SII Print Class Library for Android™
Application Programmer's Guide

Rev.12

[Products]

MP-B20 Series

Seiko Instruments Inc.

Rev.01	March 2017
Rev.02	February 2018
Rev.03	February 2019
Rev.04	August 2019
Rev.05	March 2020
Rev.06	March 2022
Rev.07	October 2022
Rev.08	April 2023
Rev.09	March 2024
Rev.10	January 2025
Rev.11	February 2026
Rev.12	June 2026

Copyright © 2017-2026 by Seiko Instruments Inc. All rights reserved.

Microsoft®, Windows®, and Microsoft Store are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.
Android™ is a trademark of Google LLC.
The Bluetooth® word mark is registered trademarks owned by Bluetooth SIG, Inc.
Oracle and Java® are registered trademarks of Oracle Corporation and/or its affiliates.
The company names and product names appearing on our manual are trademarks or registered trademarks of their respective owners.

Seiko Instruments Inc. (hereinafter referred to as ”SII”) has prepared this manual for use by SII personnel, licensees, and customers. The information contained herein is the property of SII and shall not be reproduced in whole or in part without the prior written approval of SII.

Introduction

This document describes the "SII Print Class Library for Android™" for MP-B20 Series (hereinafter referred to as "SII print class library") provided by Seiko Instruments Inc. (hereinafter referred to as "SII").

Target printers

This section lists the printers supported by SII print class library.

Printers	Interface
MP-B20 Series	Bluetooth
	USB

Terms

This section describes terms used in this manual.

Terms	Description
Printer Command	Command for controlling the printer described in "MP-B20 SERIES THERMAL PRINTER TECHNICAL REFERENCE"

Table of Contents

Chapter 1 Product Overview 1-1

1.1 Functions Provided by SII Print Class Library 1-1
1.2 SII Print Class Library Overview 1-1
1.2.1 SII Print Class Library Configuration 1-1
1.2.2 Functions Provided by Library 1-2

Chapter 2 Product Specifications 2-1

2.1 Operating Environment 2-1
2.2 Operating Conditions 2-2

Chapter 3 How to Use Library 3-1

3.1 Development Environment for Android Application 3-1
3.2 Provided Files 3-2
3.3 Add Library to Android Studio Projects 3-3
3.4 Use Developed Android Application on Android Device 3-5
3.5 Precautions 3-5

Chapter 4 Functions of Library 4-1

4.1 Log File Output Function 4-1
4.1.1 How to Set Log Output 4-1
4.1.2 Log Output Settings 4-1
4.1.3 Log File 4-2
4.2 API Reference 4-3
4.2.1 PrinterManager Class 4-4
(1) Method List 4-4
(2) Constant List 4-5
① Printer model 4-5
② Response type 4-5
③ Battery remaining capacity level 4-6
④ International character set 4-6
⑤ Codepage 4-7
⑥ Port type 4-7
⑦ Barcode or PDF417 4-7
(3) Enumerated Constant List 4-8
① Bold print (CharacterBold) 4-8
② Underline (CharacterUnderline) 4-8
③ Reverse print (CharacterReverse) 4-8
④ Inversion print (CharacterInversion) 4-8
⑤ Character font (CharacterFont) 4-9
⑥ Character scale (CharacterScale) 4-9

⑦ Alignment (PrintAlignment) 4-10
⑧ Pending data output specifying (OutputPendingData) 4-10
⑨ Barcode symbol (BarcodeSymbol) 4-10
⑩ Module size (ModuleSize) 4-11
⑪ HRI character print position (HriPosition) 4-12
⑫ N:W ratio (NwRatio) 4-13
⑬ Error correction level (ErrorCorrection) 4-13
⑭ PDF417 symbol (Pdf417Symbol) 4-13
⑮ QR Code Model (QrModel) 4-13
⑯ Data Matrix module (DataMatrixModule) 4-14
⑰ MaxiCode Mode (MaxiCodeMode) 4-15
⑱ Cutting method (CuttingMethod) 4-15
⑲ Dithering (Dithering) 4-15
⑳ Image rotation direction (Rotate) 4-15
㉑ Image scaling (ImageScale) 4-15
㉒ Batch processing selection (TransactionFunction) 4-16
(4) Method Details 4-17
PrinterManager Constructor 4-17
connect Start communicating with printer (Bluetooth) 4-17
connect Start communicating with printer (USB) 4-18
disconnect Stop communicating with printer 4-18
setBarcodeScannerListener
Start/End callback of barcode scanner 4-18
sendText Send text data 4-19
sendTextEx Send format specified text data 4-19
printBarcode Print barcode 4-20
printPDF417 Print PDF417 4-24
printQRcode Print QR Code 4-25
printDataMatrix Print Data Matrix 4-26
printMaxiCode Print MaxiCode 4-26
printGS1DataBarStacked
Print GS1 Databar Stacked 4-27
printGS1DataBarStackedOmnidirectional
Print GS1 Databar Stacked Omni-directional 4-27
printGS1DataBarExpandedStacked
Print GS1 Databar Expanded Stacked 4-28
printAztecCode Print Aztec Code 4-29
cutPaper Cut paper 4-29
feedPosition Paper form feed 4-29
openDrawer Open cash drawer 4-29
buzzer Sound buzzer 4-29
externalBuzzer Sound external buzzer 4-29
sendBinary Send binary data 4-30
sendDataFile Send specified file 4-30
sendImage Send image 4-31
printPDF Print PDF page 4-32
getStatus Get printer status 4-33

setCallbackFunctionListener
Start/End callback of printer status change....4-34
abort Abort waiting state of printer....4-35
registerLogo Register logo....4-35
printLogo Print logo....4-36
unregisterLogo Delete registered logo....4-36
registerStyleSheet Register style sheet....4-36
unregisterStyleSheet Delete registered style sheet....4-37
resetPrinter Reset printer....4-37
getPrinterResponse Get various responses from printer....4-37
startDiscoveryPrinter Start printer search (Bluetooth)....4-39
startDiscoveryPrinter Start printer search (USB)....4-39
startDiscoveryPrinter Start printer search (TCP/IP)....4-40
cancelDiscoveryPrinter Cancel printer search....4-40
getFoundPrinter Get found printer information....4-40
getSendTimeout Get send timeout period....4-40
setSendTimeout Set send timeout period....4-40
getReceiveTimeout Get receive timeout period....4-41
setReceiveTimeout Set receive timeout period....4-41
getInternationalCharacter
Get international character set....4-41
setInternationalCharacter
Set international character set....4-41
getCodePage Get codepage....4-42
setCodePage Set codepage....4-42
getPrinterModel Get printer model....4-42
getPortType Get connecting port type....4-42
isConnect Verify connection state with printer....4-43
getSocketKeepingTime Get socket keeping time....4-43
setSocketKeepingTime Set socket keeping time....4-43
getVersion Get SDK version....4-43
controlTransaction Start/End batch processing....4-43
4.2.2 PrinterEvent Class....4-45
(1) Method List....4-45
(2) End event constant....4-45
(3) Method Details....4-45
getEventType Get end event....4-45
4.2.3 PrinterListener Interface....4-46
(1) Method List....4-46
(2) Method Details....4-46
finishEvent End event of the printer search....4-46
4.2.4 PrinterInfo Class....4-47
(1) Method List....4-47
(2) Method Details....4-47
getPrinterModelName Get printer model name....4-47
getBluetoothAddress Get Bluetooth address....4-47
getMacAddress Get MAC address....4-47
getIsBonded Get pairing status....4-47

getDevicePath Get device path 4-48
4.2.5 PrinterIdentifier Class 4-49
(1) Method List 4-49
(2) Method Details 4-49
getPrinterModelID Get printer model ID 4-49
getTypeID Get type ID 4-49
getRomVersionID Get ROM version ID 4-49
getMainFirmwareVersion
Get firmware version (main) 4-50
getManufacturer Get manufacturer 4-50
getModelName Get model name 4-50
getSerialNumber Get serial number 4-50
getBootFirmwareVersion
Get firmware version (boot) 4-50
getBootFirmwareChecksum
Get firmware checksum (boot) 4-51
getMainFirmwareChecksum
Get firmware checksum (main) 4-51
getFirmwareChecksum
Get firmware checksum (main + boot) 4-51
4.2.6 PrinterException Class 4-52
(1) Method List 4-52
(2) Constant List 4-52
① Error code 4-52
(3) Method Details 4-53
PrinterException Constructor 4-53
getErrorCode Get error codes 4-53
4.2.7 CallbackFunctionListener Interface 4-54
(1) Method List 4-54
(2) Method Details 4-54
onStatusChanged Change event of printer status 4-54
4.2.8 BarcodeScannerListener Interface 4-55
4.2.9 SmartLabelManager Class 4-56

Chapter 5 Sample Program 5-1

5.1 Screen 5-1
5.1.1 Main Screen 5-1
5.1.2 [Settings] Screen 5-3
5.2 Precaution 5-3

Chapter 6 Disclaimer 6-1

Appendix A Character Set A-1

A.1 Code Page Table (Character Code Table) A-1
A.2 International Character Set A-11

Appendix B Barcode Size List B-1

B.1 Barcode Size List B-1
B.1.1 printBarcode B-1
B.1.2 printPDF417 ..... B-7
B.1.3 printQRCode B-8
B.1.4 printDataMatrix B-9
B.1.5 printMaxicode B-11
B.1.6 printGS1DataBarStacked B-12
B.1.7 printGS1DataBarStackedOmnidirectional B-13
B.1.8 printGS1DataBarExpandedStacked B-14

Chapter 1
Product Overview

This chapter describes the product overview of SII print class library.

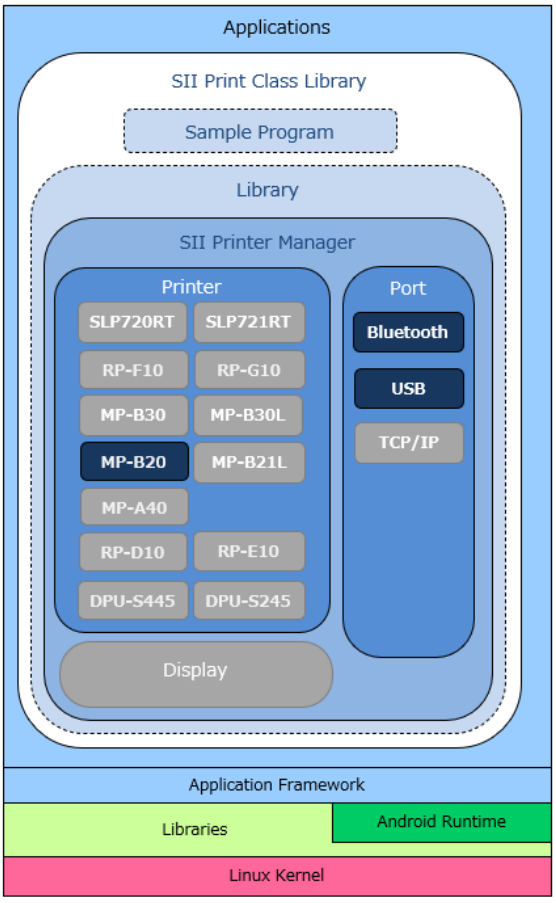
1.1 Functions Provided by SII Print Class Library

The SII print class library including the library and the sample programs provides the functions to use SII printer MP-B20 Series (hereinafter referred to as "printer") in Android applications.
Moreover, the SII print class library provides Android Studio projects as a sample program for SII print class library.

1.2 SII Print Class Library Overview

1.2.1 SII Print Class Library Configuration

The library and sample program in SII print class library are indicated with dashed lines in the figure below.



1.2.2 Functions Provided by Library

By using the library, Android applications can easily send print data and printer commands to a printer through communication port (Bluetooth or USB) on an Android device. Also, the applications can get printer status.

The library provides the following functions.

- Connecting to / disconnecting from a printer
- Sending data to a printer (print data and/or printer commands*1)
- Printing barcode and 2-dimensional barcode
- Sending a data file to a printer (print data and/or printer commands*1)
- Getting the printer status
- Aborting the waiting state of a printer
- Getting various responses from a printer
- Bulk registration of print commands
- Registering a printer status call back function
- Searching the printer by Bluetooth
- Outputting a log file

*1: Commands that read responses from the printer are not available. In order to read responses from a printer, use **getStatus** or **getPrinterResponse**.

(NOTE) MP-B20 does not support the APIs relating to page mode, Display, the barcode scanner, or label printing function.

Chapter 2
Product Specifications

This chapter describes the product specifications of the library.

2.1 Operating Environment

Operating environment for the library is shown in the following table.

<table>
<tr><td rowspan="3">Printer</td><td colspan="2">Model</td><td colspan="2">MP-B20</td></tr>
<tr><td colspan="2">F/W Version</td><td colspan="2">1.00 or later</td></tr>
<tr><td colspan="2">Communication Interface</td><td>Bluetooth</td><td>USB</td></tr>
<tr><td rowspan="15">Android Device</td><td colspan="2">Communication Port</td><td>Bluetooth*1</td><td>USB*2</td></tr>
<tr><td rowspan="14">OS</td><td>Android 7.0 (API 24)</td><td rowspan="14">Supported</td><td rowspan="14">Supported</td></tr>
<tr><td>Android 7.1 (API 25)</td></tr>
<tr><td>Android 8.0 (API 26)</td></tr>
<tr><td>Android 8.1 (API 27)</td></tr>
<tr><td>Android 9.0 (API 28)</td></tr>
<tr><td>Android 10.0 (API 29)</td></tr>
<tr><td>Android 11.0 (API 30)</td></tr>
<tr><td>Android 12.0 (API 31)</td></tr>
<tr><td>Android 12.1 (API 32)</td></tr>
<tr><td>Android 13.0 (API 33)</td></tr>
<tr><td>Android 14.0 (API 34)</td></tr>
<tr><td>Android 15.0 (API 35)</td></tr>
<tr><td>Android 16.0 (API 36)</td></tr>
<tr><td colspan="3">Supported Language</td><td colspan="2">Japanese, English</td></tr>
</table>

*1: Bluetooth connection needs to be established by SPP (Serial Port Profile).
*2: Android device needs to support USB host function.

2.2 Operating Conditions

This section describes the operating conditions for the library.
Set the Function Setting of the printer from [value] in the following table before using the library.
See "MP-B20 SERIES Thermal Printer USER'S GUIDE" for details about Function Setting and the factory default settings.

MS	Function	Value
1-1	Interface Selection (Interface)	0 : USB 1 : Wireless
3-1	Automatic Status Response Selection (Auto Status Back)	0 : Enable
3-2	Initialized Response Selection (Init. Response)	0 : Enable
3-3	Realtime Command Selection (Realtime Command)	0 : Enable
3-4	Data Discard Selection When Error Occurs (Error Through)	0 : Enable
3-5	Data Discard Selection When Output Buffer Full Occurs (Response Data Discarding)	1 : Disable

Chapter 3
How to Use Library

This chapter describes the development environment for Android application and how to use the library.

3.1 Development Environment for Android Application

In order to develop Android applications, the following tools are required. See each of the following URLs for more details.

- Android Studio

  https://developer.android.com/studio/index.html

- USB driver for Windows (When develop on Windows environment)

  https://developer.android.com/studio/run/oem-usb.html

In this chapter and after in this document, it is required to set up an environment where each tool is available.

3.2 Provided Files

The file configuration of SII print class library is as follows.

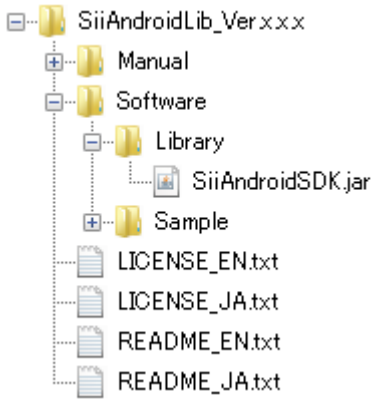


Figure 3-1

The file format of the library is jar. The file name of the library is SiiAndroidSDK.jar.

3.3 Add Library to Android Studio Projects

This section describes how to add the SII print class library to Android Studio projects.

See "Chapter 5 Sample Program" for sample programs included in SII print class library.

(1) Create a project in Android Studio and copy the library file (SiiAndroidSDK.jar) to the [libs] folder. When the [libs] folder is not automatically created, add the folder manually.
For sample programs included in the SII print class library, the folder is "\Sample\app\libs".

(2) After adding the library, the view looks like Figure 3-2.

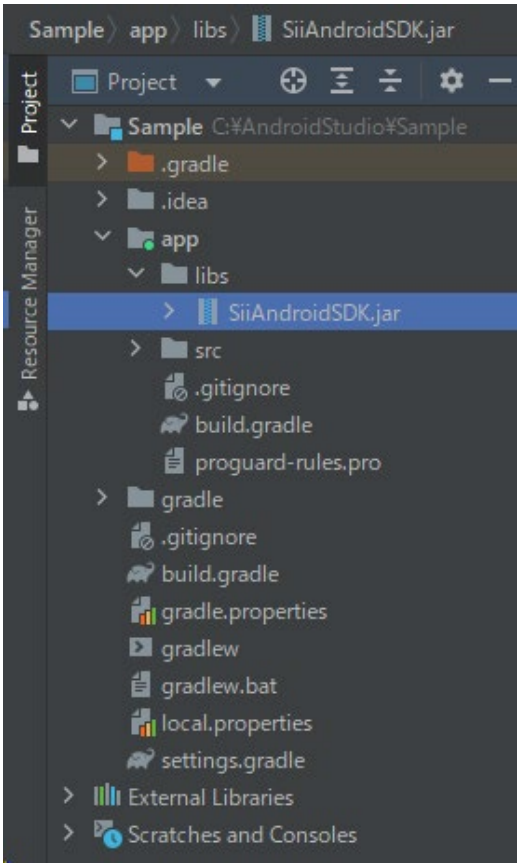


Figure 3-2

(3) Add the following to dependencies{} of build.gradle(:app) in the application.

```
implementation 'com.journeyapps:zxing-android-embedded:3.4.0@aar'
implementation 'com.google.zxing:core:3.4.1'
implementation files ('libs/SiiAndroidSDK.jar')
```

(4) Add the following to the beginning of the class file that uses the print class library.
(Import xxxx according to the function to use.)

```
import com.seikoinstruments.sdk.thermalprinter.PrinterManager;
import com.seikoinstruments.sdk.thermalprinter.xxxx;
```

(5) Add the following permission declaration to the application manifest (AndroidManifest.xml).
Also, implement a process (requestPermissions()) that requests the appropriate permissions for the application.

