル通信デバイスへデータ送信中にAPIのエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2CashChanger

デバイス制御プログラムから制御する場合と、デバイス制御スクリプトから制御する場合で、使用できる機能が異なります。

使用できる機能の差分は、[APIリファレンス](#)を参照してください。

- TM-DTとの接続開始と終了
  - Targetテキストボックスで指定した自動つり銭機と接続を開始する。
  - 接続中の自動つり銭機と接続を終了する。
- 計数モードと残置を指定
  - 手入力モードと自動計数モードから選択する。
  - 残置する金額を指定する。
- 自動つり銭機内の金種と枚数情報を取得し、結果をアプリケーションソフトのテキストボックスに表示
- 入金処理を実行し、結果をアプリケーションソフトのテキストボックスに表示
  - 入金処理を開始する。
  - 入金処理を一時停止する。
  - 入金処理を再開する。
  - 終了動作を指定して入金処理を終了する。
- 出金処理を実行
  - 金額を指定して出金する。
  - 紙幣と硬貨の枚数を指定して出金する。
- 自動つり銭機内の現金回収処理を実行し、結果をアプリケーションソフトのテキストボックスに表示
- キャッシュドロワーを開く処理を実行し、結果をアプリケーションソフトのテキストボックスに表示
- 任意コマンドを送信し、結果をアプリケーションソフトのテキストボックスに表示

- テキストボックスに入力した任意のコマンドを送信する。
  - テキストボックスに入力した OPOS の DirectIO メソッドを送信する。
- 自動つり銭機の収納庫ステータスや機器の状態変化を通知
- ステータス変更イベントを受け取り、アプリケーションのテキストボックスに表示する。
  - DirectIOEvent を受け取り、アプリケーションのテキストボックスに表示する。
  - StatusUpdateEvent を受け取り、アプリケーションのテキストボックスに表示する。
- エラーの通知
- 自動つり銭機での処理実行時に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2CAT

- TM-DT との接続開始と終了
- Target テキストボックスで指定した信用照会端末と接続を開始する。
  - 接続中の信用照会端末と接続を終了する。
- 練習モードと通常モードの切り替え
- 決済の種類と決済メディアを選択し、決済金額を指定して実行
- 決済実行結果をアプリケーションソフトのテキストボックスに表示
- 日計情報を取得し、結果をアプリケーションソフトのテキストボックスに表示
- エラーの通知
- 信用照会端末での処理実行時に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2OtherPeripheral

- TM-DT との接続開始と終了
- Target テキストボックスで指定したデバイスと接続を開始する。
  - 接続中のデバイスと接続を終了する。
- アプリケーションソフトで指定した JSON データを、デバイスへ送信
- デバイスからの応答コマンドを、テキストボックスに表示
- エラーの通知
- デバイスでの処理実行時に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2CommBox

- プリンターとの接続開始と終了
  - 指定されたコミュニケーションボックスへ接続する。
  - コミュニケーションボックスとの接続に失敗した場合、接続失敗をユーザーに通知する。
- アプリケーションソフトで指定したメンバー ID にメッセージを送信
- エラーの通知
  - メッセージ送信時に API のエラーが発生した場合、エラーの発生をユーザーに通知する。

## ePOS2EasySelection

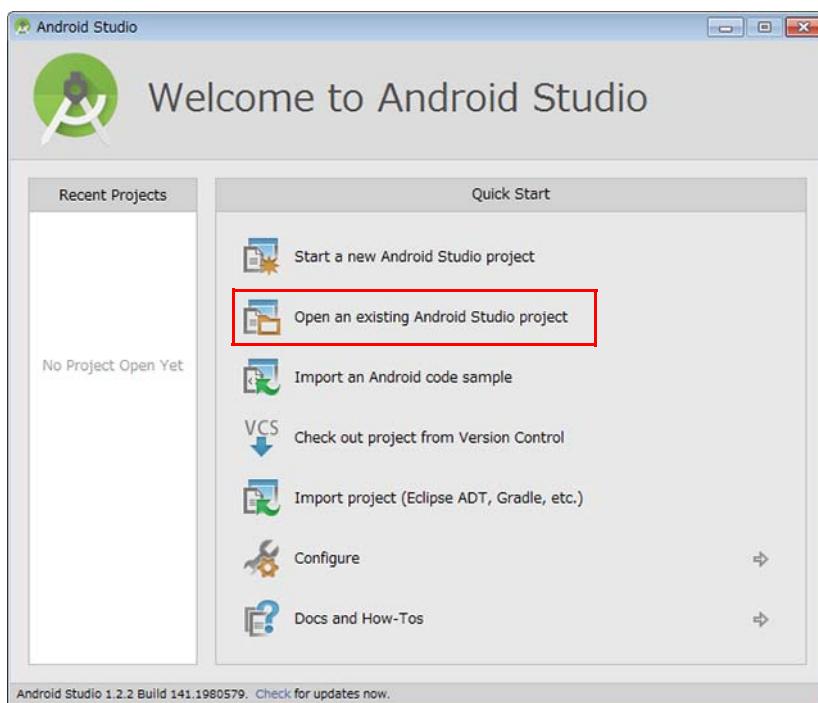
- NFC タグ / QR コードを使用してプリンターに簡単接続
  - NFC タグ からプリンター情報の取得
  - QR コードからプリンター情報の取得
  - 取得したプリンター情報からのポートオープン
- 簡単選択可能な QR コードの印刷
  - プリンターの検索結果からプリンター情報の QR コード作成

# 使用環境

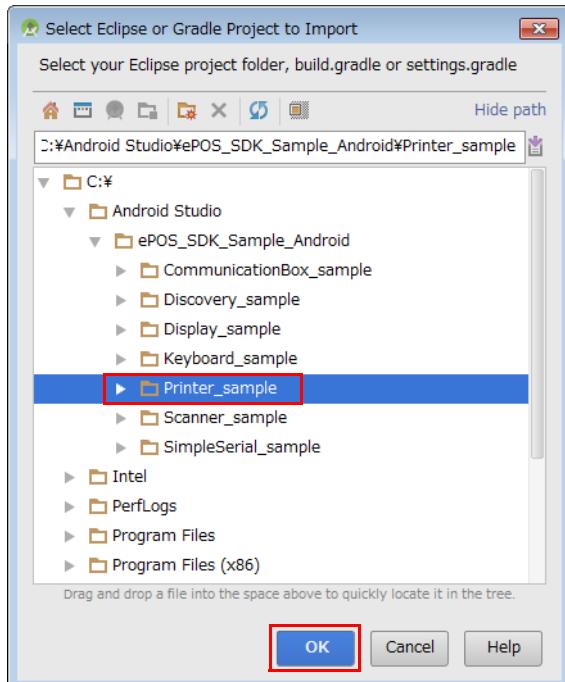
Epson ePOS SDK for Android の開発環境、アプリケーションソフトの動作環境を参照してください。

## インストール手順

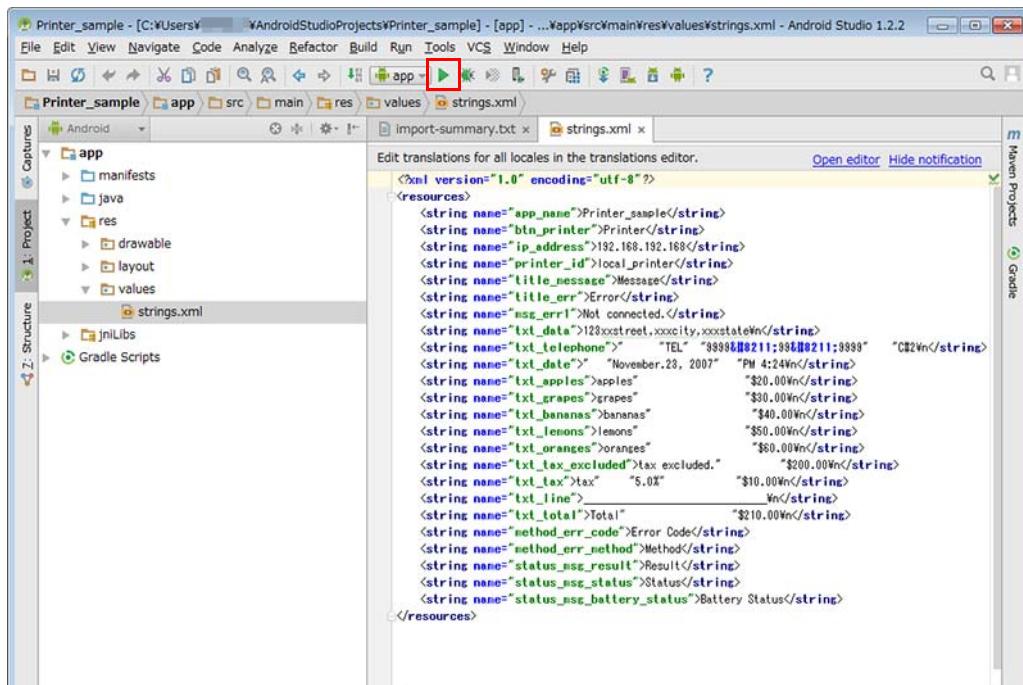
- 1** サンプルプログラムの zip ファイルを展開し、展開したファイルを任意の位置に置きます。
- 2** Android Studio を起動し、「Open an existing Android Studio Project」をクリックします。