[When using Bluetooth]

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

By completing these procedures, functions of the library become available.

3.4 Use Developed Android Application on Android Device

In order to use the developed Android applications on the Android device, configure the following settings on the Android device.

(1) Select [Settings], [Developer options], and turn on [USB debugging]. (Figure 3-3)

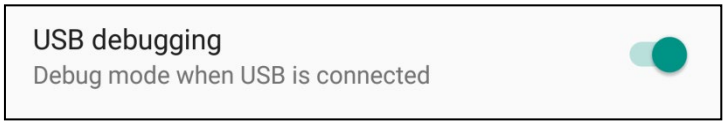


Figure 3-3

3.5 Precautions

- **About Scoped Storage**

  "Scoped Storage" that is introduced in Android 10 distinguishes between app-specific storage and external storage.
  When targeting Android 10 (API 29) or later, files that do not correspond to media files in the external storage cannot be handled directly. Files that do not correspond to media files can be handled by using the "Storage Access Framework".
  See below for details of Scoped Storage.

  - Data and file storage overview
    https://developer.android.com/training/data-storage

Chapter 4
Functions of the Library

This chapter describes the APIs of each class implemented in the library.

4.1 Log File Output Function

The logs can be retrieved and the log files can be output using the library.

4.1.1 How to Set Log Output

Log output settings can be configured by adding the config.ini file with the following content to the specific directory of the Android application that incorporates the library (e.g. internal shared storage \Android\data\<package name>\files).

config.ini

```
LOGLEVEL=x
LOGSIZEMAX=xMB
LOGOUTPUT=x
```

Reference: See "4.1.2 Log Output Settings" for details on the settings for x.

4.1.2 Log Output Settings

Item	Description	Settings
LOGLEVEL	Log level	0 : Not record the log. 1 : Records an error log when **PrinterException** is thrown. 2 : Records API execution history.
LOGSIZEMAX	Log file maximum size	1MB : Log file maximum size is 1 MB 5MB : Log file maximum size is 5 MB 10MB : Log file maximum size is 10 MB 50MB : Log file maximum size is 50 MB
LOGOUTPUT	Logcat output enabled/disabled	0 : Logcat output is disabled 1 : Logcat output is enabled

4.1.3 Log File

Log files are saved as local files in the Android application that incorporates the library.

Log file name : PrinterManagerX.log (range of X is 0 to 4)

The 1st log file is created as PrinterManager0.log. If the log file maximum size is exceeded, changes the file name to PrinterManager1.log and creates a new PrinterManager0.log.

Up to 5 log files can be created.

4.2 API Reference

The package of the library is **com.seikoinstruments.sdk.thermalprinter**.
com.seikoinstruments.sdk.thermalprinter includes the following classes.

Class Name	Description	Supported *1
PrinterManager	Provides the API used for communication with the printer and for printing. See "**4.2.1 PrinterManager Class**" for more details.	✓
PrinterEvent	Provides the API that gets the end event when **startDiscoveryPrinter** is terminated. See "**4.2.2 PrinterEvent Class**" for more details.	✓
PrinterListener	Interface for getting the end event when **startDiscoveryPrinter** is terminated. See "**4.2.3 PrinterListener Interface**" for more details.	✓
PrinterInfo	Stores the printer information found by **startDiscoveryPrinter**. See "**4.2.4 PrinterInfo Class**".	✓
PrinterIdentifier	Stores the printer ID of the printer retrieved by **getPrinterResponse**. See "**4.2.5 PrinterIdentifier Class**".	✓
PrinterException	Exception class that is thrown at API call. See "**4.2.6 PrinterException Class**" for more details.	✓
CallbackFunctionListener	Interface for getting the change event of printer status. See "**4.2.7 CallbackFunctionListener Interface**" for more details.	✓
BarcodeScannerListener	Interface for getting barcode scanner connection or barcode scanner disconnection, or received barcode data.	-
SmartLabelManager	Provides the API to specify label files or replace data.	-

*1: ✓: Supported, -: Not supported in MP-B20

(NOTE) MP-B20 does not support the APIs relating to page mode, Display, the barcode scanner, or label printing function.

4.2.1 PrinterManager Class

(1) Method List

Methods provided by the **PrinterManager** class are shown in the following table.

Name	Description	Supported *1
PrinterManager	Constructor	✓
connect	Start communicating with printer (Bluetooth)	✓
connect	Start communicating with printer (USB)	✓
disconnect	Stop communicating with printer	✓
setBarcodeScannerListener	Start/End callback of barcode scanner	-
sendText	Send text data	✓
sendTextEx	Send format specified text data	✓
printBarcode	Print barcode	✓
printPDF417	Print PDF417	✓
printQRcode	Print QR Code	✓
printDataMatrix	Print Data Matrix	✓
printMaxiCode	Print MaxiCode	✓
printGS1DataBarStacked	Print GS1 Databar Stacked	✓
printGS1DataBarStackedOmnidirectional	Print GS1 Databar Stacked Omni-directional	✓
printGS1DataBarExpandedStacked	Print GS1 Databar Expanded Stacked	✓
printAztecCode	Print Aztec Code	-
cutPaper	Cut paper*2	✓
feedPosition	Paper form feed	-
openDrawer	Open cash drawer	-
buzzer	Sound buzzer	-
externalBuzzer	Sound external buzzer	-
sendBinary	Send binary data	✓
sendDataFile	Send specified file	✓
sendImage	Send image	✓
printPDF	Print PDF page	✓
getStatus	Get printer status	✓
setCallbackFunctionListener	Start/End callback of printer status change	✓
abort	Abort waiting state of printer	✓
registerLogo	Register logo	✓
printLogo	Print logo	✓
unregisterLogo	Delete registered logo	✓
registerStyleSheet	Register style sheet	-
unregisterStyleSheet	Delete registered style sheet	-
resetPrinter	Reset printer	✓
getPrinterResponse	Get various responses from printer	✓

Name	Description	Supported *1
startDiscoveryPrinter	Start printer search (Bluetooth)	✓
startDiscoveryPrinter	Start printer search (USB)	✓
startDiscoveryPrinter	Start printer search (TCP/IP)	-
cancelDiscoveryPrinter	Cancel printer search	✓
getFoundPrinter	Get found printer information	✓
getSendTimeout	Get send timeout period	✓
setSendTimeout	Set send timeout period	✓
getReceiveTimeout	Get receive timeout period	✓
setReceiveTimeout	Set receive timeout period	✓
getInternationalCharacter	Get international character set	✓
setInternationalCharacter	Set international character set	✓
getCodePage	Get codepage	✓
setCodePage	Set codepage	✓
getPrinterModel	Get printer model	✓
getPortType	Get connecting port type	✓
isConnect	Verify connection state with printer	✓
getSocketKeepingTime	Get socket keeping time	-
setSocketKeepingTime	Set socket keeping time	-
getVersion	Get SDK version	✓
controlTransaction	Start/End batch processing	✓

*1: ✓: Supported, -: Not supported in MP-B20
*2: Only the paper feed operation to the paper cut position is performed.

(2) Constant List

① Printer model

Constants used for starting communicating with a printer and getting the printer model are shown in the following table.

Constant Name	Description	Value
PRINTER_MODEL_MP_B20	MP-B20	298
PRINTER_MODEL_DEFAULT	Default value of printer model	284

② Response type

Constants used for getting various responses from a printer are shown in the following table.

Constant Name	Description	Value
PRINTER_RESPONSE_REQUEST	Request of execution response	0
PRINTER_RESPONSE_USER_AREA	Send remaining capacity response in user area	1

Constant Name	Description	Value
PRINTER_RESPONSE_ARRANGE_USER_AREA	Send remaining capacity response in user area after defragment	2
PRINTER_RESPONSE_NV_GRAPHICS	Send NV graphics memory capacity	3
PRINTER_RESPONSE_KEY_CODE	Send key code list of defined NV graphics	4
PRINTER_RESPONSE_BATTERY_STATUS	Battery remaining capacity level	5
PRINTER_RESPONSE_MAINTENANCE_COUNTER	Send maintenance counter	7
PRINTER_RESPONSE_PRINTER_IDENTIFIER	Send printer ID	8

③ Battery remaining capacity level

Constants of the battery remaining capacity level retrieved from a printer are shown in the following table.

Constant Name	Description	Value
BATTERY_STATUS_FULL	Battery remaining capacity: approx. 80%	0
BATTERY_STATUS_MIDDLE	Battery remaining capacity: approx. 40%	1
BATTERY_STATUS_LOW	Battery remaining capacity: approx. 10%	2
BATTERY_STATUS_EMPTY	No battery	3

④ International character set

Constants used for setting/getting international character set are shown in the following table.

Constant Name	Description	Value
COUNTRY_USA	USA	0
COUNTRY_FRANCE	France	1
COUNTRY_GERMANY	Germany	2
COUNTRY_ENGLAND	United Kingdom	3
COUNTRY_DENMARK_1	Denmark I	4
COUNTRY_SWEDEN	Sweden	5
COUNTRY_ITALY	Italy	6
COUNTRY_SPAIN	Spain I	7
COUNTRY_JAPAN	Japan	8
COUNTRY_NORWAY	Norway	9
COUNTRY_DENMARK_2	Denmark II	10
COUNTRY_SPAIN_2	Spain II	11
COUNTRY_LATIN_AMERICA	Latin America	12
COUNTRY_ARABIA	Arabia	17

⑤ Codepage

Constants used for setting/getting codepage are shown in the following table.

Constant Name	Codepage	Value
CODE_PAGE_437	USA, Standard Europe (Code Page 437)	0
CODE_PAGE_KATAKANA	Katakana	1
CODE_PAGE_850	Multilingual (Code Page 850)	2
CODE_PAGE_860	Portuguese (Code Page 860)	3
CODE_PAGE_863	Canadian-French (Code Page 863)	4
CODE_PAGE_865	Nordic (Code Page 865)	5
CODE_PAGE_857	Turkish (Code Page 857)	13
CODE_PAGE_737	Greek (Code Page 737)	14
CODE_PAGE_1252	Latin (Code Page 1252)	16
CODE_PAGE_866	Russian (Code Page 866)	17
CODE_PAGE_852	Eastern Europe (Code Page 852)	18
CODE_PAGE_858	Euro (Code Page 858)	19
CODE_PAGE_855	Cyrillic (Code Page 855)	34
CODE_PAGE_864[*1*2]	Arabic (Code Page 864)	37
CODE_PAGE_1250	Central European (Code Page 1250)	45
CODE_PAGE_1251	Cyrillic (Code Page 1251)	46
CODE_PAGE_1253	Greek (Code Page 1253)	47
CODE_PAGE_1254	Turkish (Code Page 1254)	48

*1: 20ACh of the Unicode cannot be printed.
*2: Font B cannot be printed.

⑥ Port type

Constants used for starting communicating with a printer and getting the connecting port type are shown in the following table.

Constant Name	Description	Value
PRINTER_TYPE_BLUETOOTH	Bluetooth	0
PRINTER_TYPE_USB	USB	1

⑦ Barcode or PDF417

Constants used for printing barcode and printing PDF417 are shown in the following table.

Constant Name	Description	Value
BARCODE_HEIGHT_DEFAULT	Default value of barcode height	162
PDF417_MODULE_HEIGHT_DEFAULT	Default value of PDF417 height	10
PDF417_ROW_AUTO	Automatic selection of the number of rows	0

Constant Name	Description	Value
PDF417_COLUMN_AUTO	Automatic selection of the number of columns	0

(3) Enumerated Constant List

① Bold print (CharacterBold)

Constants of enumerated type used for bold print are shown in the following table.

Constant Name	Description
BOLD_CANCEL	Cancel bold print
BOLD	Specify bold print

② Underline (CharacterUnderline)

Constants of enumerated type used for underlining are shown in the following table.

Constant Name	Description
UNDERLINE_CANCEL	Cancel underline print
UNDERLINE_1	Specify 1-dot width underline print
UNDERLINE_2	Specify 2-dot width underline print

③ Reverse print (CharacterReverse)

Constants of enumerated type used for reverse print are shown in the following table.

Constant Name	Description
REVERSE_CANCEL	Cancel reverse print
REVERSE	Specify reverse print

④ Inversion print (CharacterInversion)

Constants of enumerated type used for inversion print are shown in the following table.
Inversion print cannot be added to the text data before inserting a new line feed.

Constant Name	Description
INVERSION_CANCEL	Cancel inversion print
INVERSION	Specify inversion print

⑤ Character font (CharacterFont)

Constants of enumerated type used for character font are shown in the following table.

Constant Name	Description
FONT_A	Font A (24 × 12)
FONT_B	Font B (16 × 8)

⑥ Character scale (CharacterScale)

Constants of enumerated type used for character scale are shown in the following table.

Constant Name	Description
VERTICAL_1_HORIZONTAL_1	Height × 1 and width × 1
VERTICAL_1_HORIZONTAL_2	Height × 1 and width × 2
VERTICAL_1_HORIZONTAL_3	Height × 1 and width × 3
VERTICAL_1_HORIZONTAL_4	Height × 1 and width × 4
VERTICAL_2_HORIZONTAL_1	Height × 2 and width × 1
VERTICAL_2_HORIZONTAL_2	Height × 2 and width × 2
VERTICAL_2_HORIZONTAL_3	Height × 2 and width × 3
VERTICAL_2_HORIZONTAL_4	Height × 2 and width × 4
VERTICAL_2_HORIZONTAL_6	Height × 2 and width × 6
VERTICAL_3_HORIZONTAL_1	Height × 3 and width × 1
VERTICAL_3_HORIZONTAL_2	Height × 3 and width × 2
VERTICAL_3_HORIZONTAL_3	Height × 3 and width × 3
VERTICAL_3_HORIZONTAL_4	Height × 3 and width × 4
VERTICAL_4_HORIZONTAL_1	Height × 4 and width × 1
VERTICAL_4_HORIZONTAL_2	Height × 4 and width × 2
VERTICAL_4_HORIZONTAL_3	Height × 4 and width × 3
VERTICAL_4_HORIZONTAL_4	Height × 4 and width × 4
VERTICAL_4_HORIZONTAL_6	Height × 4 and width × 6
VERTICAL_4_HORIZONTAL_8	Height × 4 and width × 8
VERTICAL_6_HORIZONTAL_2	Height × 6 and width × 2
VERTICAL_6_HORIZONTAL_4	Height × 6 and width × 4
VERTICAL_6_HORIZONTAL_6	Height × 6 and width × 6
VERTICAL_6_HORIZONTAL_8	Height × 6 and width × 8
VERTICAL_8_HORIZONTAL_4	Height × 8 and width × 4
VERTICAL_8_HORIZONTAL_6	Height × 8 and width × 6
VERTICAL_8_HORIZONTAL_8	Height × 8 and width × 8

⑦ Alignment (PrintAlignment)

Constants of enumerated type used for alignment are shown in the following table.
Alignment cannot be added to the text data before inserting a new line feed.

Constant Name	Description
ALIGNMENT_LEFT	Aligned left
ALIGNMENT_CENTER	Centered
ALIGNMENT_RIGHT	Aligned right

⑧ Pending data output specifying (OutputPendingData)

Constants of enumerated type used for pending data output specifying are shown in the following table.

Constant Name	Description
PENDING_DATA_OUTPUT_FIRST	Output pending data at first and start the processing
PENDING_DATA_OUTPUT_TOGETHER	Output pending data at the same time as the processing

⑨ Barcode symbol (BarcodeSymbol)

Constants of enumerated type used for barcode symbol are shown in the following table.

Constant Name	Description	Syntax*1
BARCODE_SYMBOL_UPC_A	UPC-A	(a)
BARCODE_SYMBOL_UPC_E	UPC-E	(a)
BARCODE_SYMBOL_EAN13	EAN13	(a)
BARCODE_SYMBOL_JAN13	JAN13	(a)
BARCODE_SYMBOL_EAN8	EAN8	(a)
BARCODE_SYMBOL_JAN8	JAN8	(a)
BARCODE_SYMBOL_CODE39	CODE39	(a), (b)
BARCODE_SYMBOL_CODE93	CODE93	(c)
BARCODE_SYMBOL_CODE128	CODE128	(c)
BARCODE_SYMBOL_ITF	ITF	(a), (b)
BARCODE_SYMBOL_CODABAR	CODABAR	(a), (b)
BARCODE_SYMBOL_EAN13_ADDON	EAN13 add-on	(a)
BARCODE_SYMBOL_JAN13_ADDON	JAN13 add-on	(a)
BARCODE_SYMBOL_GS1_OMNI_DIRECTIONAL	GS1 Databar Omni-directional	(a)
BARCODE_SYMBOL_GS1_TRUNCATED	GS1 Databar Truncated	(a)
BARCODE_SYMBOL_GS1_LIMITED	GS1 Databar Limited	(a)
BARCODE_SYMBOL_GS1_EXPANDED	GS1 Databar Expanded	(a)

*1: See "4.2.1(4) Method Details **printBarcode**" for more details of syntax.

⑩ Module size (ModuleSize)

Constants of enumerated type used for width, nominal fine element width, and module size of barcode are shown in the following table.

Constant Name	Description	Method to Use
BARCODE_MODULE_WIDTH_2	Fine element 2 dots Module width 0.250 mm	**printBarcode**
BARCODE_MODULE_WIDTH_3	Fine element 3 dots Module width 0.375 mm	
BARCODE_MODULE_WIDTH_4	Fine element 4 dots Module width 0.500 mm	
BARCODE_MODULE_WIDTH_5	Fine element 5 dots Module width 0.625 mm	
BARCODE_MODULE_WIDTH_6	Fine element 6 dots Module width 0.750 mm	
PDF417_MODULE_WIDTH_2	Nominal fine element width 2 dots	**printPDF417**
PDF417_MODULE_WIDTH_3	Nominal fine element width 3 dots	
PDF417_MODULE_WIDTH_4	Nominal fine element width 4 dots	
PDF417_MODULE_WIDTH_5	Nominal fine element width 5 dots	
PDF417_MODULE_WIDTH_6	Nominal fine element width 6 dots	
PDF417_MODULE_WIDTH_7	Nominal fine element width 7 dots	
PDF417_MODULE_WIDTH_8	Nominal fine element width 8 dots	
QR_MODULE_SIZE_2	2 dots	**printQRcode**
QR_MODULE_SIZE_3	3 dots	
QR_MODULE_SIZE_4	4 dots	
QR_MODULE_SIZE_5	5 dots	
QR_MODULE_SIZE_6	6 dots	
QR_MODULE_SIZE_7	7 dots	
QR_MODULE_SIZE_8	8 dots	
QR_MODULE_SIZE_9	9 dots	
QR_MODULE_SIZE_10	10 dots	
QR_MODULE_SIZE_11	11 dots	
QR_MODULE_SIZE_12	12 dots	
QR_MODULE_SIZE_13	13 dots	
QR_MODULE_SIZE_14	14 dots	
QR_MODULE_SIZE_15	15 dots	
QR_MODULE_SIZE_16	16 dots	

Constant Name	Description	Method to Use
DATAMATRIX_MODULE_SIZE_2	2 dots	**printDataMatrix**
DATAMATRIX_MODULE_SIZE_3	3 dots	
DATAMATRIX_MODULE_SIZE_4	4 dots	
DATAMATRIX_MODULE_SIZE_5	5 dots	
DATAMATRIX_MODULE_SIZE_6	6 dots	
DATAMATRIX_MODULE_SIZE_7	7 dots	
DATAMATRIX_MODULE_SIZE_8	8 dots	
DATAMATRIX_MODULE_SIZE_9	9 dots	
DATAMATRIX_MODULE_SIZE_10	10 dots	
DATAMATRIX_MODULE_SIZE_11	11 dots	
DATAMATRIX_MODULE_SIZE_12	12 dots	
DATAMATRIX_MODULE_SIZE_13	13 dots	
DATAMATRIX_MODULE_SIZE_14	14 dots	
DATAMATRIX_MODULE_SIZE_15	15 dots	
DATAMATRIX_MODULE_SIZE_16	16 dots	
GS1DATABAR_MODULE_SIZE_2	2 dots	● **printGS1DataBarStacked** ● **printGS1DataBarStackedOmnidirectional** ● **printGS1DataBarExpandedStacked**
GS1DATABAR_MODULE_SIZE_3	3 dots	
GS1DATABAR_MODULE_SIZE_4	4 dots	
GS1DATABAR_MODULE_SIZE_5	5 dots	
GS1DATABAR_MODULE_SIZE_6	6 dots	
GS1DATABAR_MODULE_SIZE_7	7 dots	
GS1DATABAR_MODULE_SIZE_8	8 dots	
GS1DATABAR_MODULE_SIZE_9	9 dots	
GS1DATABAR_MODULE_SIZE_10	10 dots	
GS1DATABAR_MODULE_SIZE_11	11 dots	
GS1DATABAR_MODULE_SIZE_12	12 dots	
GS1DATABAR_MODULE_SIZE_13	13 dots	
GS1DATABAR_MODULE_SIZE_14	14 dots	
GS1DATABAR_MODULE_SIZE_15	15 dots	
GS1DATABAR_MODULE_SIZE_16	16 dots	

⑪ HRI character print position (HriPosition)

Constants of enumerated type used for HRI character print position are shown in the following table.

Constant Name	Description
HRI_NONE	Not printed
HRI_POSITION_ABOVE	Above barcode
HRI_POSITION_BELOW	Below barcode
HRI_POSITION_ABOVE_BELOW	Above and below barcode (both)

⑫ N:W ratio (NwRatio)

Constants of enumerated type used for N:W ratio are shown in the following table.

Constant Name	Description
NWRATIO_1TO2	1:2
NWRATIO_1TO2_5	1:2.5
NWRATIO_1TO3	1:3

⑬ Error correction level (ErrorCorrection)

Constants of enumerated type used for error correction level are shown in the following table.

Constant Name	Description	Method to Use
PDF417_ERROR_CORRECTION_0	Error correction level 0	**printPDF417**
PDF417_ERROR_CORRECTION_1	Error correction level 1	
PDF417_ERROR_CORRECTION_2	Error correction level 2	
PDF417_ERROR_CORRECTION_3	Error correction level 3	
PDF417_ERROR_CORRECTION_4	Error correction level 4	
PDF417_ERROR_CORRECTION_5	Error correction level 5	
PDF417_ERROR_CORRECTION_6	Error correction level 6	
PDF417_ERROR_CORRECTION_7	Error correction level 7	
PDF417_ERROR_CORRECTION_8	Error correction level 8	
QR_ERROR_CORRECTION_L	Error correction level L	**printQRcode**
QR_ERROR_CORRECTION_M	Error correction level M	
QR_ERROR_CORRECTION_H	Error correction level H	
QR_ERROR_CORRECTION_Q	Error correction level Q	

⑭ PDF417 symbol (Pdf417Symbol)

Constants of enumerated type used for PDF417 symbol are shown in the following table.

Constant Name	Description
PDF417_STANDARD	PDF417
PDF417_COMPACT	Compact PDF417

⑮ QR Code Model (QrModel)

Constants of enumerated type used for QR Code Model are shown in the following table.

Constant Name	Description
QR_MODEL_1	QR Code Model 1
QR_MODEL_2	QR Code Model 2

⑯ Data Matrix module (DataMatrixModule)

Constants of enumerated type used for Data Matrix module are shown in the following table.

Constant Name	Description
DATA_MATRIX_AUTO	Number of modules: Automatic
DATA_MATRIX_10_10	Number of modules: 10 × 10
DATA_MATRIX_12_12	Number of modules: 12 × 12
DATA_MATRIX_14_14	Number of modules: 14 × 14
DATA_MATRIX_16_16	Number of modules: 16 × 16
DATA_MATRIX_18_18	Number of modules: 18 × 18
DATA_MATRIX_20_20	Number of modules: 20 × 20
DATA_MATRIX_22_22	Number of modules: 22 × 22
DATA_MATRIX_24_24	Number of modules: 24 × 24
DATA_MATRIX_26_26	Number of modules: 26 × 26
DATA_MATRIX_32_32	Number of modules: 32 × 32
DATA_MATRIX_36_36	Number of modules: 36 × 36
DATA_MATRIX_40_40	Number of modules: 40 × 40
DATA_MATRIX_44_44	Number of modules: 44 × 44
DATA_MATRIX_48_48	Number of modules: 48 × 48
DATA_MATRIX_52_52	Number of modules: 52 × 52
DATA_MATRIX_64_64	Number of modules: 64 × 64
DATA_MATRIX_72_72	Number of modules: 72 × 72
DATA_MATRIX_80_80	Number of modules: 80 × 80
DATA_MATRIX_88_88	Number of modules: 88 × 88
DATA_MATRIX_96_96	Number of modules: 96 × 96
DATA_MATRIX_104_104	Number of modules: 104 × 104
DATA_MATRIX_120_120	Number of modules: 120 × 120
DATA_MATRIX_132_132	Number of modules: 132 × 132
DATA_MATRIX_144_144	Number of modules: 144 × 144
DATA_MATRIX_8_18	Number of modules: 8 × 18
DATA_MATRIX_8_32	Number of modules: 8 × 32
DATA_MATRIX_12_26	Number of modules: 12 × 26
DATA_MATRIX_12_36	Number of modules: 12 × 36
DATA_MATRIX_16_36	Number of modules: 16 × 36
DATA_MATRIX_16_48	Number of modules: 16 × 48

⑰ MaxiCode Mode (MaxiCodeMode)

Constants of enumerated type used for MaxiCode Mode are shown in the following table.

Constant Name	Description
MAXI_CODE_2	Mode2
MAXI_CODE_3	Mode3
MAXI_CODE_4	Mode4
MAXI_CODE_5	Mode5

⑱ Cutting method (CuttingMethod)

Constants of enumerated type used for cutting method are shown in the following table.

Constant Name	Description
CUT_FULL	No cut Paper feed to cut position
CUT_PARTIAL	

⑲ Dithering (Dithering)

Constants of enumerated type used for dithering are shown in the following table.

Constant Name	Description
DITHERING_DISABLE	Dithering is disabled
DITHERING_ERRORDIFFUSION	Dithering is enabled

⑳ Image rotation direction (Rotate)

Constants of enumerated type used for image rotation direction are shown in the following table.

Constant Name	Description
ROTATE_NONE	No rotation
ROTATE_180	Rotate 180 degrees

㉑ Image scaling (ImageScale)

Constants of enumerated type used for image scaling are shown in the following table.

Constant Name	Description
IMAGE_SCALE_NONE	No scaling
IMAGE_SCALE_WIDTH_FIT	Scale to fit print width

㉒ Batch processing selection (TransactionFunction)

Constants of enumerated type used for batch processing selection are shown in the following table.

Constant Name	Description
TRANSACTION_CLEAR	Cancel batch processing
TRANSACTION_START	Start batch processing
TRANSACTION_PRINT	Finish batch printing and batch processing

(4) Method Details

PrinterManager — Constructor

Constructor for **com.seikoinstruments.sdk.thermalprinter.PrinterManager** class.

Syntax (a) public **PrinterManager**(Context *context*)

(b) public **PrinterManager**()

Parameter

context	Specify application context to call this method. Example: **MainActivity.this**

Note Use the syntax (a) when using this method newly.
Syntax (b) is a method that will be unsupported in the future. Also, syntax (b) does not support the log function.

connect — Start communicating with printer (Bluetooth)

Starts communication with a printer by Bluetooth connection.
In syntax (a), communication with the printer is always performed in a secure mode.
In syntax (b), communication with the printer is performed by specifying either a secure mode or an insecure mode.

Syntax (a) public void **connect**(int *printerModel*, String *address*) throws **PrinterException**

(b) public void **connect**(int *printerModel*, String *address*, boolean *secure*) throws **PrinterException**

Parameter

printerModel	Printer model constant for Bluetooth connection See "4.2.1(2)① Printer model" for available constants.
address	Bluetooth address Example: "00:11:22:AA:BB:CC"
secure	true Communicates with a printer in secure mode false Communicates with a printer in insecure mode Normally, communication in secure mode is recommended.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.

Description Call this method before using other **PrinterManager** class methods.

The printer specified by *printerModel* is connected to the Bluetooth address specified by *address*.
Also, printer initial setting is performed at the connection based on *printerModel* specified.

Monitoring of the printer status is started with this method. The latest printer status can be retrieved from **getStatus**.
Changes of the printer status can be notified as events by **onStatusChanged** and **setCallbackFunctionListener**.

connect — Start communicating with printer (USB)

Starts communication with a printer.by USB connection.
In syntax (a), the printer model can be specified.
In syntax (b), the printer model can be specified and the context for calling the application can also be defined.

Syntax (a) public void **connect**(int *printerModel*) throws **PrinterException**

(b) public void **connect**(int *printerModel*, Context *context*) throws **PrinterException**

Parameter

printerModel	Printer model constant for USB connection See "4.2.1(2)① Printer model" for available constants.
context	Specify application context to call this method. Example: **MainActivity.this**

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.

Description Call this method before using other **PrinterManager** class methods.

The printer specified by *printerMode*l is connected.
Also, printer initial setting is performed at the connection based on *printerModel* specified.

Monitoring of the printer status is started with this method. The latest printer status can be retrieved from **getStatus**.
Changes of the printer status can be notified as events by **onStatusChanged** and **setCallbackFunctionListener**.

Note Use the syntax (a) when using this method newly.
Syntax (b) is a method that will be unsupported in the future.

disconnect — Stop communicating with printer

Stops communicating with the printer and monitoring the printer status.

Syntax public void **disconnect**() throws **PrinterException**

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.

Description This method discards the print data kept by **controlTransaction**.

Note It is recommended to get the execution response by **PRINTER_RESPONSE_REQUEST** of **getPrinterResponse** method before executing this method. If not, the communication is disconnected by this method before the print data sending from Android device to the printer is completed, and a part of the data may be lost.
If you do not execute **getPrinterResponse** in your program, evaluate your program to confirm no problems arise.

setBarcodeScannerListener — Start/End callback of barcode scanner

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **setBarcodeScannerListener**(BarcodeScannerListener *listener*) throws **PrinterException**

sendText — Send text data

Sends text data.

Syntax public void **sendText**(String *text*) throws **PrinterException**

Parameter *text* Text data to send to the printer
Data size that can be specified at 1 time is 16 KB (16384 bytes).

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description This method encodes the specified text data to printable text data based on **setInternationalCharacter** and **setCodePage**, and then sends it to the printer.

This method does not add any line feed code at the end of the text data. In order to print to the end, add a line feed code to the end of the text data.

sendTextEx — Send format specified text data

Sends format specified text data to the printer.
In syntax (a) and (c), processing starts after pending data is output.
In syntax (b), processing starts in accordance with the specified output control constant for pending data.

Syntax (a) public void **sendTextEx**(String *text*,
CharacterBold *bold*,
CharacterUnderline *underline*,
CharacterReverse *reverse*,
CharacterFont *font*,
CharacterScale *scale*,
PrintAlignment *alignment*) throws **PrinterException**

(b) public void **sendTextEx**(String *text*,
CharacterBold *bold*,
CharacterUnderline *underline*,
CharacterReverse *reverse*,
CharacterFont *font*,
CharacterScale *scale*,
PrintAlignment *alignment,*
OutputPendingData *output*) throws **PrinterException**

(c) public void **sendTextEx**(String *text*,
CharacterBold *bold*,
CharacterUnderline *underline*,
CharacterInversion *inversion*,
CharacterFont *font*,
CharacterScale *scale*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter *text* Text data to send to the printer
Data size that can be specified at 1 time is 16 KB (16384 bytes).

bold Bold print
See "4.2.1(3)① Bold print (CharacterBold)" for available constants.

underline	Underline See "4.2.1(3)② Underline (CharacterUnderline)" for available constants.
reverse	Reverse print See "4.2.1(3)③ Reverse print (CharacterReverse)" for available constants.
inversion	Inversion print See "4.2.1(3)④ Inversion print (CharacterInversion)" for available constants.
font	Font See "4.2.1(3)⑤ Character font (CharacterFont)" for available constants.
scale	Character scale See "4.2.1(3)⑥ Character scale (CharacterScale)" for available constants.
alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.
output	Pending data output specifying See "4.2.1(3)⑧ Pending data output specifying (OutputPendingData)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description This method encodes the specified text data to printable text data based on **setInternationalCharacter** and **setCodePage**, and then sends it to the printer.

For laying out text data by sending following printer commands with **sendBinary** or **sendDataFile**, specify **PENDING_DATA_OUTPUT_TOGETHER** at *output* in the method of syntax (b).

- "Horizontal Tab"
- "Specify Absolute Position"
- "Specify Relative Position"

When the method of the syntax (a) is executed or **PENDING_DATA_OUTPUT_FIRST** is specified at *output* in the method of syntax (b), the print position set in above becomes invalid.

When **PENDING_DATA_OUTPUT_TOGETHER** is specified at output in the method of syntax (b), this method does not add any line feed code at the end of the text data. In order to print to the end, add a line feed code to the end of the text data.

printBarcode — Print barcode

Prints barcode.
In syntax (a), barcode data can be specified as a character string.
In syntax (b), barcode data can be specified as a character string, and the N:W ratio can also be specified.
In syntax (c), barcode data can be specified as a byte array.
The syntax (d) is not supported.

Syntax (a) public void **printBarcode**(BarcodeSymbol *barcodeSymbol*,
String *text*,
ModuleSize *moduleSize*,
int *moduleHeight*,
HriPosition *hriPosition*,
CharacterFont *hriFont*,
PrintAlignment *alignment*) throws **PrinterException**

(b) public void **printBarcode**(BarcodeSymbol *barcodeSymbol*,
String *text*,
ModuleSize *moduleSize*,
int *moduleHeight*,
HriPosition *hriPosition*,
CharacterFont *hriFont*,
PrintAlignment *alignment*,
NwRatio *nwRatio*) throws **PrinterException**

(c) public void **printBarcode**(BarcodeSymbol *barcodeSymbol*,
byte[] *data*,
ModuleSize *moduleSize*,
int *moduleHeight*,
HriPosition *hriPosition*,
CharacterFont *hriFont*,
PrintAlignment *alignment*) throws **PrinterException**

(d) public void **printBarcode**(BarcodeSymbol *barcodeSymbol*,
String *text*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter *barcodeSymbol* Barcode symbol
See "4.2.1(3)⑨ Barcode symbol (BarcodeSymbol)" for available constants.

text (data) Barcode data to send to the printer
The input conditions for barcode data are as follows.

Barcode	Number of Data	Inputtable Data Character String (Data)	Remarks
UPC-A	11 to 12 characters	'0' to '9'	
UPC-E	11 to 12 characters	'0' to '9'	
EAN13 JAN13	12 to 13 characters	'0' to '9'	
EAN8 JAN8	7 to 8 characters	'0' to '9'	
CODE39	1 to 150 characters	'0' to '9' 'A' to 'Z' ' ', '$', '%', '+', '-', '.', '/'	Start code and stop code ('*') are automatically added.
CODE93	1 to 150 bytes	(0x00 to 0x2E)	Input data with 0x2F or more at the end.