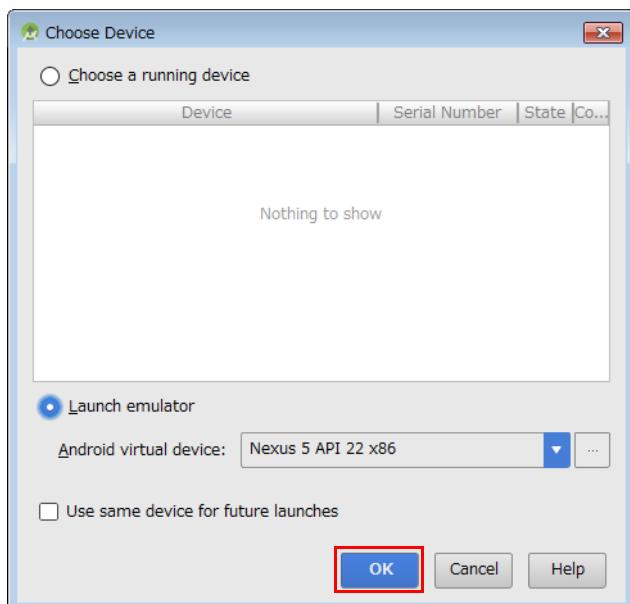
**3** 手順1で用意したサンプルプログラムからインストールするサンプルプログラムを選択し、[OK]をクリックします。



**4** [Run 'app']をクリックします。



5 デバイスを選択し、[OK] をクリックします。



# 活用ガイド

## USB接続するには

インターフェイスがUSBの場合、あらかじめアプリケーションソフトでUSBデバイスへのアクセス許可を取得することを推奨します。



あらかじめUSBデバイスへのアクセス許可を取得せず、connectでポートオープンする場合、以下の注意点があります。

- アクセス許可取得のダイアログで[OK]を押すと、ポートオープンまでに10秒前後の時間がかかります。
- アクセス許可取得のダイアログで[キャンセル]を押すと、30秒のタイムアウト待ちになります。

アプリケーションソフトでアクセス許可を取得する方法は以下の通りです。

### 1 AndroidManifest.xmlに以下のコードを追記します。

```
<manifest ...>
  <application>
    <activity ...>
      <intent-filter>
        <action android:name="android.hardware.usb.action.USB_DEVICE_ATTACHED" />
      </intent-filter>
      <meta-data android:name="android.hardware.usb.action.USB_DEVICE_ATTACHED"
        android:resource="@xml/device_filter" />
    </activity>
  </application>
</manifest>
```

### 2 ソースファイルにres/xml/device\_filter.xmlを追加します。

### 3 device\_filter.xmlファイルに以下のコードを記述します。

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <usb-device vendor-id="1208" />
</resources>
```

### 4 アクセス許可を取得する際、ダイアログが表示されます。[OK]を押します。

# プリンターを検索するには

```

検索開始
FilterOption filterOption = null;
filterOption = new FilterOption();

// 検索対象のポートを指定する。
filterOption.PortType = Discovery.SOCKET_ALL;

// Broadcast アドレスを指定する。
filterOption.broadcast = "255.255.255.255";

// 検索するデバイスモデルを指定する。
filterOption.deviceModel = Discovery.MODEL_ALL;

// 検索するデバイスの種類を指定する。
filterOption.deviceType = Discovery.TYPE_ALL;

検索を開始する
try {
    Discovery.start(mContext, filterOption, mDiscoveryListener);
}
catch (Exception e) {
    // エラーメッセージを表示する
}

検知されたデバイスを通知する
private DiscoveryListener mDiscoveryListener = new DiscoveryListener() {
    @Override
    public void onDiscovery(final DeviceInfo deviceInfo) {
        runOnUiThread(new Runnable() {
            @Override
            public synchronized void run() {
                // 検知されたデバイスをアプリケーションソフトに表示する
            }
        });
    }
};

検索を停止する
private void stopDiscovery() {
    try {
        Discovery.stop();
    }
    catch (Epos2Exception e) {
        // エラーメッセージを表示する
    }
}

```

## 常時監視するには

アプリケーションソフト起動時  
Printer printer = null;

```
try {
    printer = new Printer(printer.TM_T88, printer.MODEL_ANK, this);
}
catch (Exception e) {
    // エラーメッセージを表示する
}

printer.setStatusChangeListener(this);

try {
    printer.connect("TCP:192.168.192.168", printer.PARAM_DEFAULT);
}
catch (Exception e) {
    // エラーメッセージを表示する
}

try {
    printer.setInterval(3000);
    printer.startMonitor();
}
catch (Exception e) {
    // エラーメッセージを表示する
}
```

繰り返す  
// 印字データをバッファリングする (addText..)  
// 印字データを送信する (sendData)

アプリケーションソフト終了時  
try {
 printer.stopMonitor();
}
catch (Exception e) {
 // エラーメッセージを表示する
}
try {
 printer.disconnect();
}
catch (Exception e) {
 // エラーメッセージを表示する
}