Barcode	Number of Data	Inputtable Data Character String (Data)	Remarks
CODE128	2 to 150 bytes	(0x00 to 0x66)	When inputting the start code (0x67 to 0x69) of the CODE128 code set. Input data with 0x67 or more at the end.
		(0x00 to 0x7F)	When starting with a CODE128 special code start code ("{A", "{B", "{C").
ITF	2 to 150 characters (However, an even number)	'0' to '9'	
CODABAR	1 to 150 characters	'0' to '9' '$', '+', '-', '.', '/', ':'	It is needed to specify one of 'A' to 'D' at the beginning and end.
EAN13 add-on JAN13 add-on	Add-on 2: 14 to 15 characters Add-on 5: 17 to 18 characters	'0' to '9'	
Customer Bar Code_JP	-	-	Not supported.
GS1 Databar Omni-directional	13 characters	'0' to '9'	Check digit is automatically added.
GS1 Databar Truncated	13 characters	'0' to '9'	Check digit is automatically added.
GS1 Databar Limited	13 characters	'0' to '9'	Check digit is automatically added.
GS1 Databar Expanded	2 to 255 characters	' ' to "" '%' to '?' 'A' to 'Z' '_' 'a' to 'z' '{'	

moduleSize Barcode width
See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.

moduleHeight Barcode height (dot)

- When *barcodeSymbol* is set to the following, the valid range is 1 to 255.
  BARCODE_SYMBOL_UPC_A
  BARCODE_SYMBOL_UPC_E
  BARCODE_SYMBOL_EAN13
  BARCODE_SYMBOL_JAN13
  BARCODE_SYMBOL_EAN8
  BARCODE_SYMBOL_JAN8
  BARCODE_SYMBOL_CODE39
  BARCODE_SYMBOL_CODE93
  BARCODE_SYMBOL_CODE128
  BARCODE_SYMBOL_ITF
  BARCODE_SYMBOL_CODABAR
  BARCODE_SYMBOL_EAN13_ADDON
  BARCODE_SYMBOL_JAN13_ADDON

- When *barcodeSymbol* is set to the following, the valid range is different by *barcodeSymbol* and *moduleSize*.

barcodeSymbol	*moduleSize*	**Valid Range**
BARCODE_SYMBOL_GS1_OMNI_DIRECTIONAL		
	BARCODE_MODULE_WIDTH_2	66 to 255
	BARCODE_MODULE_WIDTH_3	99 to 255
	BARCODE_MODULE_WIDTH_4	132 to 255
	BARCODE_MODULE_WIDTH_5	165 to 255
	BARCODE_MODULE_WIDTH_6	198 to 255
BARCODE_SYMBOL_GS1_TRUNCATED		
	BARCODE_MODULE_WIDTH_2	26 to 255
	BARCODE_MODULE_WIDTH_3	39 to 255
	BARCODE_MODULE_WIDTH_4	52 to 255
	BARCODE_MODULE_WIDTH_5	65 to 255
	BARCODE_MODULE_WIDTH_6	78 to 255
BARCODE_SYMBOL_GS1_LIMITED		
	BARCODE_MODULE_WIDTH_2	20 to 255
	BARCODE_MODULE_WIDTH_3	30 to 255
	BARCODE_MODULE_WIDTH_4	40 to 255
	BARCODE_MODULE_WIDTH_5	50 to 255
	BARCODE_MODULE_WIDTH_6	60 to 255
BARCODE_SYMBOL_GS1_EXPANDED		
	BARCODE_MODULE_WIDTH_2	68 to 255
	BARCODE_MODULE_WIDTH_3	102 to 255
	BARCODE_MODULE_WIDTH_4	136 to 255
	BARCODE_MODULE_WIDTH_5	170 to 255
	BARCODE_MODULE_WIDTH_6	204 to 255

hriPosition HRI character print position
See "4.2.1(3)⑪ HRI character print position (HriPosition)" for available constants.

hriFont HRI character font
See "4.2.1(3)⑤ Character font (CharacterFont)" for available constants.

alignment Alignment
See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

nwRatio N:W ratio
See "4.2.1(3)⑫ N:W ratio (NwRatio)" for available constants.
Depending on specified *nwRatio* and *moduleSize*, the wide element width is set in the following table.

moduleSize	*nwRatio*		
	NWRATIO_ 1TO2	**NWRATIO_ 1TO2_5**	**NWRATIO_ 1TO3**
BARCODE_MODULE_WIDTH_2	0.500 mm (4 dots)	0.625 mm (5 dots)	0.750 mm (6 dots)
BARCODE_MODULE_WIDTH_3	0.750 mm (6 dots)	1.000 mm (8 dots)	1.125 mm (9 dots)
BARCODE_MODULE_WIDTH_4	1.000 mm (8 dots)	1.250 mm (10 dots)	1.500 mm (12 dots)
BARCODE_MODULE_WIDTH_5	1.250 mm (10 dots)	1.625 mm (13 dots)	1.875 mm (15 dots)
BARCODE_MODULE_WIDTH_6	1.500 mm (12 dots)	1.875 mm (15 dots)	2.250 mm (18 dots)

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printPDF417 Print PDF417

Prints PDF417.
In syntax (a), the PDF417 symbol can be specified.
In syntax (b), the PDF417 symbol is fixed to standard PDF417.

Syntax (a) public void **printPDF417**(String *text*,
ErrorCorrection *errorCorrection*,
int *row*,
int *column*,
ModuleSize *moduleSize*,
int *moduleHeight*,
PrintAlignment *alignment*,
Pdf417Symbol *pdf417Symbol*) throws **PrinterException**

(b) public void **printPDF417**(String *text*,
ErrorCorrection *errorCorrection*,
int *row*,
int *column*,
ModuleSize *moduleSize*,
int *moduleHeight*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter *text* Barcode data to send to the printer

errorCorrection Error correction level
See "4.2.1(3)⑬ Error correction level (ErrorCorrection)" for available constants.

row The number of rows (row)
The valid range is 0 to 90.
When 0 is specified, the number of rows is automatically set.

	column	The number of columns in data area The valid range is 0 to 30. When 0 is specified, the number of columns in the data area is automatically set.
	moduleSize	Nominal fine element width See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.
	moduleHeight	Module height (dot) The valid range is 2 to 127. When the module height is set smaller, some barcode scanners may not read it. Set 3 or more for normal use.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.
	pdf417Symbol	Symbol of PDF417 See "4.2.1(3)⑭ PDF417 symbol (Pdf417Symbol)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printQRcode — Print QR Code

Prints QR Code.
In syntax (a), QR Code Model can be specified.
In syntax (b), QR Code Model is fixed to QR Code Model 2.

Syntax (a) public void **printQRcode**(String *text*,
ErrorCorrection *errorCorrection*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*,
QrModel *model*) throws **PrinterException**

(b) public void **printQRcode**(String *text*,
ErrorCorrection *errorCorrection*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter	*text*	Barcode data to send to the printer The version for either syntax (a) or (b) is automatically set depending on the number of data specified on *text*.
	errorCorrection	Error correction level See "4.2.1(3)⑬ Error correction level (ErrorCorrection)" for available constants.
	moduleSize	Module size See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

	model	QR Code Model See "4.2.1(3)⑮ QR Code Model (QrModel)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printDataMatrix — Print Data Matrix

Prints Data Matrix.

Syntax
```
public void printDataMatrix(String text,
                            DataMatrixModule dataMatrixModule,
                            ModuleSize moduleSize,
                            PrintAlignment alignment) throws PrinterException
```

Parameter	*text*	Barcode data to send to the printer
	dataMatrixModule	The number of Data Matrix modules See "4.2.1(3)⑯ Data Matrix module (DataMatrixModule)" for available constants.
	moduleSize	Module size See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printMaxiCode — Print MaxiCode

Prints MaxiCode.

Syntax
```
public void printMaxiCode(String text,
                          MaxiCodeMode maxiCodeMode,
                          PrintAlignment alignment) throws PrinterException
```

Parameter	*text*	Barcode data to send to the printer • When *maxiCodeMode* is **MAXI_CODE_2** Add the service class (3 digits), the country code (3 digits), and the postal code (9 digits) to the beginning of the data. • When *maxiCodeMode* is **MAXI_CODE_3** Add the service class (3 digits), the country code (3 digits), and the postal code (6 digits) to the beginning of the data.

maxiCodeMode MaxiCode Mode
See "4.2.1(3)⑰ MaxiCode Mode (MaxiCodeMode)" for available constants.

alignment Alignment
See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printGS1DataBarStacked Print GS1 Databar Stacked

Prints GS1 Databar Stacked.

Syntax public void **printGS1DataBarStacked**(String *text*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter *text* Barcode data to send to the printer
Enter 13 characters from '0' to '9'. The leading '01' is automatically added by the printer. The check digit is automatically calculated by the printer.

moduleSize Module size
See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.

alignment Alignment
See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printGS1DataBarStackedOmnidirectional Print GS1 Databar Stacked Omni-directional

Prints GS1 Databar Stacked Omni-directional.

Syntax public void **printGS1DataBarStackedOmnidirectional**(String *text*,
int *moduleHeight*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter *text* Barcode data to send to the printer
Enter 13 characters from '0' to '9'. The leading '01' is automatically added by the printer. The check digit is automatically calculated by the printer.

	moduleHeight	Barcode module height (the number of the modules) The valid range is 33 to 255.
	moduleSize	Module size See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printGS1DataBarExpandedStacked — Print GS1 Databar Expanded Stacked

Prints GS1 Databar Expanded Stacked.

Syntax public void **printGS1DataBarExpandedStacked**(String *text*,
int *column*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter	*text*	Barcode data to send to the printer Enter any number of characters using the followings: ' ', '!', '"', '%', '&', ''', '(', ')', '*', '+', ',', '-', '.', '/', ':', ';', '<', '=', '>', '?', '_', '0' to '9', 'A' to 'Z', 'a' to 'z'. Enter '{1' to FNC1. Be sure to input the check digit because it is not automatically calculated by the printer.
	column	The number of columns Specify the number of the columns in 1 line. The valid range is the even number from 2 to 20.
	moduleSize	Module size See "4.2.1(3)⑩ Module size (ModuleSize)" for available constants.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Reference See "Appendix B Barcode Size List" for details of the barcode size.

printAztecCode — Print Aztec Code

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **printPDF417**(String *text*,
int *layer*,
int *errorCorrection*,
ModuleSize *moduleSize*,
AztecSymbol *aztecSymbol*,
PrintAlignment *alignment*) throws **PrinterException**

cutPaper — Cut paper

Feeds the paper to the paper cut position. The paper is not cut.

Syntax public void **cutPaper**(CuttingMethod *cuttingMethod*) throws **PrinterException**

Parameter *cuttingMethod* Cutting method
See "4.2.1(3)⑱ Cutting method (CuttingMethod)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

feedPosition — Paper form feed

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **feedPosition**(FeedPosition *feedPosition*) throws **PrinterException**

openDrawer — Open cash drawer

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **openDrawer**(DrawerNum *drawerNum*, PulseWidth *onOffTime*) throws **PrinterException**

buzzer — Sound buzzer

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **buzzer**(int *onTime*, int *offTime*) throws **PrinterException**

externalBuzzer — Sound external buzzer

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **externalBuzzer**(BuzzerPattern *buzzer pattern*, int *buzzerCount*)
throws **PrinterException**

sendBinary — Send binary data

Sends binary data to the printer.

Syntax public void **sendBinary**(byte [] *binary*) throws **PrinterException**

Parameter

binary	Binary data to send to the printer Data size that can be specified at 1 time is 3 MB (3145728 bytes).

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description This method sends the specified binary data to the printer without conversion.

By sending printer commands as binary data with this method, printer functions which are not supported in the library become available. However, this method does not support commands which get responses from the printer.

sendDataFile — Send specified file

Sends file data.
In syntax (a), dithering can be specified.
In syntax (b), dithering is fixed to disabled.

Syntax (a) public void **sendDataFile**(String *fileName*,
PrintAlignment *alignment*,
Dithering *dithering*) throws **PrinterException**

(b) public void **sendDataFile**(String *fileName*,
PrintAlignment *alignment*) throws **PrinterException**

Parameter

fileName	File path of the data file to send to the printer The formats that can be entered are described below. • Absolute path string handled by Java standard class "java.io.File" When targeting Android 10 (API 29) or later, please note that some files cannot be handled directly. See "3.5 Precautions - About Scoped Storage" for details. • URI string of the following scheme name handled by the class "android.net.Uri" prepared for Android ・file:// ・content:// It is necessary to specify the URI string obtained from "Storage Access Framework" for this parameter. Please note that URI created without being obtained from "Storage Access Framework" may not be able to open the file. The maximum file size that can be specified is 16 MB (16777216 bytes). The file extensions that can be sent and the file transmission are described below.

- .bmp, .jpg, .jpeg, .png
  Data is sent to the printer as image data. Colored image data is converted to monochrome image by binarization and sent to the printer. Printing is performed at one time after mapping the image data in memory of the printer.

- .txt
  Data is sent to the printer as text data. Text data format supports UTF-8. This method encodes the text data to printable text data based on the settings of **setInternationalCharacter** and **setCodePage**, and then sends it to the printer.
  This method does not add a line feed code at the end of the text data. In order to print to the end, add a line feed code to the end of the text data.

- .bin, .dat
  Data is sent to the printer as the binary data without conversion.

alignment Alignment
The alignment is valid only when the file extension specified on *fileName* is .bmp, .jpg, .jpeg, .png, or .txt.
See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

dithering Dithering
The dithering is valid only when the file extension specified on *fileName* is .bmp, .jpg, .jpeg, or .png.
See "4.2.1(3)⑲ Dithering (Dithering)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

sendImage — Send image

Prints an image of the specified “android.graphics.Bitmap” type.
In syntax (a), image alignment and dithering can be specified.
In syntax (b), image alignment can be specified, and dithering is fixed to disabled.
In syntax (c), image alignment is fixed to left alignment, and dithering is fixed to disabled.

Syntax (a) public void **sendImage**(Bitmap *bitmap*,
PrintAlignment *alignment*,
Dithering *dithering*) throws **PrinterException**

(b) public void **sendImage**(Bitmap *bitmap*,
PrintAlignment *alignment*) throws **PrinterException**

(c) public void **sendImage**(Bitmap *bitmap*) throws **PrinterException**

Parameter *bitmap* "android.graphics.Bitmap" type object to send to the printer
Data is sent to the printer as image. Colored image file is converted to monochrome image by binarization and registered.

alignment Alignment
See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

dithering Dithering
See "4.2.1(3)⑲ Dithering (Dithering)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

printPDF Print PDF page

Prints the specified page of the selected PDF file.

Syntax
```
public void printPDF(String fileName,
                     int startIndex,
                     int endIndex,
                     Rotate rotate,
                     ImageScale imageScale,
                     int bottomMargin,
                     Dithering dithering,
                     PrintAlignment alignment) throws PrinterException
```

Parameter

fileName

File path of the PDF file
The file extension for supporting PDF is .pdf.
The maximum file size that can be specified is 16 MB (16777216 bytes).
The formats that can be entered are described below.

- Absolute path string handled by Java standard class "java.io.File"
  When targeting Android 10 (API 29) or later, please note that some files cannot be handled directly.
  See "3.5 Precautions - About Scoped Storage" for details.

- URI string of the following scheme name handled by the class "android.net.Uri" prepared for Android
  - file://
  - content://

  It is necessary to specify the URI string obtained from "Storage Access Framework" for this parameter. Please note that URI created without being obtained from "Storage Access Framework" may not be able to open the file.

startIndex

Start number of the printed page
The range is -1, 1 to the number of pages in the PDF file.
When -1 is specified for *startIndex*, all pages are printed.
When -1 is specified for *startIndex*, the value of *endIndex* is ignored.
When 1 or more is specified for *startIndex*, pages from the page number specified in *startIndex* to the page number specified in *endIndex* are printed.
When a value more than the value specified for *endIndex* is specified for *startIndex*, an error occurs.
When an out-of-range value is specified for *startIndex* or *endIndex*, an error occurs.

endIndex

End number of the printed page
The range is 1 to 2147483647.
When a value more than the number of pages in the PDF file is specified for *endIndex*, pages from the page number specified in *startIndex* to the last page of the PDF file are printed.

rotate

Rotation direction of the image
See "4.2.1(3)⑳ Image rotation direction (Rotate)" for available constants.

	imageScale	Image scaling See "4.2.1(3)㉑ Image scaling (ImageScale)" for available constants. When **IMAGE_SCALE_WIDTH_FIT** is specified for *imageScale*, the image width is converted to the print width of the printer while maintaining the aspect ratio.
	bottomMargin	Bottom margin (dot) The range is -1, 0 to 2400. When -1 is specified for *bottomMargin*, the image is created and printed while maintaining the bottom margin. When a value between 0 and 2400 is specified for *bottomMargin*, the bottom margin is changed to the specified size. When a value between 0 and 2400 is specified for *bottomMargin*, blank pages are not printed.
	dithering	Dithering See "4.2.1(3)⑲ Dithering (Dithering)Dithering (Dithering)" for available constants.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description PDF is converted to printable format for the printer and is sent to the printer.

The color image data is converted to monochrome image by binarization.

Note No guarantee of printing operation when more than 100 pages are printed at a time.

getStatus — Get printer status

Gets the latest printer status.
In syntax (a), the printer status is returned as the return value.
In syntax (b), the printer status is stored in an array of type int.

Syntax (a) public int **getStatus**() throws **PrinterException**

(b) public void **getStatus**(int [] *buf*) throws **PrinterException**

Return value Status retrieved from the printer

Parameter	*buf*	Status retrieved from the printer

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description Status retrieved from the printer is stored to an integer array.

The printer status is shown below.
When the connection failed, the printer status is shown in 0x80000000.

Bit	Function	Value	
		0	1
0	Voltage error	No error	Error
1	Hardware error	No error	Error
2	Head temperature error	No error	Error
3	Reserved	Fixed	-
4	Out-of-paper error	No error	Error
5	Reserved	Fixed	-
6	Reserved	Fixed	-
7	Reserved	Fixed	-
8	FEED Switch status	OFF	ON
9	Reserved	Fixed	-
10	Paper feed status	Stop	Operating
11	Return-waiting status	Not waiting	Waiting
12	Reserved	Fixed	-
13	Reserved	-	Fixed
14	Reserved	-	Fixed
15	Reserved	-	Fixed
16	FLASH memory rewriting	Not rewriting	Rewriting
17	Reserved	-	Fixed
18	Reserved	-	Fixed
19	Reserved	-	Fixed
20 to 22	Battery remaining capacity level	000: No battery 001: Low (Battery remaining capacity: approx. 10%) 011: Middle (Battery remaining capacity: approx. 40%) 111: Full (Battery remaining capacity: approx. 80%)	
23	Battery error	No error	Error
24 to 31	Reserved	-	Fixed

setCallbackFunctionListener — Start/End callback of printer status change

Starts or ends callback to be executed according to changes of the printer status.

Syntax public void **setCallbackFunctionListener**(CallbackFunctionListener *listener*) throws **PrinterException**

Parameter *listener* Instance of **CallbackFunctionListener** interface

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.

Description Register the process executed by callback with **onStatusChanged**.
When specify the instance of **CallbackFunctionListener** interface in *listener* and execute this method, the callback is started.
When specify null in *listener* and execute this method, the callback is stopped.

The instance kept by **CallbackFunctionLister** interface is discarded by any of the following:
・Execute this method specifying null in *listener*
・Execute **disconnect**

This call of the method can be used when **connect** is executed and **isConnect** is true.

abort — Abort waiting state of printer

Aborts the waiting state of the printer.

Syntax public void **abort**() throws **PrinterException**

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description When sending of image data by **sendDataFile** is aborted, the printer does not accept other processes until the specified image data is received completely. (Methods and transmission data are misinterpreted and recognized as a part of the image data.) To solve this situation, use this method to abort the waiting state of the printer.
Note that when executing this method, a part of unprinted image data may be printed.

registerLogo — Register logo

Registers image data to NV graphics memory in the printer as a logo.
In syntax (a), dithering can be specified.
In syntax (b), dithering is fixed to be disabled.

Syntax (a) public void **registerLogo**(String *fileName*,
String *id*,
Dithering *dithering*) throws **PrinterException**

(b) public void **registerLogo**(String *fileName*,
String *id*) throws **PrinterException**

Parameter *fileName* File path of image file to register as a logo
The formats that can be entered are described below.

- Absolute path string handled by Java standard class "java.io.File"
  When targeting Android 10 (API 29) or later, please note that some files cannot be handled directly.
  See "3.5 Precautions - About Scoped Storage" for details.

- URI string of the following scheme name handled by the class "android.net.Uri" prepared for Android
  ・file://
  ・content://
  It is necessary to specify the URI string obtained from "Storage Access Framework" for this parameter. Please note that URI created without being obtained from "Storage Access Framework" may not be able to open the file.

The file extensions for supporting image data are .bmp, .jpg, .jpeg, and .png. When the image data is colored, it is converted to monochrome image by binarization and registered.

	id	Logo ID to register (key code) Specify the logo ID to register by character string of 2 characters. The valid characters are ASCII character code from 20h (space) to 7Eh (tilde) such as alphanumeric ('0' to '9', 'A' to 'Z', 'a' to 'z').
	dithering	Dithering See "4.2.1(3)⑲ Dithering (Dithering)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

printLogo — Print logo

Prints the registered logo.

Syntax public void **printLogo**(String *id*, PrintAlignment *alignment*) throws **PrinterException**

Parameter	*id*	Logo ID to be printed (key code) Specify the ID of the registered logo as a character string.
	alignment	Alignment See "4.2.1(3)⑦ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

unregisterLogo — Delete registered logo

Deletes the registered logo.

Syntax public void **unregisterLogo**(String *id*) throws **PrinterException**

Parameter	*id*	Logo ID to delete (key code) Specify the ID of the registered logo as a character string.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

registerStyleSheet — Register style sheet

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **registerStyleSheet**(String f*ileName*, int *num*) throws **PrinterException**

unregisterStyleSheet — Delete registered style sheet

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **unregisterStyleSheet**(int *num*) throws **PrinterException**

resetPrinter — Reset printer

Resets the printer hardware.

Syntax public void **resetPrinter**() throws **PrinterException**

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description For Bluetooth connection:
The printer hardware reset is performed by the printer command "Printer Reset".

For USB connection:
The printer reset is performed by using the SOFT_RESET function in USB printer class.

getPrinterResponse — Get various responses from printer

Gets response data from the printer.

Syntax public void **getPrinterResponse**(int *id*, Object *buf*) throws **PrinterException**

Parameter *id* Response type constant
See "4.2.1(2)② Response type" for available constants.

buf Buffer that stores the retrieved response data
This method stores the response data specified by *id* to the object specified by *buf*.
The buffer type varies depending on the response type constant.
See the following table for buffer types.

Response Type Constant		
	Parameter	**Description**
PRINTER_RESPONSE_REQUEST (Execution response request)		
	buf	Specify an int type array of length 1. Specify 0 to 15 (00h to 0Fh) for *buf[0]*. When the response is retrieved successfully, the response code of the execution response request is stored to *buf[0]* with 128 to 143 (80h to 8Fh).
PRINTER_RESPONSE_USER_AREA (Send remaining capacity of user area)		
	buf	Specify an int type array of length 1. When the response is retrieved successfully, the remaining capacity of the user area is stored as a numerical value in bytes.

Response Type Constant	
Parameter	**Description**
PRINTER_RESPONSE_ARRANGE_USER_AREA (Send remaining user area response after defragment)	
buf	Specify an int type array of length 1. When the response is retrieved successfully, the remaining capacity of the user area after defragment is stored as a numerical value in bytes.
PRINTER_RESPONSE_NV_GRAPHICS (Send NV graphics memory capacity)	
buf	Specify an int type array of length 1. When the response is retrieved successfully, the NV graphics memory capacity is stored as a numerical value in bytes.
PRINTER_RESPONSE_KEY_CODE (Send key code list of defined NV graphics)	
buf	Specify an ArrayList<String> array. When the response is retrieved successfully, the key code of NV graphics is as a stored in string array. Example: *buf.size()* = 3, *buf[0]* = "22", *buf[1]* = "23", *buf[2]* = "24", etc.
PRINTER_RESPONSE_BATTERY_STATUS (Battery remaining capacity level)	
buf	Specify an integer array of length 1. When the response is retrieved successfully, the battery remaining capacity level is stored as a value. See "4.2.1(2)③ Battery remaining capacity level" for details of the value. Battery remaining capacity level **BATTERY_STATUS_FULL** : Full (Battery remaining capacity: approx. 80%) **BATTERY_STATUS_MIDDLE** : Middle (Battery remaining capacity: approx.40%) **BATTERY_STATUS_LOW** : Low (Battery remaining capacity: approx. 10%) **BATTERY_STATUS_EMPTY** : No battery
PRINTER_RESPONSE_MAINTENANCE_COUNTER (Send maintenance counter)	
buf	Specify an int array of length 8. When the response is retrieved successfully, the maintenance counter is stored as shown below. *buf[0]* : Number of paper feed lines (unit: 100 dot-line) *buf[1]* : Number of thermal head activation times (unit: 100 dot-line) *buf[2]* : Number of autocutter drive times*1 *buf[3]* : Drive time of printer (unit: minute) *buf[4]* : Number of paper feed lines (integrated value) (unit: 100 dot-line) *buf[5]* : Number of thermal head activation times (integrated value) (unit: 100 dot-line) *buf[6]* : Number of autocutter drive times (integrated value)*1 *buf[7]* : Drive time of printer (integrated value) (unit: minute)
PRINTER_RESPONSE_PRINTER_IDENTIFIER (Send printer ID)	
buf	Specify an ArrayList<PrinterIdentifier> array. When the response is retrieved successfully, a **PrinterIdentifier** class object is stored in the first element of *buf*. Example: *buf.size()* = 1, *buf.get[0]* = "**PrinterIdentifier** class object" See "**4.2.5 PrinterIdentifier Class**" for the **PrinterIdentifier** class object.

*1: Not supported. Always returns 0.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When the data transmission is failed, the communication with the printer is ended, and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

startDiscoveryPrinter — Start printer search (Bluetooth)

Searches for the printer using the Bluetooth connection. The found printer information is stored in **PrinterInfo** class.
In syntax (a), the instance of **PrinterListener** can be specified.
In syntax (b), the instance of **PrinterListener** can be specified and the context for calling the application can also be defined.

Syntax (a) public void **startDiscoveryPrinter**(PrinterListener *listener*) throws **PrinterException**

(b) public void **startDiscoveryPrinter**(PrinterListener *listener*, Context *context*) throws **PrinterException**

Parameter

listener	Instance of **PrinterListener** Completion of this method or cancellation by **cancelDiscoveryPrinter** is notified to the user application as an end event by **finishEvent** through the instance set in *listener*.
context	Context of the application

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.

Description This method may discover other printers besides SII printer.
In addition, the printers in which the Bluetooth connection is already established by the library or other applications are not found.

Do not call this method from the main thread of the application.

Note Use the syntax (a) when using this method newly.
Syntax (b) is a method that will be unsupported in the future.

startDiscoveryPrinter — Start printer search (USB)

Searches for the printer using the USB connection. The found printer information is stored in **PrinterInfo** class.

Syntax public void **startDiscoveryPrinter**(PrinterListener *listener,* int *deviceType*) throws **PrinterException**

Parameter

listener	Instance of **PrinterListener** Completion of this method or cancellation by **cancelDiscoveryPrinter** is notified to the user application as an end event by **finishEvent** through the instance set in *listener*.
deviceType	Port type Specify **PRINTER_TYPE_USB**.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.

Description This method searches for SII printer. The printer information of the found printer is stored to **PrinterInfo** class described later.

startDiscoveryPrinter — Start printer search (TCP/IP)

This method is not supported. When this method is executed, it searches SII printer other than MP-B20.

Syntax public void **startDiscoveryPrinter**(PrinterListener *listener*, int *retry*, int *timeout*) throws**PrinterException**

cancelDiscoveryPrinter — Cancel printer search

Cancels **startDiscoveryPrinter** under execution.

Syntax public void **cancelDiscoveryPrinter**()

Description Cancellation by this method is notified as an end event to the user application by **finishEvent** through the instance set in *listener* of **startDiscoveryPrinter**.

getFoundPrinter — Get found printer information

Gets the information of the printer found by **startDiscoveryPrinter** in ArrayList from the **PrinterInfo** class, which is the storage destination.

Syntax public ArrayList<**PrinterInfo**> **getFoundPrinter**()

Return value ArrayList of **PrinterInfo** class

getSendTimeout — Get send timeout period

Gets the send timeout period.

Syntax public int **getSendTimeout**()

Return value Send timeout period (millisecond: ms)

Description This method can get the send timeout period regardless of whether **isConnect** is true or false.

setSendTimeout — Set send timeout period

Sets the send timeout period.

Syntax public void **setSendTimeout**(int *sendTimeout*)

Parameter *sendTimeout* Send timeout period (millisecond: ms)
The valid range is 100 to 90000.
The value is set to 10000 ms when the value out of the valid range is specified.

Description When the send timeout period is not set by this method, the value is set to 10000.

This method can set the send timeout period regardless of whether **isConnect** is true or false. The set timeout period becomes effective at the next data sending.

getReceiveTimeout — Get receive timeout period

Gets the receive timeout period.

Syntax public int **getReceiveTimeout**()

Return value Receive timeout period (millisecond: ms)

Description This method can get the receive timeout period regardless of whether **isConnect** is true or false.

setReceiveTimeout — Set receive timeout period

Sets the receive timeout period.

Syntax public void **setReceiveTimeout**(int *receiveTimeout*)

Parameter *receiveTimeout* Receive timeout period (millisecond: ms)
The valid range is 100 to 90000.
The value is set to 10000 ms when the value out of the valid range is specified.

Description When the receive timeout period is not set by this method, the value is set to 10000.

This method can set the receive timeout period regardless of whether **isConnect** is true or false.
The set timeout period becomes effective at the next data receiving.

getInternationalCharacter — Get international character set

Gets the value of international character set.

Syntax public int **getInternationalCharacter**()

Return value See "4.2.1(2)④ International character set" for details of the value.

Description When the text data is sent by **sendText**, **sendTextEx** or **sendDataFile**, the print result for the following character codes varies. See "Appendix A Character Set" for details about characters to be printed.

Character codes whose print result varies depending on the international character set configuration
0x23, 0x24, 0x40, 0x5B, 0x5C, 0x5D, 0x5E, 0x60, 0x7B, 0x7C, 0x7D, 0x7E

setInternationalCharacter — Set international character set

Sets the value of international character set.

Syntax public void **setInternationalCharacter**(int *internationalCharacter*)

Parameter *internationalCharacter* International character set constant
See "4.2.1(2)④ International character set" for the values available for setting.
When an invalid value is specified, it is ignored.

Description When the international character set is not set by this method, it is as follows depending on the language setting of an Android device.

When the language setting of the Android device is Japanese:
COUNTRY_JAPAN
When the language setting of the Android device is other languages than Japanese:
COUNTRY_USA

getCodePage — Get codepage

Gets the value of codepage.

Syntax public int **getCodePage**()

Return value See "4.2.1(2)⑤ Codepage" for details of the value.

Description The encoder used for sending the text data by **sendText**, **sendTextEx**, or **sendDataFile** is changed. See "Appendix A Character Set" for details about characters to be printed.

setCodePage — Set codepage

Sets the value of codepage.

Syntax public void **setCodePage**(int *codePage*)

Parameter *codePage* Codepage constant
See "4.2.1(2)⑤ Codepage" for the values available for setting.
When an invalid value is specifies, it is ignored.

Description When the codepage is not set by this method, it is as follows depending on the language setting of an Android device.

When the language setting of the Android device is Japanese:
CODE_PAGE_KATAKANA
When the language setting of the Android device is other languages than Japanese:
CODE_PAGE_1252

getPrinterModel — Get printer model

Gets the value of the connecting printer model.

Syntax public int **getPrinterModel**()

Return value See "4.2.1(2)① Printer model" for details of the value.
PRINTER_MODEL_DEFAULT is returned when **isConnect** is false.

Description Even when the printer is not connected, when **connect** has been succeeded once, the printer model value successfully connected last time is returned.

getPortType — Get connecting port type

Gets the port type used for connecting with the printer.

Syntax public int **getPortType**()

Return value See "4.2.1(2)⑥ Port type" for details of the value.
PRINTER_TYPE_BLUETOOTH is returned when **isConnect** is false.

Description Even when the printer is not connected, when **connect** has been succeeded once, the port type value successfully connected last time is returned.

isConnect — Verify connection state with printer

Verifies connection state with the printer.

Syntax public boolean **isConnect**()

Return value

true	Connected to a printer
false	Not connected to a printer

Description When the data transmission is failed, the communication with the printer is ended, and this method returns false. When false is returned, reconnect with the printer by **connect**.

getSocketKeepingTime — Get socket keeping time

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public int **getSocketKeepingTime**()

setSocketKeepingTime — Set socket keeping time

This method is not supported. When this method is executed, **PrinterException** is thrown.

Syntax public void **setSocketKeepingTime**(int *socketKeepingTime*)

getVersion — Get SDK version

Gets the SDK version as a character string.

Syntax public String **getVersion**()

Return velue SDK version character string (Example: When the SDK version is Ver.1.0.0, the return value is "1.0.0")

Description This method can get the SDK version regardless of whether **isConnect** is true or false.

controlTransaction — Start/End batch processing

Starts or ends batch processing.

Syntax public void **controlTransaction**(TransactionFunction *transactionFunction*)
throws **PrinterException**

Parameter

transactionFunction	Batch processing selection See "4.2.1(3)㉒ Batch processing selection (TransactionFunction) " for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while calling this method.
See "**4.2.6 PrinterException Class**" for details of the error.
When data transmission fails, communication with the printer may be terminated and **PrinterException** may be thrown. See **isConnect** for verifying the connection state with the printer.

Description The procedure of batch processing is as follows:

(1) Start batch processing.
Specify **TRANSACTION_START**.

(2) Execute the method.
In the case of the batch processing target method, buffering of transmission data is started.
The transmission data of the batch processing target method executed during buffering is buffered in the transmission buffer without being sent to the printer.
The maximum size of transmission data to be buffered is system dependent.
If the buffered transmission data exceeds the maximum size, the batch processing target method at the time of exceeding becomes an error. If an error occurs, the transmission data up to the error is retained.
As for the retained transmission data, finish the batch processing in step (3).
In the case of a method other than the batch processing target method, transmission data is immediately executed without being buffered.

(3) Finish batch processing.
When **TRANSACTION_PRINT** is specified, the buffered transmission data is sent to the printer. The buffered transmission data is retained even after sent to the printer.
The retained transmission data is discarded by any of the following:
・Specify **TRANSACTION_CLEAR**
・Specify **TRANSACTION_START**
・Execute **disconnect**

The batch processing target methods are as follows:
・**sendText**
・**sendTextEx**
・**printBarcode**
・**printPDF417**
・**printQRcode**
・**printDataMatrix**
・**printMaxiCode**
・**printGS1DataBarStacked**
・**printGS1DataBarStackedOmnidirectional**
・**printGS1DataBarExpandedStacked**
・**sendBinary**
・**sendDataFile**
・**sendImage**
・**printPDF**
・**printLogo**[*1]

*1: **printLogo** under batch processing does not notify the error even when the registered logo does not exist.