```
public void onPtrStatusChange(Printer printerObj, final int eventType) {  
    runOnUiThread(new Runnable() {  
        @Override  
        public synchronized void run() {  
            switch (eventType) {  
                case Printer.EVENT_ONLINE:  
                    break;  
                case Printer.EVENT_OFFLINE:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_POWER_OFF:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_COVER_CLOSE:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_COVER_OPEN:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_PAPER_OK:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_PAPER_NEAR_END:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_PAPER_EMPTY:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_DRAWER_HIGH:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_DRAWER_LOW:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_BATTERY_ENOUGH:  
                    // 対処メッセージ表示  
                    break;  
                case Printer.EVENT_BATTERY_EMPTY:  
                    // 対処メッセージ表示  
                    break;  
                default:  
                    break;  
            }  
        }  
    });  
}
```



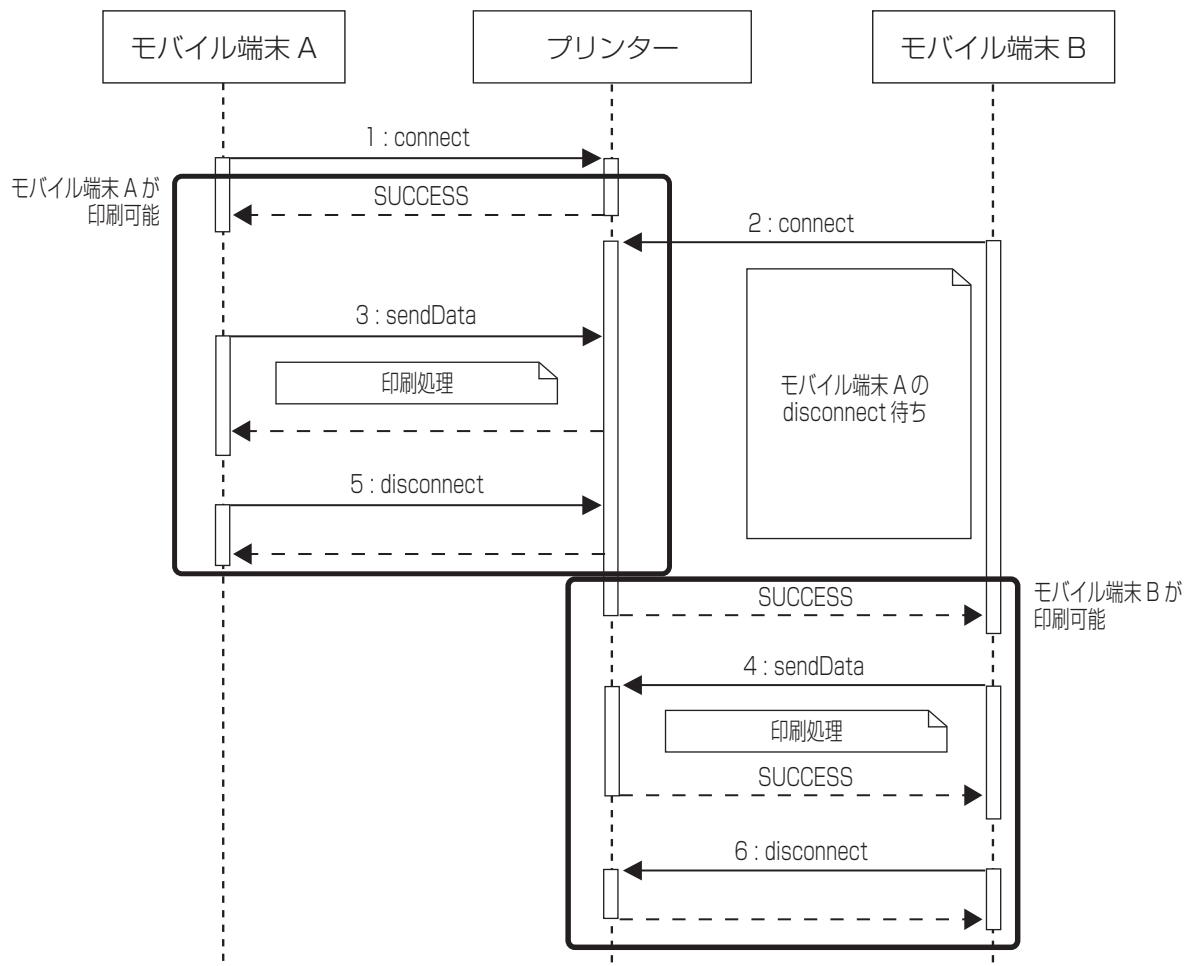
「対処メッセージ表示」に表示するメッセージは、[Error Status と対処方法](#)を参照してください。