4.2.2 PrinterEvent Class

PrinterEvent class gets the end event that occurs when **startDiscoveryPrinter** is terminated.

(1) Method List

Methods provided by the **PrinterEvent** class are shown in the following table.

Name	Description
getEventType	Get end event

(2) End event constant

Constants used for getting the end event are shown in the following table.

Constant Name	Description	Value
EVENT_FINISHED_DISCOVERY	Completion of **startDiscoveryPrinter**	1
EVENT_CANCELED_DISCOVERY	Cancellation by **cancelDiscoveryPrinter**	2

(3) Method Details

getEventType Get end event

Gets the end event when **startDiscoveryPrinter** is terminated.

Syntax public int **getEventType**()

Return value See "4.2.2(2) End event constant" for details of the value.

Description Whether **startDiscoveryPrinter** has been completed or the search has been canceled by **cancelDiscoveryPrinter** can be determined by the end event.
Even when the printer was not discovered, **EVENT_FINISHED_DISCOVERY** is returned.

4.2.3 PrinterListener Interface

PrinterListener interface is for getting the end event when **startDiscoveryPrinter** is terminated.

(1) Method List

Methods of the **PrinterListener** interface are shown in the following table.

Name	Description
finishEvent	End event of the printer search

(2) Method Details

finishEvent End event of the printer search

End event that is called when **startDiscoveryPrinter** is completed, or when **cancelDiscoveryPrinter** is executed.

Syntax public void **finishEvent**(**PrinterEvent** *event*)

Parameter *event* End event
It is specified by **PrinterEvent** class.

Description This method is an interface, so it is not implemented.
Implement this method in the user application that receives the notification of the end event by completion of **startDiscoveryPrinter** or cancellation by **cancelDiscoveryPrinter**.
Determine the type of the end event by **getEventType** in **PrinterEvent** class.

4.2.4 PrinterInfo Class

PrinterInfo class stores the information of the printer which has been searched by **startDiscoveryPrinter**.

(1) Method List

Printer model name (Bluetooth device name), Bluetooth address, port name (device path) and pairing status can be retrieved. Methods of **PrinterInfo** class are shown in the following table.

Name	Description
getPrinterModelName	Get printer model name
getBluetoothAddress	Get Bluetooth address
getIsBonded	Get pairing status
getDevicePath	Get device path

(2) Method Details

getPrinterModelName Get printer model name

Gets the character string of the printer model name (Bluetooth device name) from the printer information found by **startDiscoveryPrinter**.

Syntax public String **getPrinterModelName**()

Return value Printer model name (Bluetooth device name)

getBluetoothAddress Get Bluetooth address

Gets the character string of the Bluetooth address from the printer information found by **startDiscoveryPrinter**.

Syntax public String **getBluetoothAddress**()

Return value Bluetooth address

getIsBonded Get pairing status

Gets the status of pairing from the printer information found by **startDiscoveryPrinter**.

Syntax public boolean **getIsBonded**()

Return value true Paired
false Not paired

getDevicePath Get device path

Gets the character string of the USB device file path from the printer information found by **startDiscoveryPrinter**.

Syntax public String **getDevicePath**()

Return value Device path

4.2.5 PrinterIdentifier Class

PrinterIdentifier class stores the printer ID of the printer retrieved by **getPrinterResponse**.

(1) Method List

Printer model ID, type ID, ROM version ID, firmware version (main), manufacturer, model name, serial number, firmware version (boot), firmware checksum (boot), firmware checksum (main) and firmware checksum (main + boot) can be retrieved. Methods of **PrinterIdentifier** class are shown in the following table.

Name	Description
getPrinterModelID	Get printer model ID
getTypeID	Get type ID
getRomVersionID	Get ROM version ID
getMainFirmwareVersion	Get firmware version (main)
getManufacturer	Get manufacturer
getModelName	Get model name
getSerialNumber	Get serial number
getBootFirmwareVersion	Get firmware version (boot)
getBootFirmwareChecksum	Get firmware checksum (boot)
getMainFirmwareChecksum	Get firmware checksum (main)
getFirmwareChecksum	Get firmware checksum (main + boot)

(2) Method Details

getPrinterModelID Get printer model ID

Gets the printer model ID from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public byte **getPrinterModelID**()

Return value Printer model ID

getTypeID Get type ID

Gets the type ID from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public byte **getTypeID**()

Return value Type ID

getRomVersionID Get ROM version ID

Gets the ROM version ID from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public byte **getRomVersionID**()

Return value ROM version ID

getMainFirmwareVersion — Get firmware version (main)

Gets the firmware version (main) as a string from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public String **getMainFirmwareVersion**()

Return value Firmware version (main)

getManufacturer — Get manufacturer

Gets the manufacturer name as a string from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public String **getManufacturer**()

Return value Manufacturer name

getModelName — Get model name

Gets the model name as a string from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public String **getModelName**()

Return value Model name

getSerialNumber — Get serial number

Gets the serial number as a string from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public String **getSerialNumber**()

Return value Serial number

getBootFirmwareVersion — Get firmware version (boot)

Gets the firmware version (boot) as a string from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public String **getBootFirmwareVersion**()

Return value Firmware version (boot)

getBootFirmwareChecksum — Get firmware checksum (boot)

Gets the firmware checksum (boot) from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public byte[] **getBootFirmwareChecksum**()

Return value Firmware checksum (boot)

getMainFirmwareChecksum — Get firmware checksum (main)

Gets the firmware checksum (main) from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public byte[] **getMainFirmwareChecksum**()

Return value Firmware checksum (main)

getFirmwareChecksum — Get firmware checksum (main + boot)

Gets the firmware checksum (main + boot) from the printer ID of the printer retrieved by **getPrinterResponse**.

Syntax public byte[] **getFirmwareChecksum**()

Return value Firmware checksum (main + boot)

4.2.6 PrinterException Class

(1) Method List

Methods provided by the **PrinterException** class are shown in .the following table.

Name	Description
PrinterException	Constructor
getErrorCode	Get error codes

(2) Constant List

① Error code

Constants used for getting error codes are shown in following table.

Constant Name	Description	Value
ERROR_ACCESS_DENIED	Failed to get the handle.*1	-1
	An unavailable port was specified.	
	An unsupported method was specified.	
ERROR_SHARING_ VIOLATION	An already opened port was specified.	-11
ERROR_PORT_NOT_ OPENED	The port is not open.	-12
ERROR_DEVICE_NOT_CONNECTED	There is a problem with connection between the Android device and the printer.	-21
ERROR_OFFLINE	Disconnected state or the printer is offline.	-22
ERROR_DEVICE_INITIALIZE_FAILED	Failed to change the printer settings. Data sending to the printer is not completed within the send timeout period, or data receiving from the printer is not completed within the receive timeout period.	-31
ERROR_DATA_SIZE_ZERO	0-byte data was specified.	-101
ERROR_OVER_MAX_DATA_SIZE	Maximum data size is exceeded.	-102
ERROR_ENCODE_FAILED	An error occurred in encoding text data.*1	-111
ERROR_TIMEOUT	Send timeout occurred.	-201
	Receive timeout occurred.	
ERROR_FILE_NOT_FOUND	The specified file is not found.	-301
ERROR_FILE_USED	The specified file is in use by another process.	-302
ERROR_FILE_INVALID	The specified file is invalid.	-303
ERROR_LOW_MEMORY	Memory shortage occurred when loading image data file.	-311
ERROR_OVER_ MAX_IMAGE	Either or both of width and height of image data exceeds the number of printable maximum dots.	-312
ERROR_LOGO_NOT_ DEFINED	The logo is not registered.	-313
ERROR_LOW_USER_AREA	Remaining user area is insufficient.	-401
ERROR_LOW_EXTERNAL_ RAM	Remaining RAM capacity is insufficient.	-402
ERROR_INVALID_PARAM	The specified parameter is invalid.	-9999

*1: Abnormal processing might have occurred.

(3) Method Details

PrinterException — Constructor

Constructor for the **com.seikoinstruments.sdk.thermalprinter.PrinterException** class.

Syntax public **PrinterException**(int *code*, String *message*)

getErrorCode — Get error codes

Gets the error code for thrown exception.

Syntax public int **getErrorCode**()

Return value See "4.2.6(2) Constant List" for details of the error.

4.2.7 CallbackFunctionListener Interface

CallbackFunctionListener Interface is an interface for getting the change event of printer status.

(1) Method List

Method of **CallbackFunctionListener** Interface is shown below.

Name	Description
onStatusChanged	Change event of printer status

(2) Method Details

onStatusChanged Change event of printer status

Syntax public void **onStatusChanged**(int *status*)

Parameter *status* Printer status

Description This method is called at the following timing.
・When **setCallbackFunctionListener** is executed.
・When the printer status is changed.

The change event of printer status is notified when **isConnect** is true.

This method is an interface, so it is not implemented.
Implement the optional process in the class that receives a callback of the printer status change.

Do not execute the APIs of **PrinterManager** within this method.

4.2.8 BarcodeScannerListener Interface

BarcodeScannerListener Interface is an interface for getting barcode scanner connection, barcode scanner disconnection, or received barcode data.

MP-B20 does not support this interface.

4.2.9 SmartLabelManager Class

SmartLabelManager class provides the function to convert the label file (*.sl, *.slex) created using SII Layout Editor into the printable data from the printer.

Do not use this class because it is not supported.

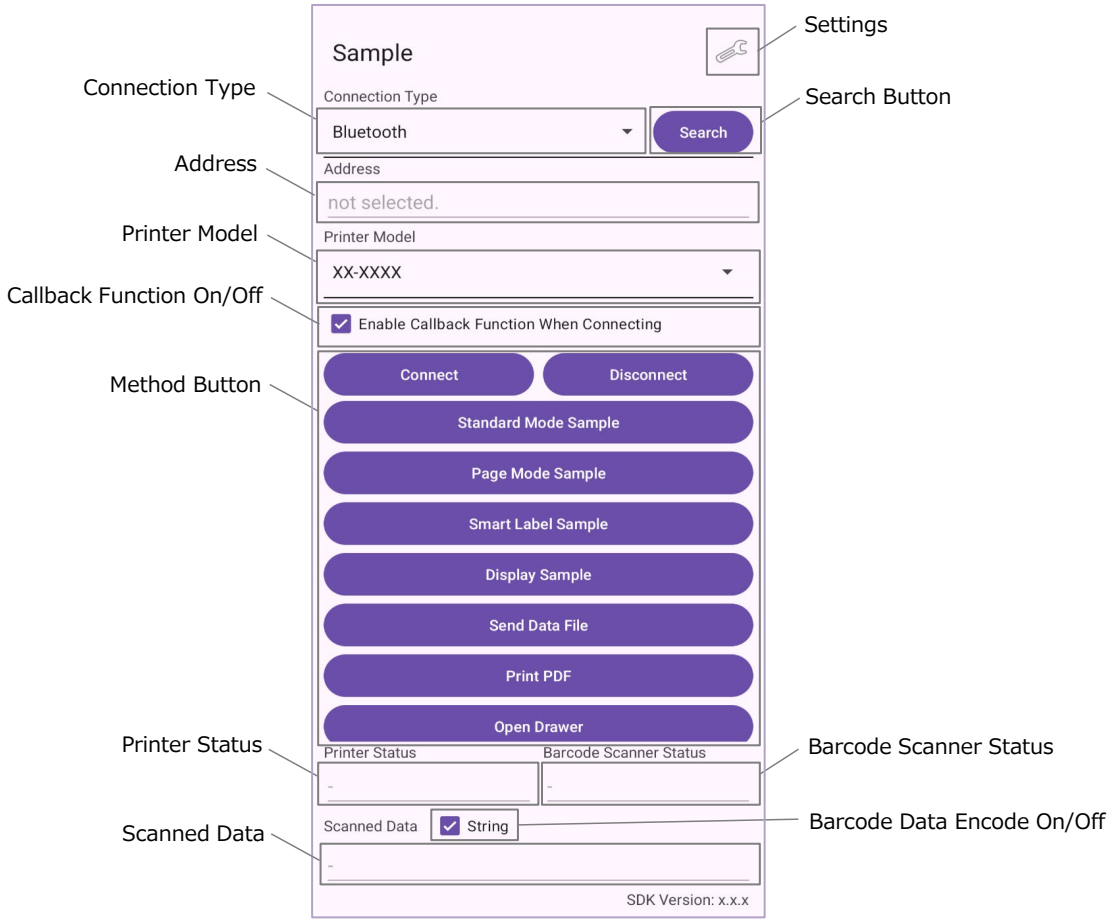
Chapter 5
Sample Program

This chapter describes the sample program provided by SII print class library.

5.1 Screen

SII print class library includes the sample program in Android Studio project format.
This section describes the screen of the sample program.

5.1.1 Main Screen

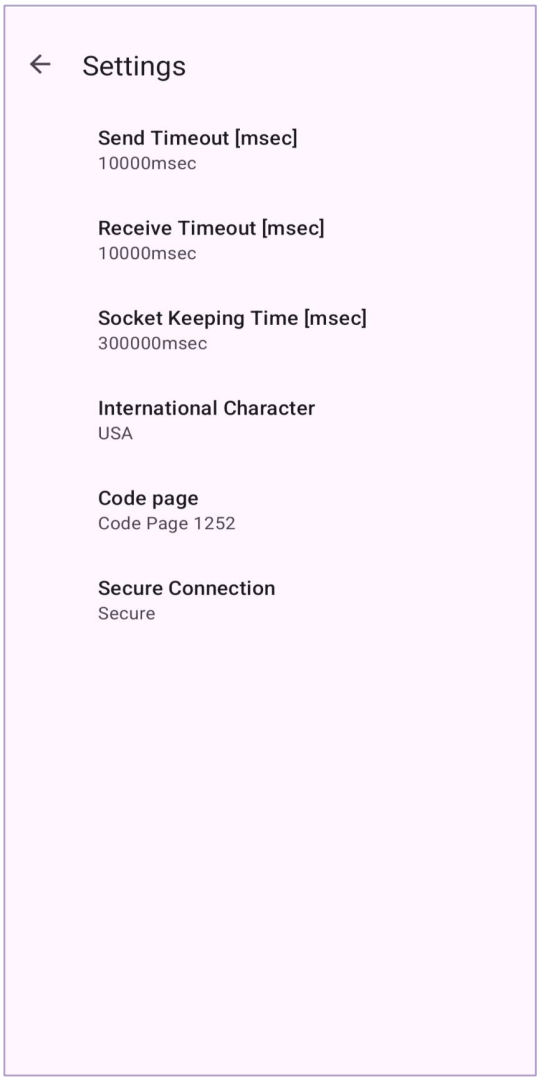


Item	Description
Settings	Tapping the [Settings] button opens the function setting screen. In order to go back to the main screen, tap ← button on the top left of the screen.
Connection type	Selects the connection type to the printer.
Search Button	Starts searching for the type of printer specified in [Connection Type]. Transits to the printer search view. A list of the searched printers is displayed. The printer is selected by tapping the searched printer and returns to the main view.
Address	Displays the information about the selected printer.
Printer Model	Specifies the printer model.
Callback Function On/Off	Select whether to enable the callback function when connecting to the printer. On : Starts the callback function when connecting. Off : The callback function does not respond.
Method Button*1	In addition to the method buttons for executing **connect** and **disconnect**, the sample by the combination of some methods can be printed and checked for the operation of peripheral devices.
Printer Status	Displays the printer status. When [Callback Function On/Off] is On, the latest printer status is displayed.
Barcode Scanner Status	Displays the connection status of the barcode scanner. MP-B20 does not support the barcode scanner.
Barcode Data Encode On/Off	Selects the conversion of barcode data read by the barcode scanner. MP-B20 does not support the barcode scanner.
Scanned Data	Displays the barcode data read by the barcode scanner. MP-B20 does not support the barcode scanner.

*1: Supported functions vary by model. Only supported functions can be operated.

5.1.2 [Settings] Screen

Various setting functions are displayed in [Settings].



5.2 Precaution

The sample program is subject to change without notice.

No guarantee of proper operation and support are provided for the sample program.

Chapter 6
Disclaimer

We closely monitor the development of SII print class library in order to avoid problems. However, we are not responsible for any damages arising out of the use of SII print class library.

Appendix A
Character Set

A.1 Codepage Table (Character Code Table)

The codepages when **COUNTRY_USA** is set for the international character set are shown below. Print results of the specific character codes vary depending on the setting of the international character set. See "A.2 International Character Set" for the specific character codes.