# 一台のプリンターを複数のモバイル端末から使用するには

モバイル端末 A とモバイル端末 B から、1 台のプリンターを使用する場合の処理の流れを以下に示します。

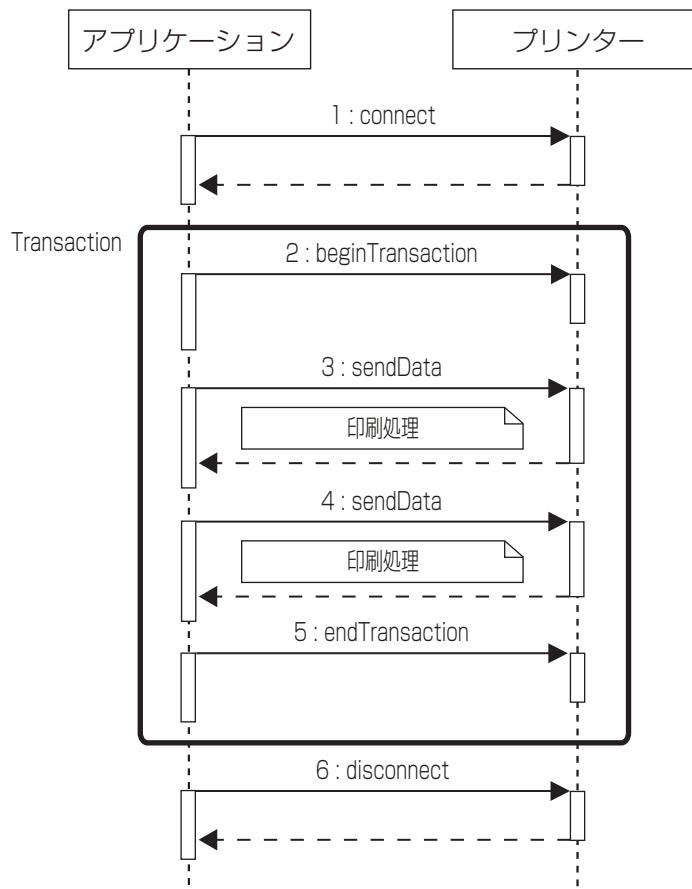


以下の図は、TM プリンターを制御する場合の処理の流れを説明しています。



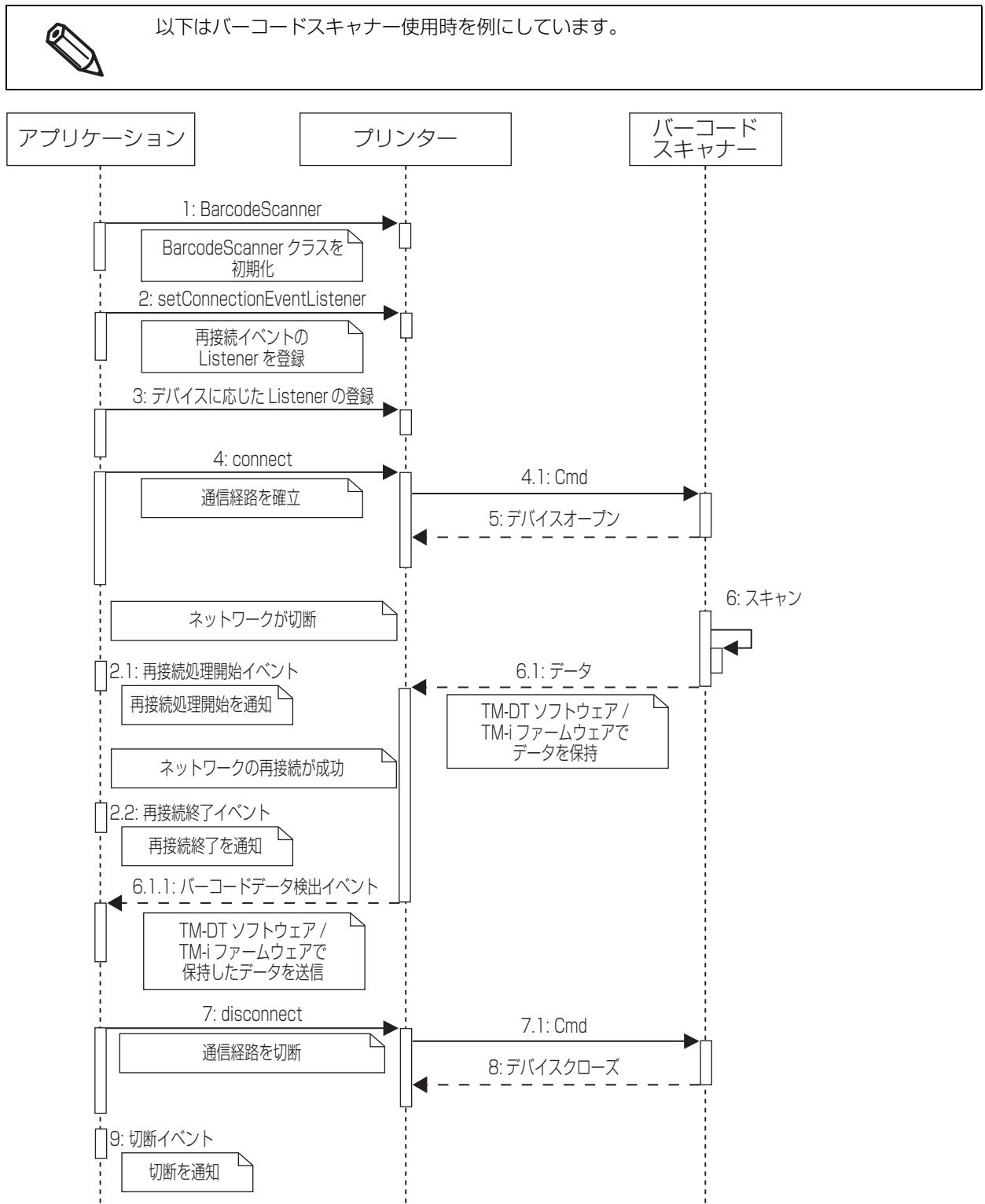
## トランザクションを指定するには

連続して行うひとまとめの印刷処理（一枚のレシートや一枚のクーポンなど）を `beginTransaction` と `endTransaction` で囲んでください。



# ネットワークを自動で再接続するには

プリンターとのネットワーク通信が切断された場合、自動で再接続するシーケンスは、以下の通りです。



## アプリケーション間でデータを送受信するには

TM-i シリーズ / TM-DT シリーズを使用すると、コミュニケーションボックス機能を使用して、簡単なプログラムでアプリケーション間でのデータの送受信が可能です。送受信できるデータはテキストデータ、イメージデータです。

### コミュニケーションボックスを使用したアプリケーション例

#### スマートデバイスのカスタマーディスプレイ化

プリンターに接続したバーコードスキャナーでデータを読み込み、カスタマーディスプレイ化したスマートデバイスへ送信します。

データを受信したスマートデバイスは、イメージデータとともに受信データを表示します。

#### スマートデバイスを使った受付端末システム

お客様のスマートデバイスで入力した予約を店舗のスマートデバイスで受け付け、準備完了後に呼び出しメールをお客様のスマートデバイスへ送信します。

#### オーダーエントリーシステムとキッチンディスプレイ

店員がスマートデバイスを使って注文した料理をキッチンのスマートデバイス（キッチンディスプレイ）で管理し、調理完了後に配膳依頼を店員のスマートデバイスへ送信します。

#### デジタルサイネージ(電子看板)

店舗のメインスマートデバイスから、各コーナーのスマートデバイスへ適切な「お知らせ」を送信します。

#### 周辺機器の使用権管理

TM-DT シリーズに接続した周辺機器を複数のスマートデバイスから使用する場合、周辺機器の使用が競合する可能性があります。コミュニケーションボックスを利用すると、この問題を解決できます。

例：

周辺機器の使用権を取得したスマートデバイスが、使用権の取得を他のスマートデバイスへ一斉送信します。

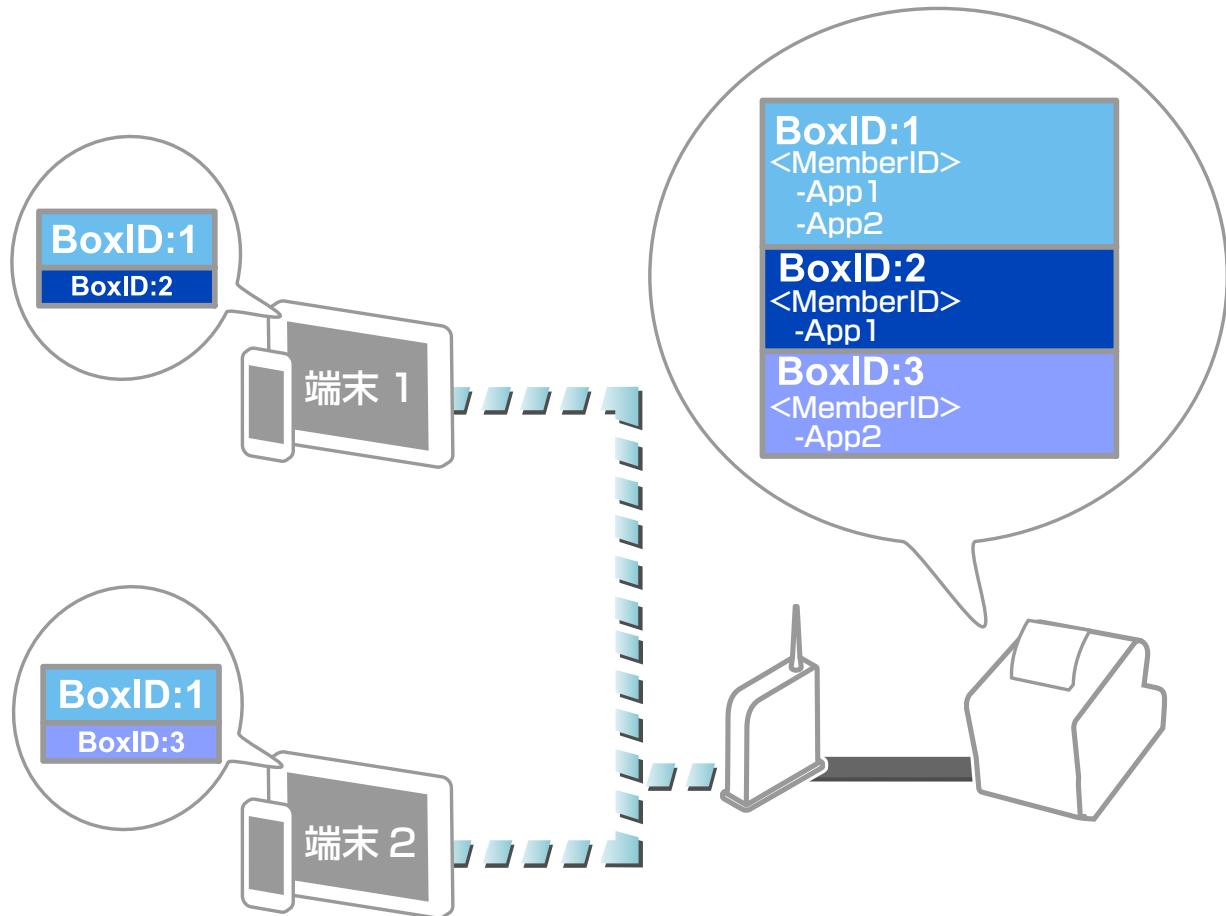
周辺機器を使い終わったら、使用権の破棄を他のスマートデバイスへ一斉送信することで、次のスマートデバイスが使用権を取得して周辺機器を使用できます。

## コミュニケーションボックスの仕組み

コミュニケーションボックスはボックス ID で管理します。

コミュニケーションボックスに所属しているアプリケーション間でデータの送受信が可能です。

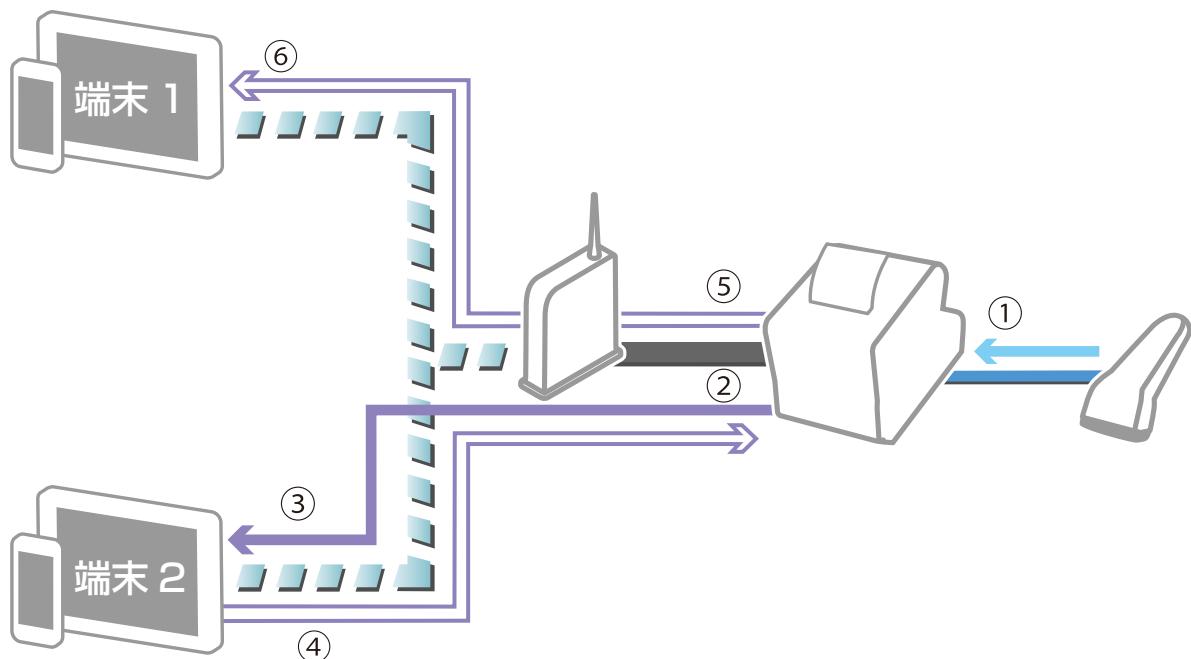
下図の場合、端末 1 と端末 2 のアプリケーションは、ボックス ID 「BoxID:1」 のコミュニケーションボックスを使って、アプリケーション間でデータの送受信ができます。



## コミュニケーションボックスの仕様

機能	仕様
作成できるコミュニケーションボックスの最大数	20 個
1つのコミュニケーションボックスに所属できるアプリケーションの最大数	20 個
1つのコミュニケーションボックスが保持できる送信履歴の容量	10240 バイト
一度に送信できるデータサイズ	1024 バイト

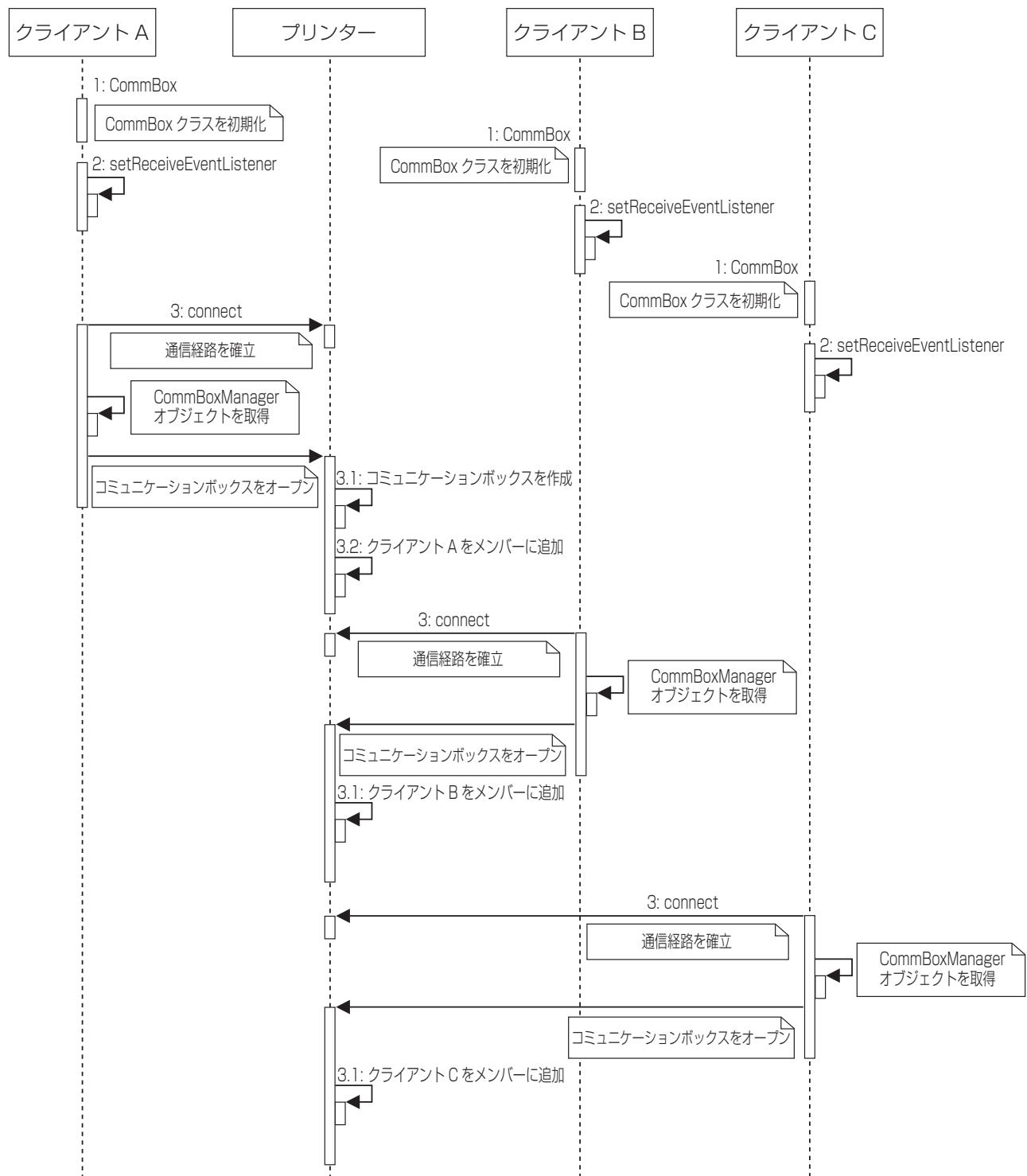
## コミュニケーションボックスの使用したデータ処理例

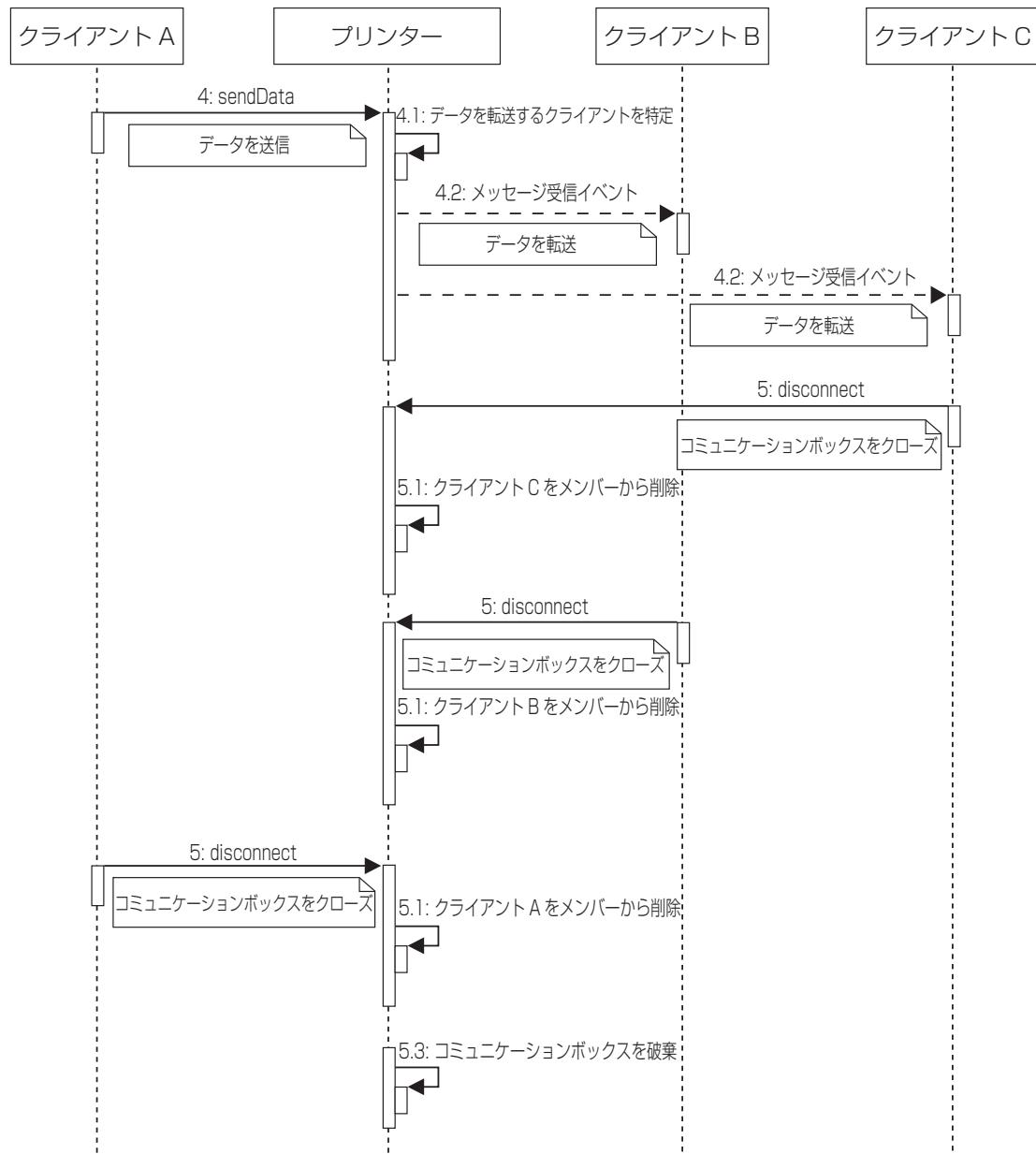


- 1** バーコードスキャナーが読み取ったバーコードデータを、プリンターが受信します。
- 2** プリンターは、バーコードデータを端末2に通知します。
- 3** 端末2はバーコードデータを取得し、POSデータに変換します。
- 4** 端末2は、コミュニケーションボックスにPOSデータを送信します。
- 5** POSデータが格納されたことを端末1に通知します。
- 6** 端末1は、コミュニケーションボックスに格納されたPOSデータを取得します。

## 基本的なプログラミングシーケンス

シーケンス図中の「クライアント」は、アプリケーションを指しています。



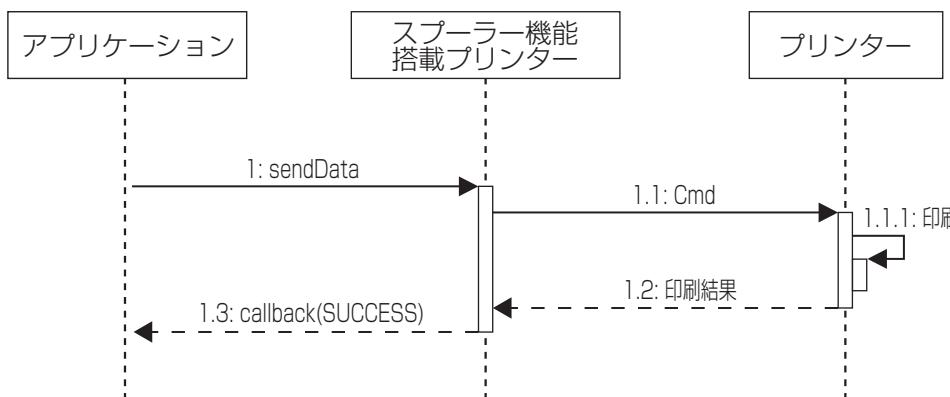


## 迂回印刷するには

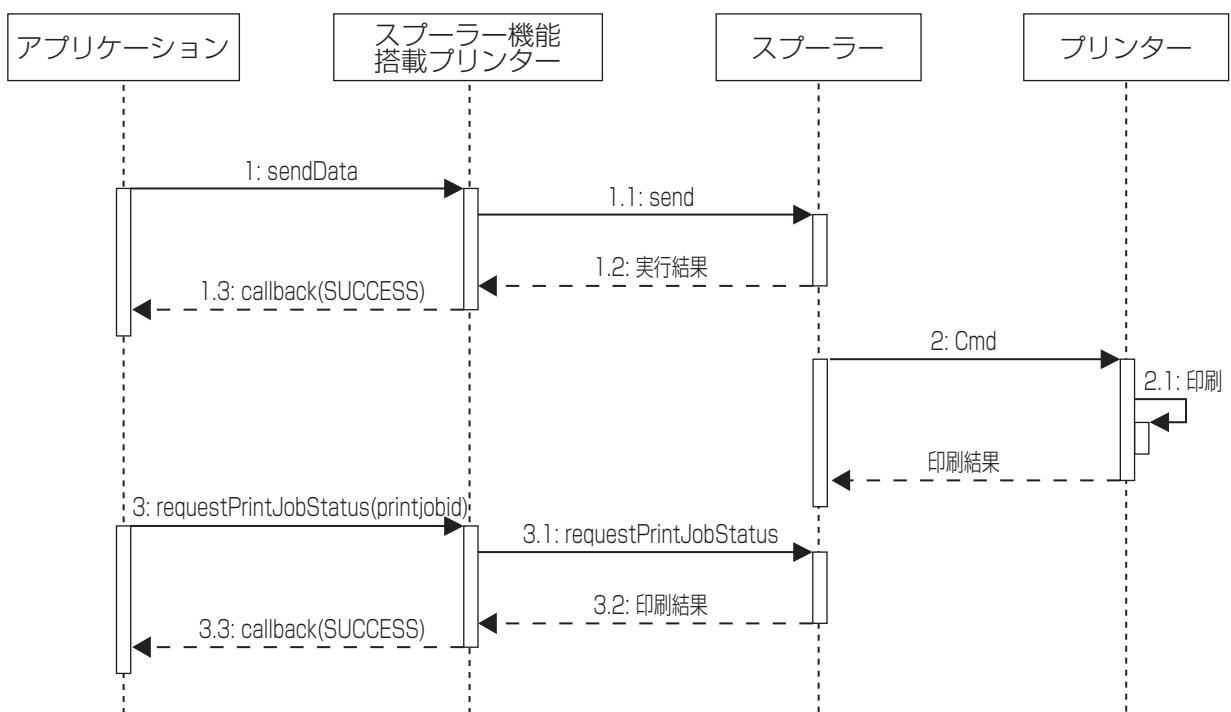
プリンターのスプーラー機能を使用します。

スプーラー機能を搭載したプリンターは、[プリンターごとの提供機能](#)を参照してください。

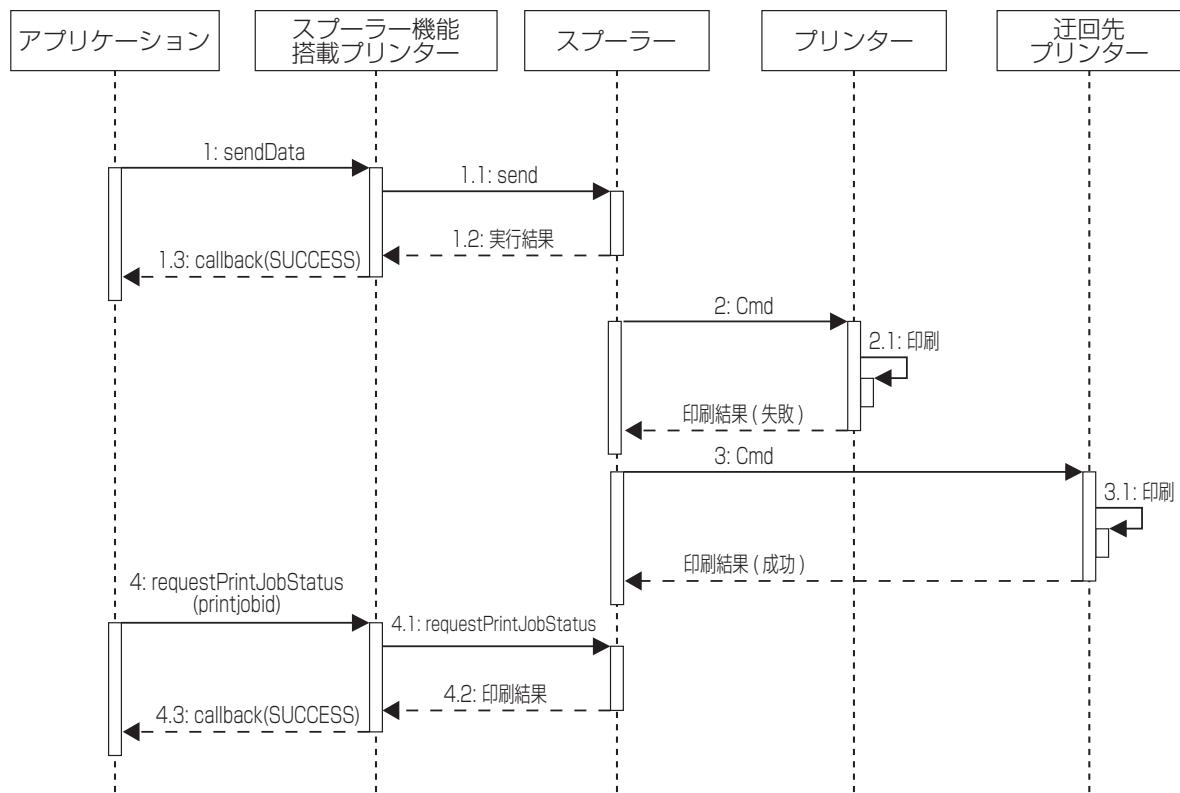
スプーラー機能を無効にした場合、アプリケーションからスプーラー機能搭載プリンターにリクエストを送信すると、印刷を直ちに実行し、印刷完了後にレスポンスをアプリケーションに返します。



スプーラー機能を有効にした場合、アプリケーションからスプーラー機能搭載プリンターにリクエストを送信すると、印刷データをキューに入れ、印刷完了を待たずにレスポンスをアプリケーションに返します。



この時、出力先のプリンターが印刷できない場合、スプーラー機能搭載プリンターは、アプリケーションにエラーを返しません。迂回印刷を有効にすると、代替プリンターで印刷でき、アプリケーションは、印刷結果も後で取得できます。以下のシーケンス図を参照してください。



## NFC タグを使ってプリンターを選択するには

EasySelect クラスの `parseNFC` を使って NFC タグを解析します。解析した結果に基づいて `connect` API を呼び出すことでプリンターと接続できます。