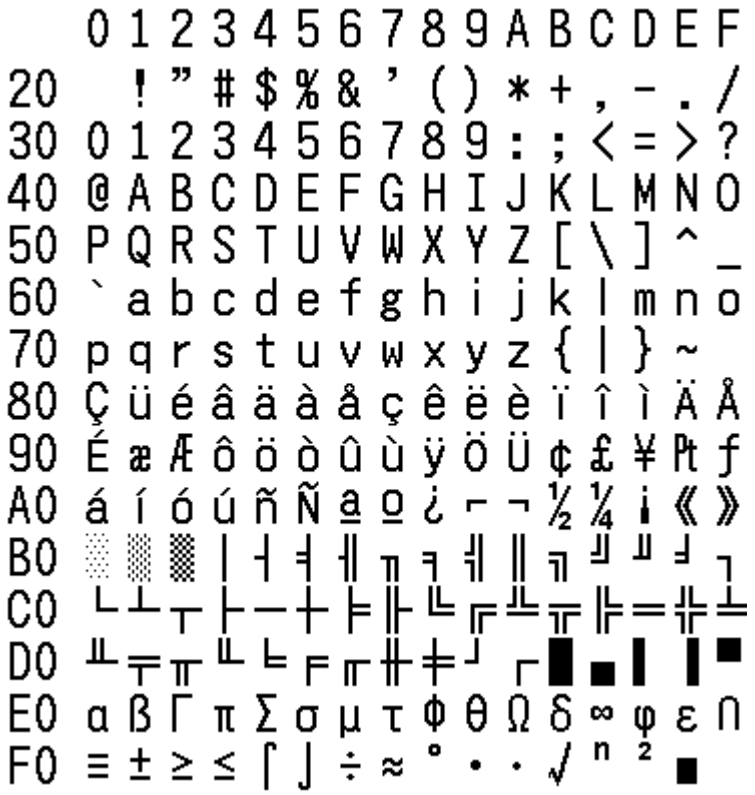


Figure A-1 CODE_PAGE_437 (USA, Standard Europe)

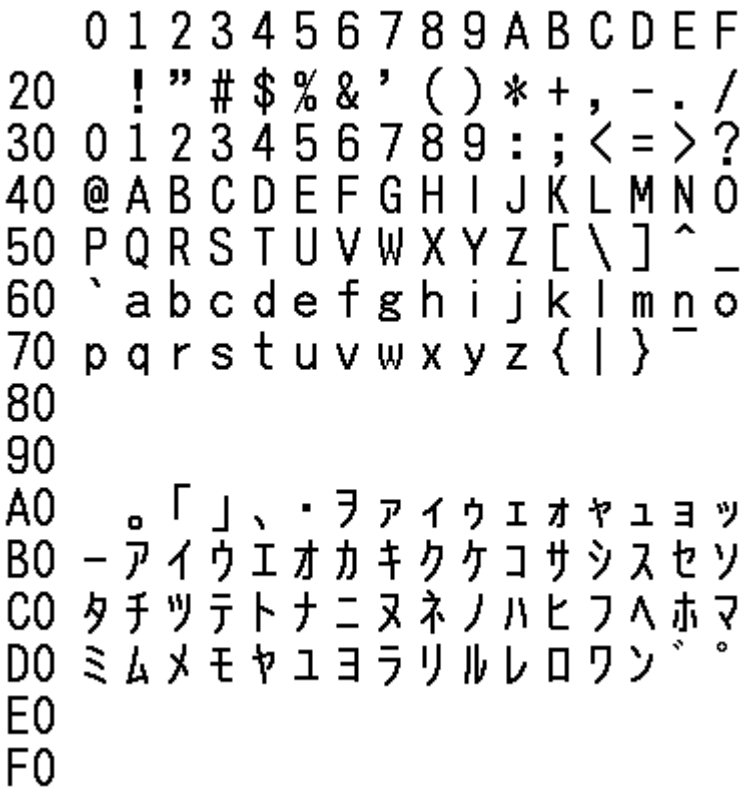


Figure A-2 CODE_PAGE_KATAKANA

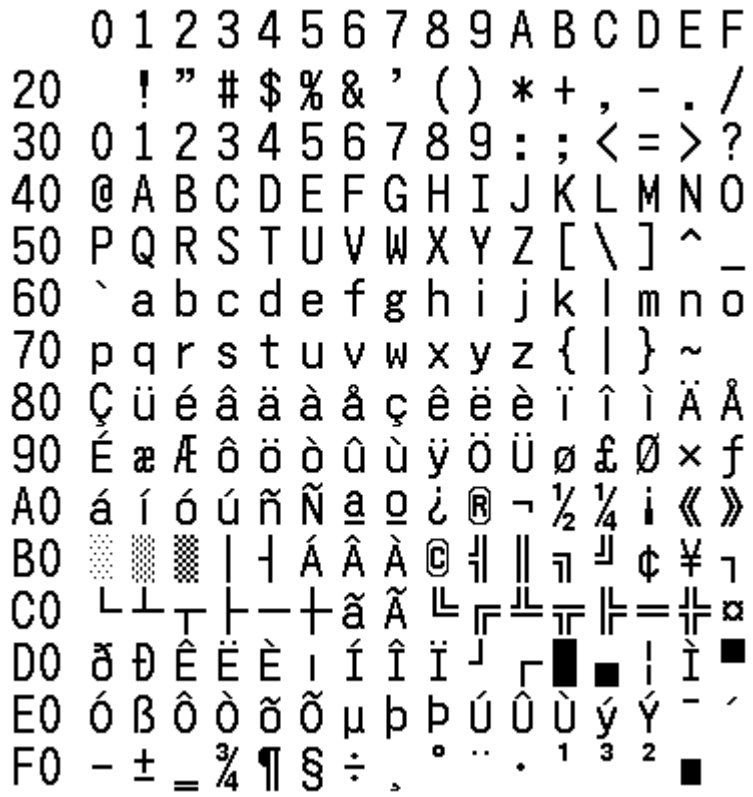


Figure A-3 CODE_PAGE_850 (Multilingual)

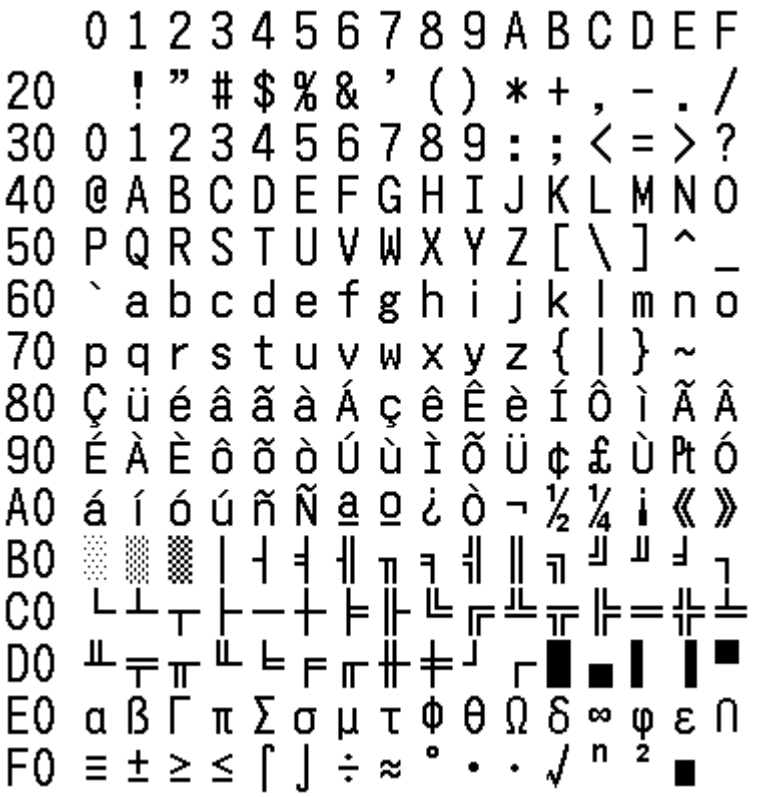


Figure A-4 CODE_PAGE_860 (Portuguese)

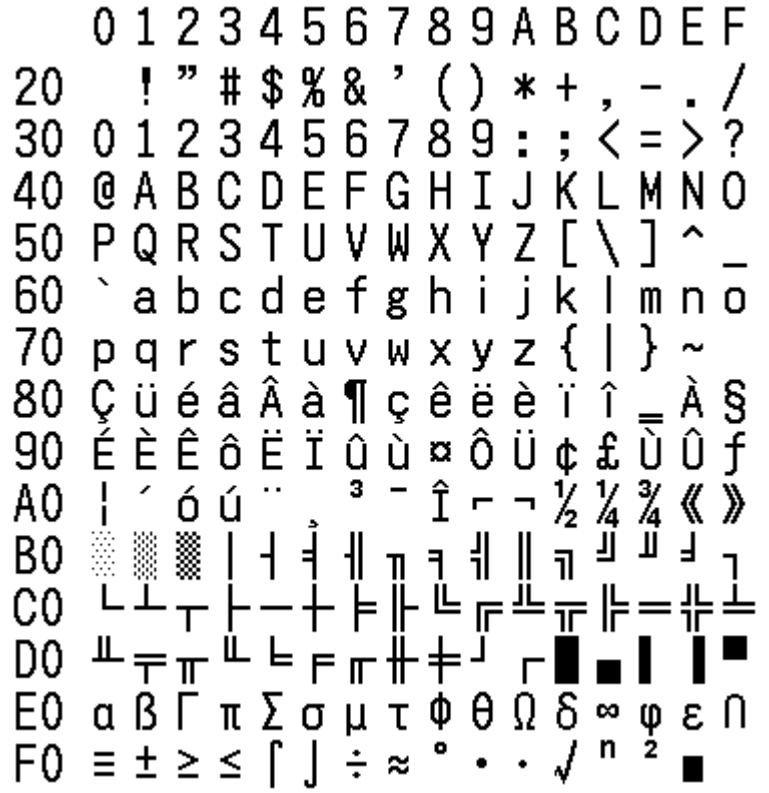


Figure A-5 CODE_PAGE_863 (Canadian-French)

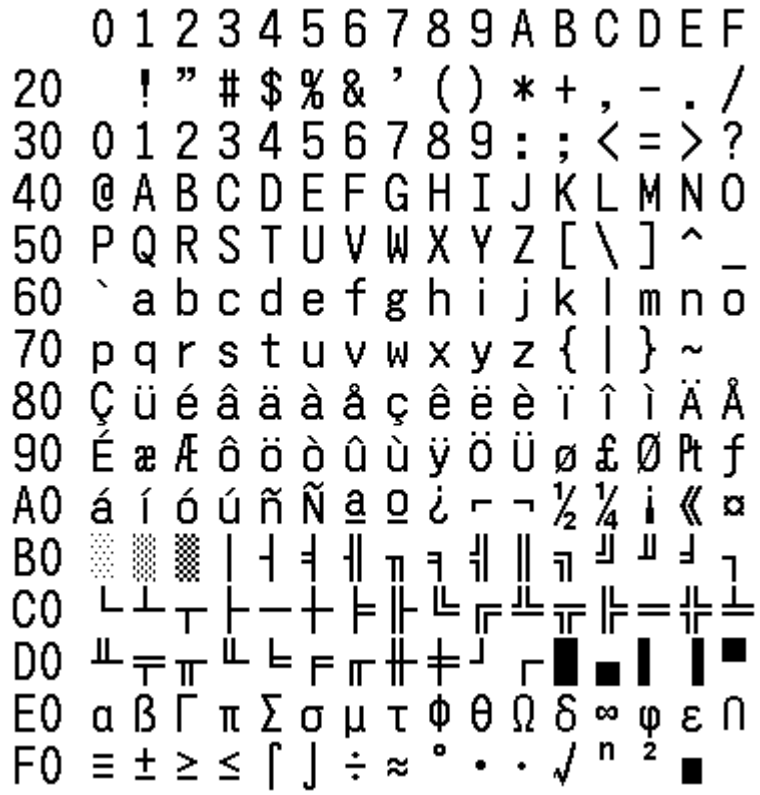


Figure A-6 CODE_PAGE_865 (Nordic)

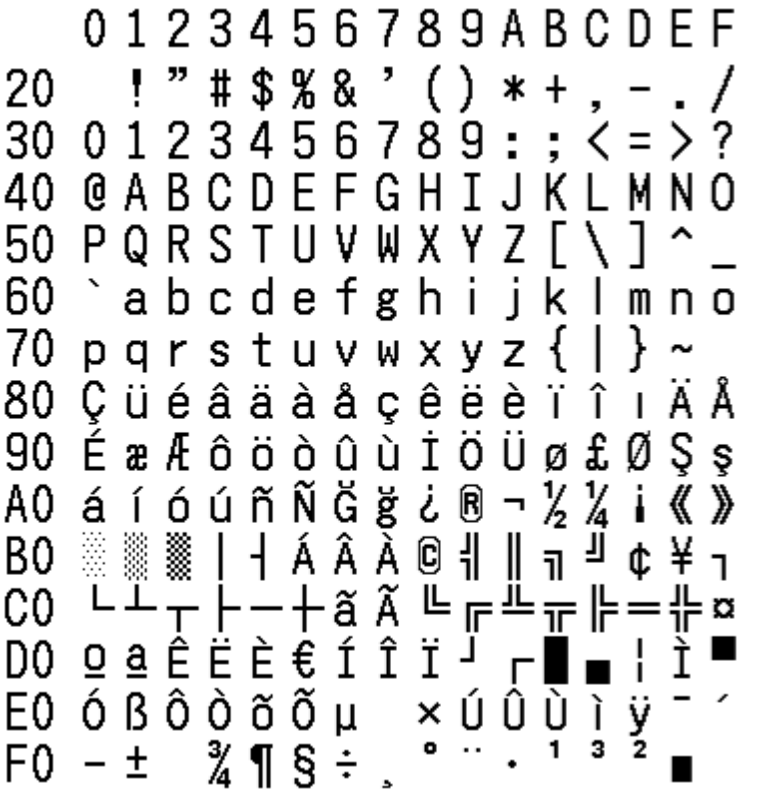


Figure A-7 CODE_PAGE_857 (Turkish)

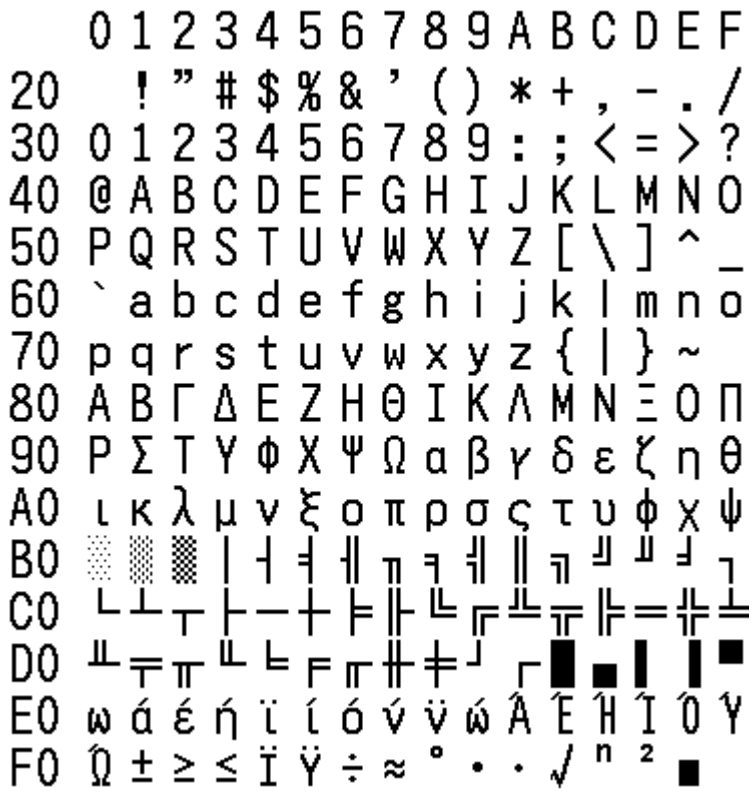


Figure A-8 CODE_PAGE_737 (Greek)

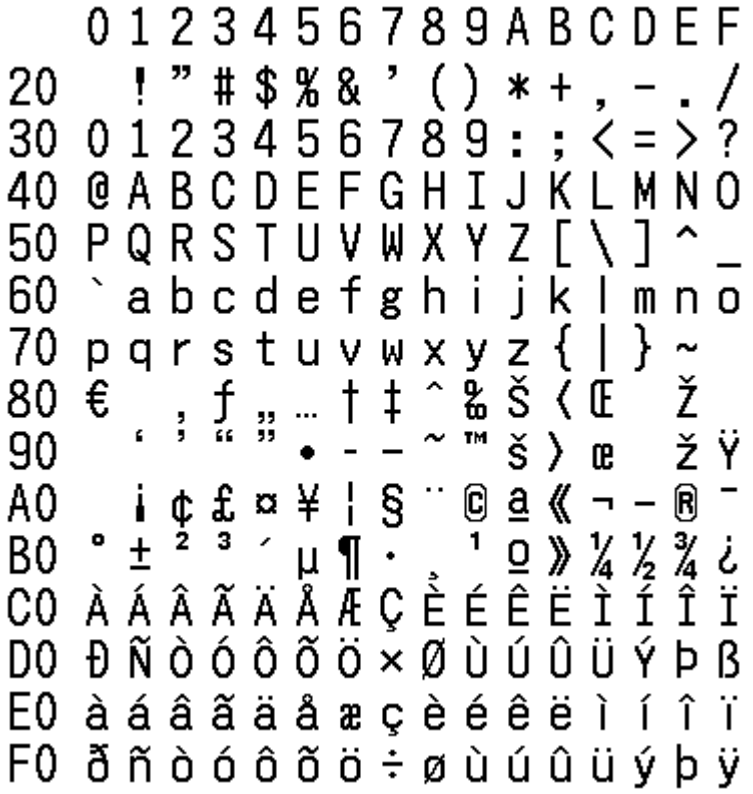


Figure A-9 CODE_PAGE_1252 (Latin)