```
Printer printer;

@Override
protected void onNewIntent(Intent intent) {

    //NFC タグは、onNewIntent で受け取る
    EasySelect easySelect = new EasySelect();

    Tag tag = (Tag)intent.getParcelableExtra( NfcAdapter.EXTRA_TAG );

    //NFC タグの解析
    ArrayList<EasySelectInfo> easySelectInfoArray = null;
    easySelectInfoArray = easySelect.parseNFC( tag, PARSE_NFC_TIMEOUT_DEFAULT );
    if ((null == easySelectInfoArray) || (easySelectInfoArray.size() == 0)){
        // 簡単選択用の NFC ではなかった場合
        return ;
    }

    // 一つ目の NFC データを取得する
    EasySelectInfo easySelectInfo = easySelectInfoArray.get(0);

    String printerName = easySelectInfo.printerName;
    if(printerName == null || printerName.equals("")){
        printerName = "TM-T88V";
        // 使用するプリンターネーム
    }

    // プリンターネームから PrinterSeries への変換
    int printerSeries = convertPrinterNameToPrinterSeries(printerName);

    String macAddress = easySelectInfo.macAddress;
    if(macAddress == null || macAddress.equals("")){
        //macAddress を取得できなかった場合
    }

    try {
        // Printer の初期化
        printer = new Printer(printerSeries, Printer.MODEL_ANK, getApplicationContext());

        // connect パラメータの生成
        String targetText = convertEasySelectInfoToTargetString(easySelectInfo);

        // プリンターとの接続
        printer.connect(targetText, Printer.PARAM_DEFAULT);

        // 例外処理
    } catch (Epos2Exception e) {
        // 例外処理を記述する
    }
}
```

```
private int convertPrinterNameToPrinterSeries(String printerName) {  
    int printerSeries = Printer.TM_T88;  
  
    if(printerName.equals("TM-T88V")){  
        printerSeries = Printer.TM_T88;  
    }else if(printerName.equals("TM-m10")){  
        printerSeries = Printer.TM_M10;  
    }else if(printerName.equals("TM-m30")){  
        printerSeries = Printer.TM_M30;  
    }else if(printerName.equals("TM-P20")){  
        printerSeries = Printer.TM_P20;  
    }else if(printerName.equals("TM-P60II")){  
        printerSeries = Printer.TM_P60II;  
    }else if(printerName.equals("TM-P80")){  
        printerSeries = Printer.TM_P80;  
    }else{  
        // 利用するプリンターに合わせて変換処理を追加  
    }  
  
    return printerSeries;  
}  
  
private String convertEasySelectInfoToString(EasySelectInfo easySelectInfo) {  
    StringBuffer sb = new StringBuffer();  
  
    switch(easySelectInfo.deviceType){  
        case EasySelectDeviceType.TCP:  
            sb.append("TCP:");  
            break;  
        case EasySelectDeviceType.BLUETOOTH:  
            sb.append("BT:");  
            break;  
    }  
    sb.append(easySelectInfo.macAddress);  
    return sb.toString();  
}
```



本機能は、NFC 対応製品のみ使用できます。

## QRコードを使ってプリンターを選択するには

EasySelect クラスの `parseQR` を使って QR コードを解析します。解析した結果に基づいて `connect` API を呼び出すことでプリンターと接続できます。

```
EasySelect easySelect = new EasySelect();
String data;
Printer printer;

public void ConnectPrinterByQRCode() {

    // カメラ画像から取得した QR コードデータを格納

    // QR コードの解析
    EasySelectInfo easySelectInfo = easySelect.parseQR(data);
    if (null == easySelectInfo) {
        // 簡単選択用の QR コードでは無かった場合
        return ;
    }

    // プリンターナンバーから PrinterSeries への変換
    int printerSeries = convertPrinterNameToPrinterSeries(easySelectInfo.printerName);

    try {
        // Printer の初期化
        printer = new Printer(printerSeries, Printer.MODEL_ANK, getApplicationContext());

        // connect パラメーターの生成
        String targetText = convertEasySelectInfoToTargetString(easySelectInfo);

        // プリンターとの接続
        printer.connect(targetText, Printer.PARAM_DEFAULT);

        // 例外処理
    } catch (Epos2Exception e) {
        // 例外処理を記述する
    }
}

private int convertPrinterNameToPrinterSeries(String printerName) {

    int printerSeries = Printer.TM_T88;

    if(printerName.equals("TM-T88V")){
        printerSeries = Printer.TM_T88;
    }else if(printerName.equals("TM-m10")){
        printerSeries = Printer.TM_M10;
    }else if(printerName.equals("TM-m30")){
        printerSeries = Printer.TM_M30;
    }else if(printerName.equals("TM-P20")){
        printerSeries = Printer.TM_P20;
    }else if(printerName.equals("TM-P60II")){
        printerSeries = Printer.TM_P60II;
    }else if(printerName.equals("TM-P80")){
        printerSeries = Printer.TM_P80;
    }else{
        // 利用するプリンターに合わせて変換処理を追加
    }

    return printerSeries;
}
```

```
private String convertEasySelectInfoToTargetString(EasySelectInfo easySelectInfo) {  
    StringBuffer sb = new StringBuffer();  
  
    switch(easySelectInfo.deviceType){  
        case EasySelectDeviceType.TCP:  
            sb.append("TCP:");  
            break;  
        case EasySelectDeviceType.BLUETOOTH:  
            sb.append("BT:");  
            break;  
    }  
    sb.append(easySelectInfo.macAddress);  
    return sb.toString();  
}
```

## プリンター選択用の QR コードの印刷方法

### ステータスシートに QR コードを印刷可能な製品

ステータスシートに QR コードを印刷できる製品があります。ステータスシートの印刷方法は各製品の詳細取扱説明書を参照してください。

### ステータスシートに QR コードを印刷できない製品

[createQR](#) を使用して QR コードを作成します。それを [addSymbol](#) に指定して印刷します。

サンプルプログラムの「QR コードを作成する」を参照してください。

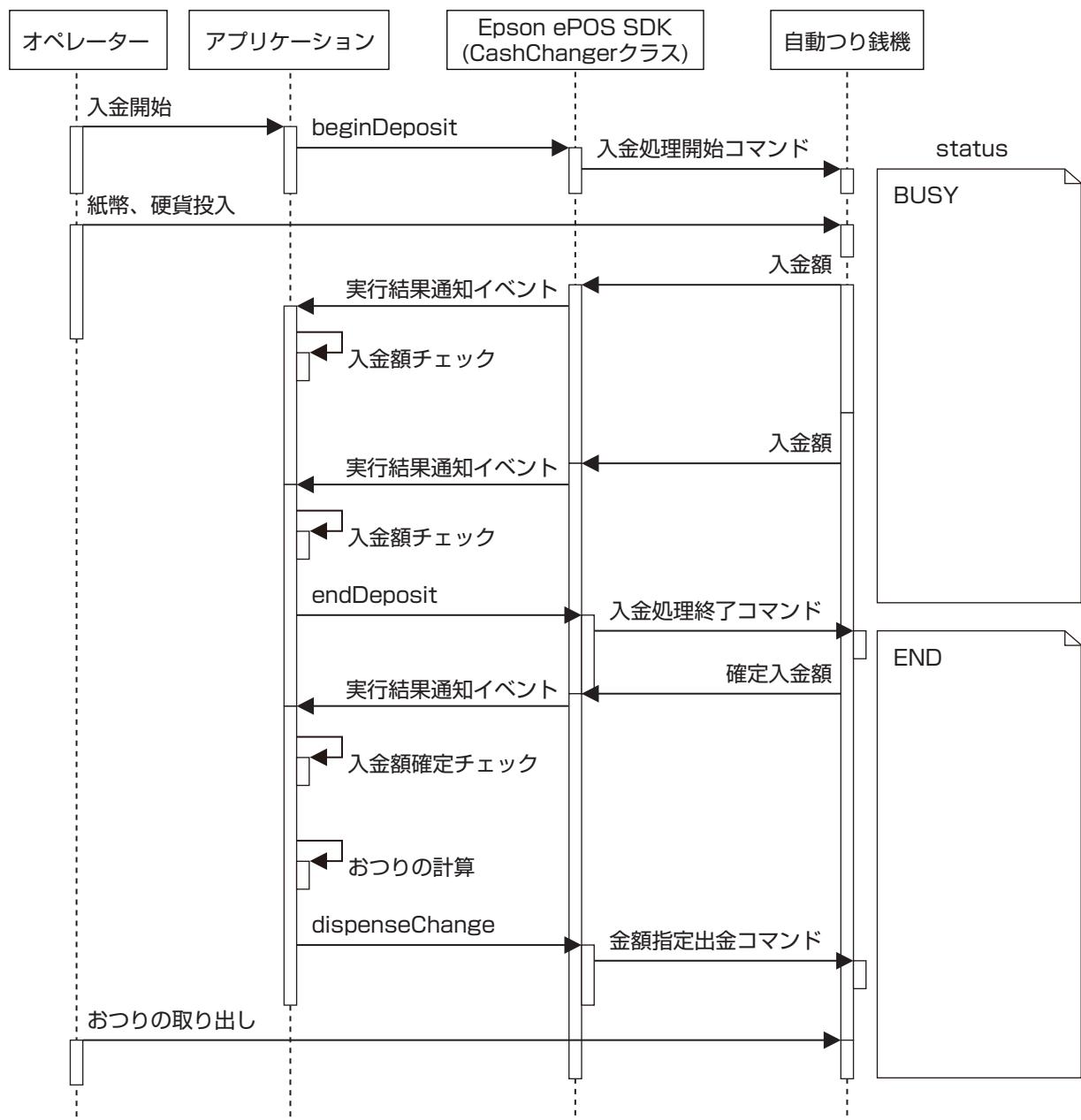
# 自動つり銭機を制御するには

[CashChanger クラス](#)で自動つり銭機を制御します。

デバイス制御プログラムを使う場合と、デバイス制御スクリプトを使う場合で制御シーケンスが異なります。

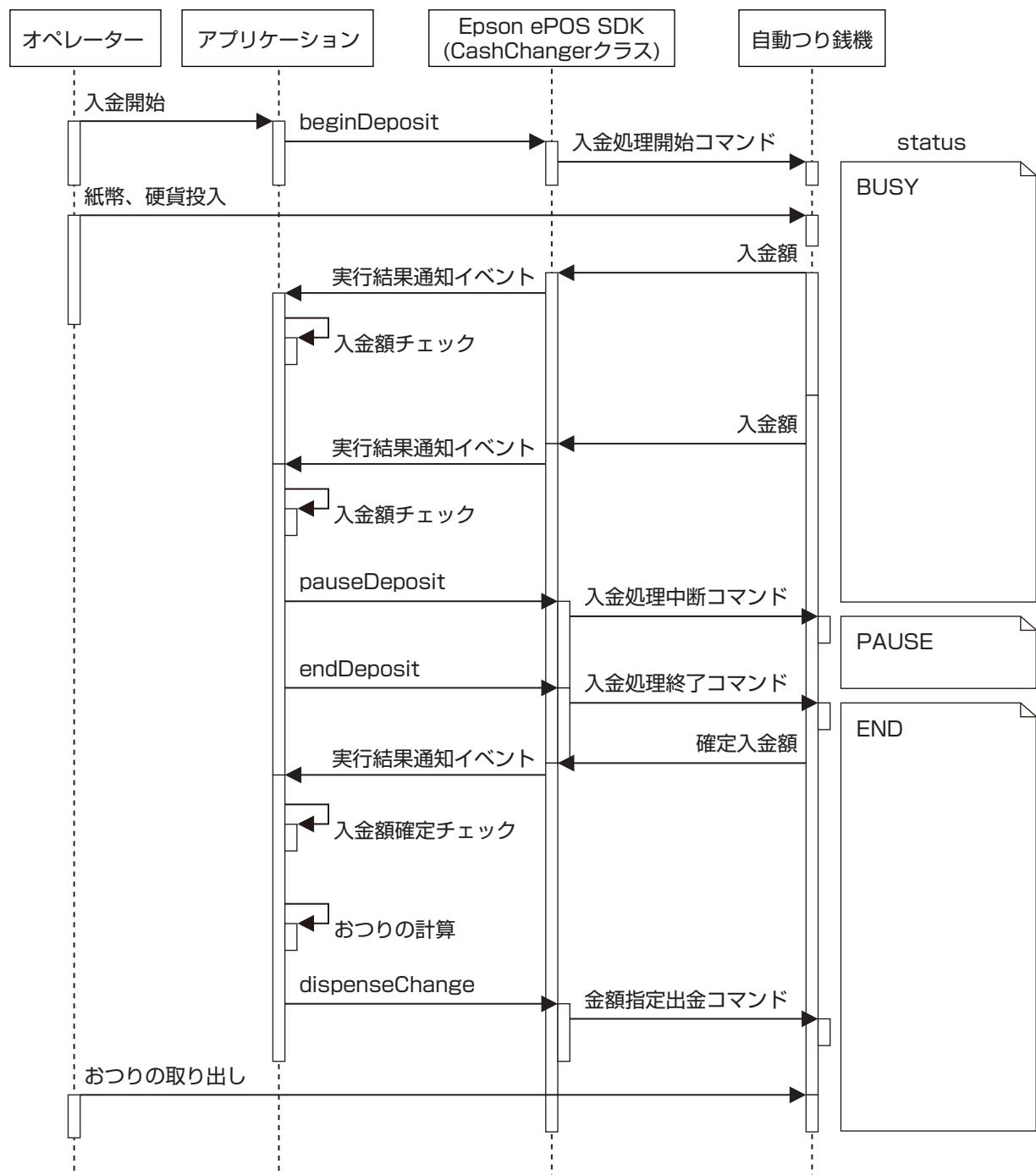
## デバイス制御プログラム

- 紙幣や硬貨が投入されるたびに、入金額を通知する実行結果通知イベント（[setDepositEventListener](#)）が通知されます。
- 自動つり銭機内の処理によって、紙幣や硬貨の投入後に実行結果通知イベント（[setDepositEventListener](#)）が複数回通知されることがあります。
- 入金額のチェックは、入金額が精算金額を上回るまで繰り返してください。
- 入金額確定チェック時は、自動つり銭機の status が END であることを確認してください。



## デバイス制御スクリプト

- 紙幣や硬貨が投入されるたびに、入金額を通知する `setDepositEventListener` が通知されます。
- 自動つり銭機内の処理によって、紙幣や硬貨の投入後に `setDepositEventListener` が複数回通知されることがあります。
- 入金額のチェックは、入金額が精算金額を上回るまで繰り返してください。
- 入金額確定チェック時は、自動つり銭機の `status` が `END` であることを確認してください。

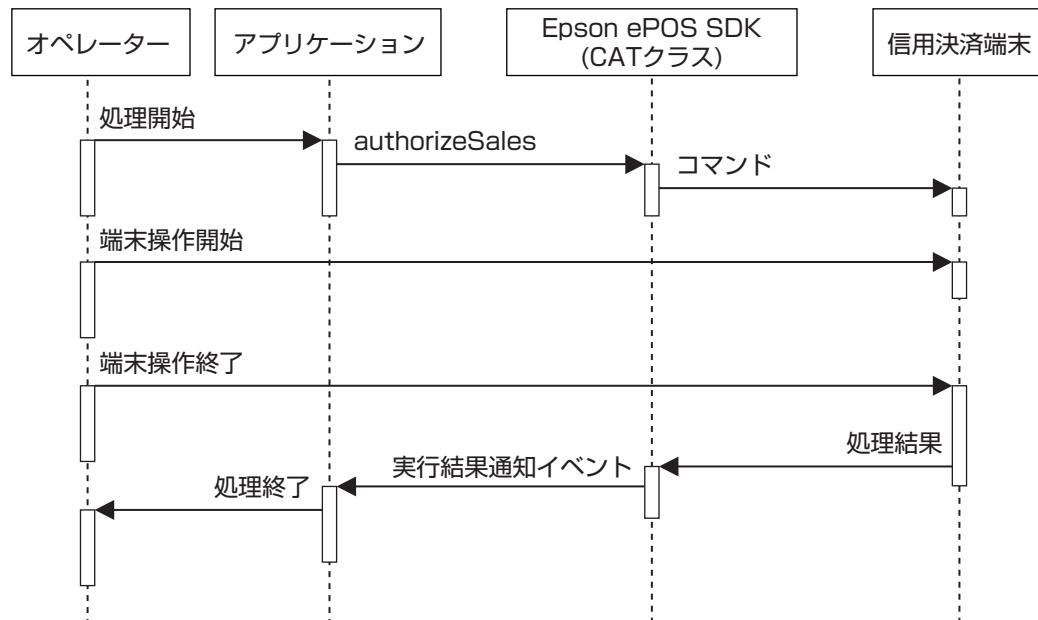


## 信用決済端末を制御するには

以下に [CAT クラス](#) で信用決済端末を制御するシーケンス図を示します。



以下のシーケンス図では、売上処理をする場合の制御シーケンスを例にしています。

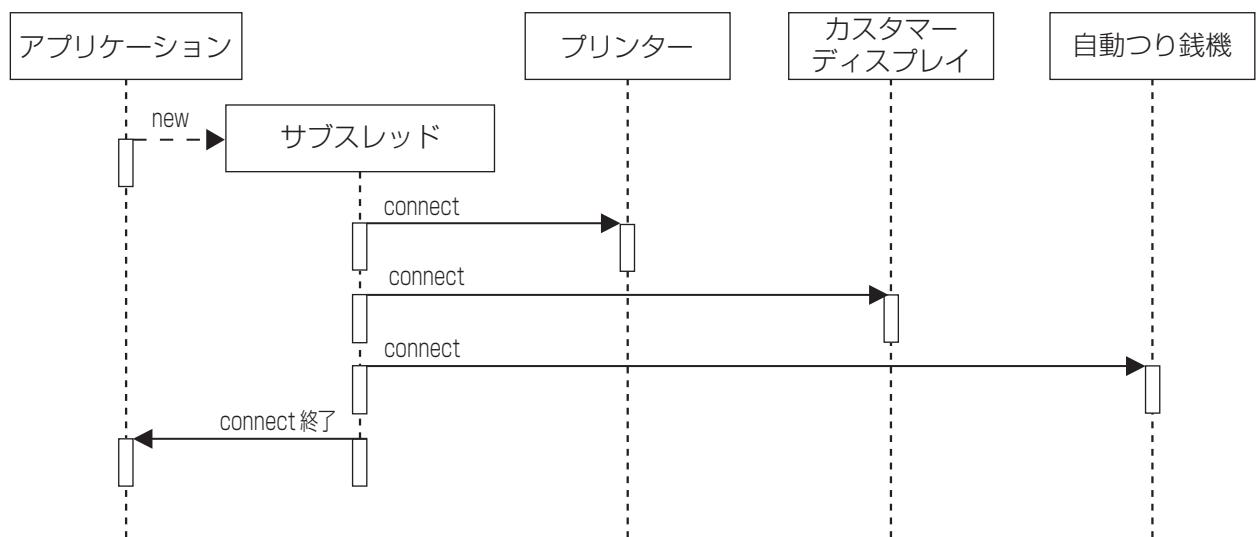


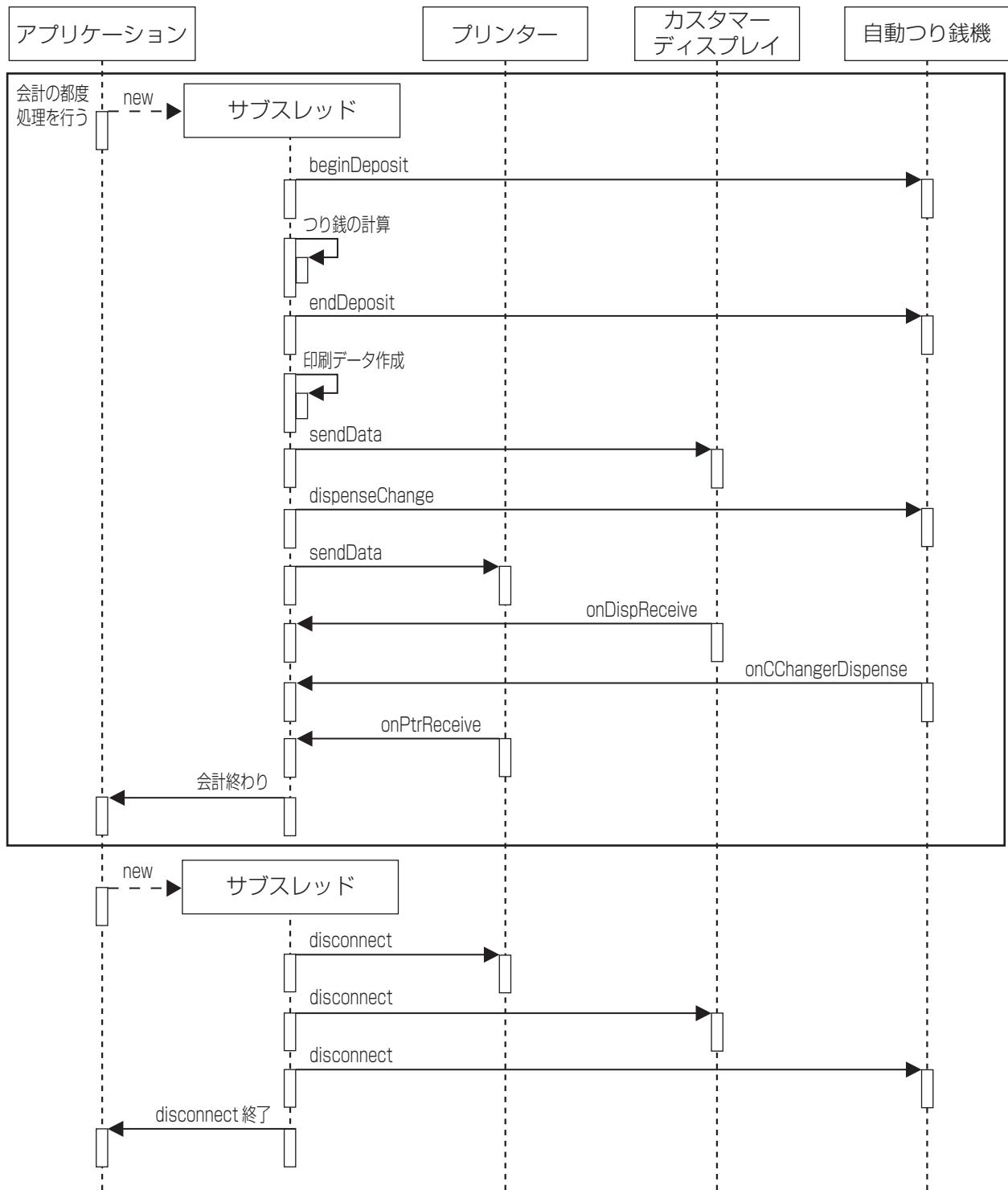
## 複数の周辺機器を制御するには

以下にプリンターとカスタマーディスプレイ、自動つり銭機を制御するシーケンス図を示します。



- ・ プリンターおよび周辺機器との通信を接続/切断する場合は、1つのスレッドから各周辺機器へ順番に connect API/disconnect API を呼び出してください。
- ・ 以下の処理は 1 つのスレッドで並行処理できます。
  - \* カスタマーディスプレイの表示
  - \* つり銭の出金
  - \* レシート印刷





# 付録

## オープンソースソフトウェアライセンス

Epson ePOS SDK for Android が提供するサンプルプログラムとライブラリーには、以下のオープンソースソフトウェアを使用しています。

### サンプルプログラム

#### ZXing(<https://github.com/zxing/zxing>)

ZXing is licensed based on Apache 2.0 license (<http://www.apache.org/licenses/LICENSE-2.0.html>).

### ライブラリー

#### libxml2

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2003 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

**icu4c**

ICU License - ICU 1.8.1 and later  
COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1995-2012 International Business Machines Corporation and others

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

## **zlib**

zlib.h -- interface of the 'zlib' general purpose compression library  
version 1.2.8, April 28th, 2013

Copyright (C) 1995-2013 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly      Mark Adler  
jloup@gzip.org      madler@alumni.caltech.edu

The data format used by the zlib library is described by RFCs (Request for Comments) 1950 to 1952 in the files <http://tools.ietf.org/html/rfc1950> (zlib format), rfc1951 (deflate format) and rfc1952 (gzip format).

## LICENSE ISSUES

---

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts.

### OpenSSL License

---

```
/*
=====
 * Copyright (c) 1998-2016 The OpenSSL Project. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. All advertising materials mentioning features or use of this
 *    software must display the following acknowledgment:
 *    "This product includes software developed by the OpenSSL Project
 *    for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
 *
 * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
 *    endorse or promote products derived from this software without
 *    prior written permission. For written permission, please contact
 *    openssl-core@openssl.org.
 *
 * 5. Products derived from this software may not be called "OpenSSL"
 *    nor may "OpenSSL" appear in their names without prior written
 *    permission of the OpenSSL Project.
 *
 * 6. Redistributions of any form whatsoever must retain the following
 *    acknowledgment:
 *    "This product includes software developed by the OpenSSL Project
 *    for use in the OpenSSL Toolkit (http://www.openssl.org/)"
 *
```

\* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY  
 \* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE  
 \* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR  
 \* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR  
 \* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,  
 \* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT  
 \* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;  
 \* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)  
 \* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,  
 \* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)  
 \* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED  
 \* OF THE POSSIBILITY OF SUCH DAMAGE.  
 \*

---

\*  
 \* This product includes cryptographic software written by Eric Young  
 \* (eay@cryptsoft.com). This product includes software written by Tim  
 \* Hudson (tjh@cryptsoft.com).  
 \*  
 \*/

#### Original SSLeay License

---

```

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
 * All rights reserved.
 *
 * This package is an SSL implementation written
 * by Eric Young (eay@cryptsoft.com).
 * The implementation was written so as to conform with Netscapes SSL.
 *
 * This library is free for commercial and non-commercial use as long as
 * the following conditions are aheared to. The following conditions
 * apply to all code found in this distribution, be it the RC4, RSA,
 * Ihash, DES, etc., code; not just the SSL code. The SSL documentation
 * included with this distribution is covered by the same copyright terms
 * except that the holder is Tim Hudson (tjh@cryptsoft.com).
 *
 * Copyright remains Eric Young's, and as such any Copyright notices in
 * the code are not to be removed.
 * If this package is used in a product, Eric Young should be given attribution
 * as the author of the parts of the library used.
 * This can be in the form of a textual message at program startup or
 * in documentation (online or textual) provided with the package.
 *

```

\* Redistribution and use in source and binary forms, with or without  
\* modification, are permitted provided that the following conditions  
\* are met:  
\* 1. Redistributions of source code must retain the copyright  
\* notice, this list of conditions and the following disclaimer.  
\* 2. Redistributions in binary form must reproduce the above copyright  
\* notice, this list of conditions and the following disclaimer in the  
\* documentation and/or other materials provided with the distribution.  
\* 3. All advertising materials mentioning features or use of this software  
\* must display the following acknowledgement:  
\* "This product includes cryptographic software written by  
\* Eric Young (eay@cryptsoft.com)"  
\* The word 'cryptographic' can be left out if the routines from the library  
\* being used are not cryptographic related :).  
\* 4. If you include any Windows specific code (or a derivative thereof) from  
\* the apps directory (application code) you must include an acknowledgement:  
\* "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"  
\*  
\* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND  
\* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE  
\* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE  
\* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE  
\* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL  
\* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS  
\* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)  
\* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT  
\* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY  
\* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF  
\* SUCH DAMAGE.  
\*  
\* The licence and distribution terms for any publically available version or  
\* derivative of this code cannot be changed. i.e. this code cannot simply be  
\* copied and put under another distribution licence  
\* [including the GNU Public Licence.]  
\*/