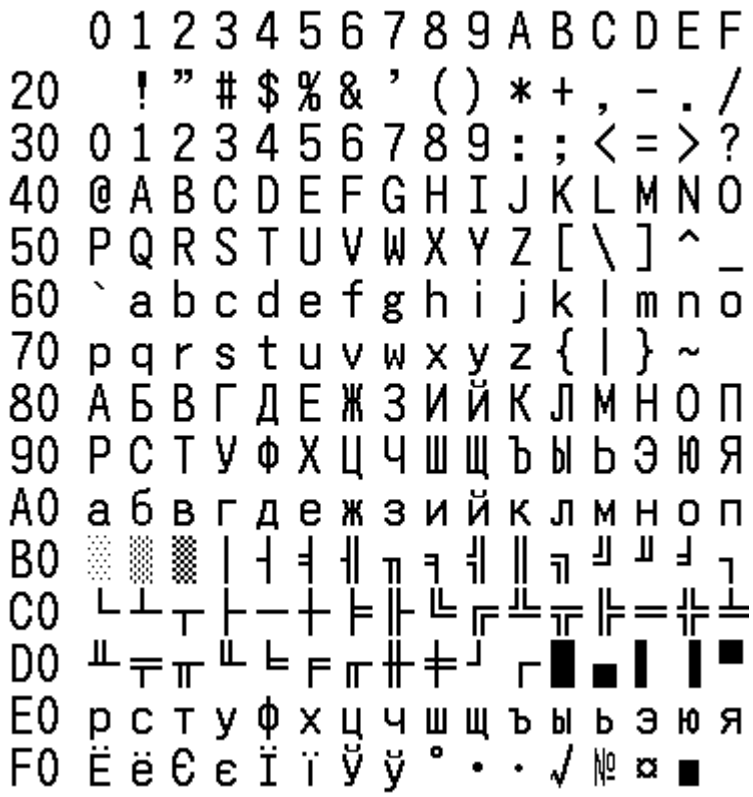


Figure A-10 CODE_PAGE_866 (Russian)

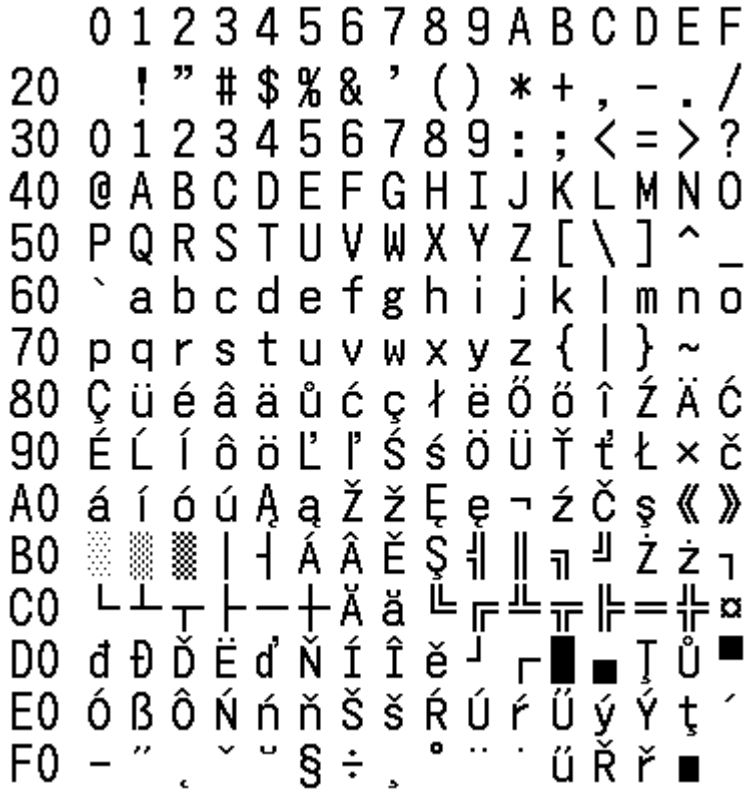


FIgure A-11 CODE_PAGE_852 (Eastern Europe)

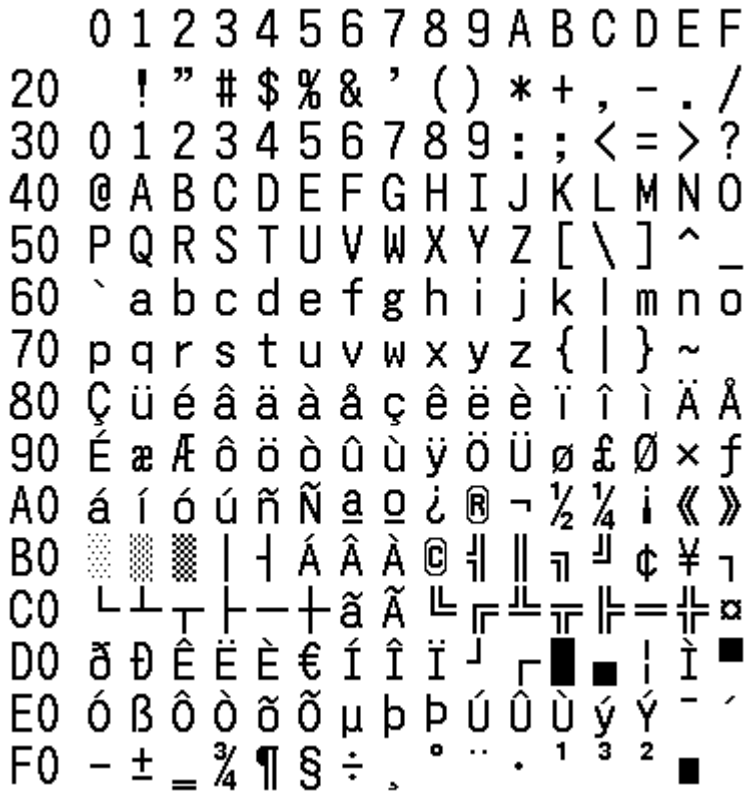


Figure A-12 CODE_PAGE_858 (Euro)

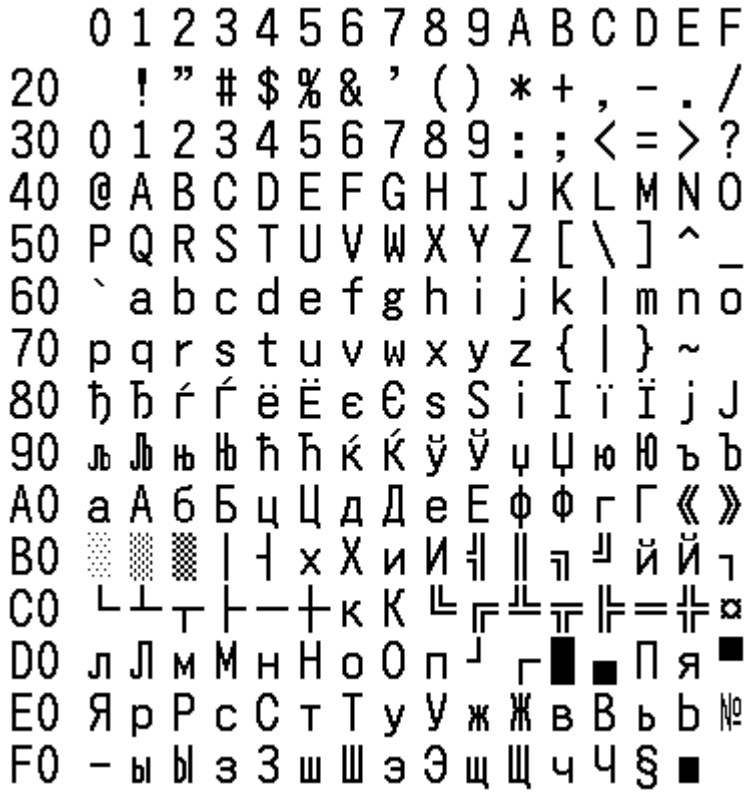


Figure A-13 CODE_PAGE_855 (Cyrillic)

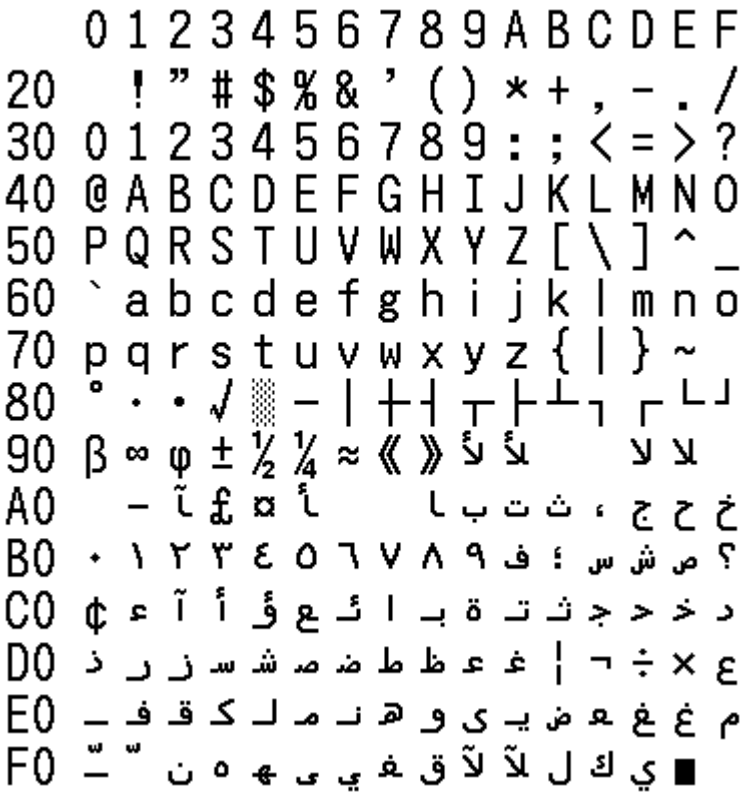


Figure A-14 CODE_PAGE_864 (Arabic)

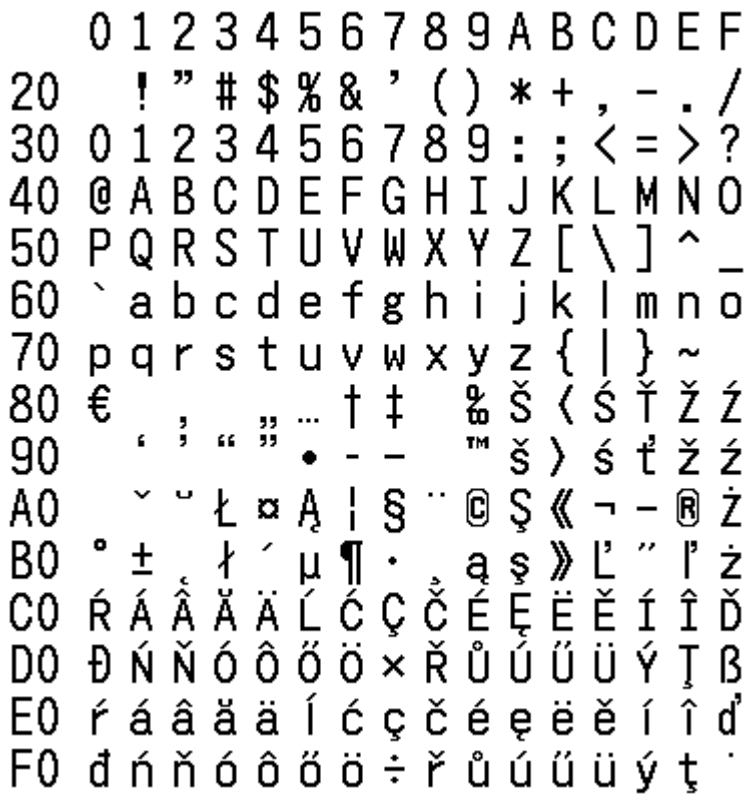


Figure A-15 CODE_PAGE_1250 (Central European)



Figure A-16 CODE_PAGE_1251 (Cyrillic)

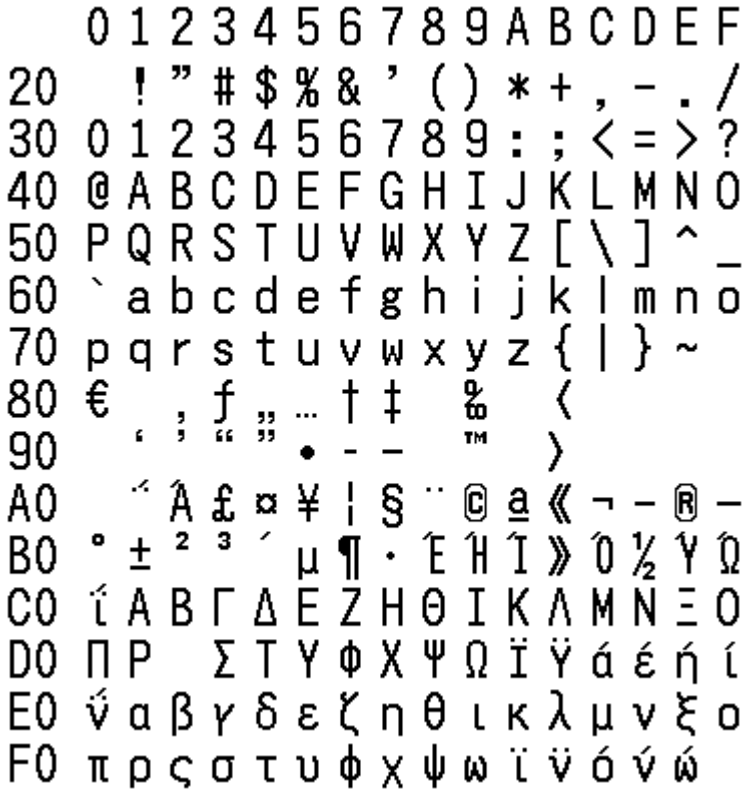


Figure A-17 CODE_PAGE_1253 (Greek)



Figure A-18 CODE_PAGE_1254 (Turkish)

A.2 International Character Set

Print results of the specific character codes vary depending on the setting of the international character set.
The following table shows the specific character codes and their print results.

	23	24	40	5B	5C	5D	5E	60	7B	7C	7D	7E
COUNTRY_USA	#	$	@	[	\	]	^	`	{	|	}	~
COUNTRY_FRANCE	#	$	à	°	ç	§	^	`	é	ù	è	¨
COUNTRY_GERMANY	#	$	§	Ä	Ö	Ü	^	`	ä	ö	ü	ß
COUNTRY_ENGLAND	£	$	@	[	\	]	^	`	{	|	}	~
COUNTRY_DENMARK_1	#	$	@	Æ	Ø	Å	^	`	æ	ø	å	~
COUNTRY_SWEDEN	#	¤	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
COUNTRY_ITALY	#	$	@	°	\	é	^	ù	à	ò	è	ì
COUNTRY_SPAIN	₧	$	@	¡	Ñ	¿	^	`	¨	ñ	}	~
COUNTRY_JAPAN	#	$	@	[	¥	]	^	`	{	|	}	~
COUNTRY_NORWAY	#	¤	É	Æ	Ø	Å	Ü	é	æ	ø	å	ü
COUNTRY_DENMARK_2	#	$	É	Æ	Ø	Å	Ü	é	æ	ø	å	ü
COUNTRY_SPAIN_2	#	$	á	¡	Ñ	¿	é	`	í	ñ	ó	ú
COUNTRY_LATIN_AMERICA	#	$	á	¡	Ñ	¿	é	ü	í	ñ	ó	ú
COUNTRY_ARABIA	#	$	@	[	\	]	^	`	{	|	}	~

Figure A-19 International Character Set

Appendix B
Barcode Size List

B.1 Barcode Size List

B.1.1 printBarcode



(1) Height of the barcode image

hriFont	*hriPosition*	Length from Top of Barcode to Reference Point	Height of Barcode Image
FONT_A	**HRI_NONE**	*moduleHeight*	*moduleHeight*
	HRI_POSITION_ABOVE	*moduleHeight* + 32	*moduleHeight* + 32
	HRI_POSITION_BELOW	*moduleHeight*	*moduleHeight* + 32
	HRI_POSITION_ABOVE_BELOW	*moduleHeight* + 64	*moduleHeight* + 64
FONT_B	**HRI_NONE**	*moduleHeight*	*moduleHeight*
	HRI_POSITION_ABOVE	*moduleHeight* + 24	*moduleHeight* + 24
	HRI_POSITION_BELOW	*moduleHeight*	*moduleHeight* + 24
	HRI_POSITION_ABOVE_BELOW	*moduleHeight* + 48	*moduleHeight* + 48

(2) Width of the barcode image

barcodeSymbol	*moduleSize*	**Width of Barcode Image**
BARCODE_UPC_A	**BARCODE_MODULE_WIDTH_2**	226
	BARCODE_MODULE_WIDTH_3	339
	BARCODE_MODULE_WIDTH_4	452
	BARCODE_MODULE_WIDTH_5	565
	BARCODE_MODULE_WIDTH_6	678
BARCODE_UPC_E	**BARCODE_MODULE_WIDTH_2**	130
	BARCODE_MODULE_WIDTH_3	195
	BARCODE_MODULE_WIDTH_4	260
	BARCODE_MODULE_WIDTH_5	325
	BARCODE_MODULE_WIDTH_6	390
BARCODE_EAN13	**BARCODE_MODULE_WIDTH_2**	226
	BARCODE_MODULE_WIDTH_3	339
	BARCODE_MODULE_WIDTH_4	452
	BARCODE_MODULE_WIDTH_5	565
	BARCODE_MODULE_WIDTH_6	678
BARCODE_JAN13	**BARCODE_MODULE_WIDTH_2**	226
	BARCODE_MODULE_WIDTH_3	339
	BARCODE_MODULE_WIDTH_4	452
	BARCODE_MODULE_WIDTH_5	565
	BARCODE_MODULE_WIDTH_6	678
BARCODE_EAN8	**BARCODE_MODULE_WIDTH_2**	162
	BARCODE_MODULE_WIDTH_3	243
	BARCODE_MODULE_WIDTH_4	324
	BARCODE_MODULE_WIDTH_5	405
	BARCODE_MODULE_WIDTH_6	486
BARCODE_JAN8	**BARCODE_MODULE_WIDTH_2**	162
	BARCODE_MODULE_WIDTH_3	243
	BARCODE_MODULE_WIDTH_4	324
	BARCODE_MODULE_WIDTH_5	405
	BARCODE_MODULE_WIDTH_6	486
BARCODE_CODE93	**BARCODE_MODULE_WIDTH_2**	18 × number of barcode data + 96
	BARCODE_MODULE_WIDTH_3	27 × number of barcode data + 144
	BARCODE_MODULE_WIDTH_4	36 × number of barcode data + 192
	BARCODE_MODULE_WIDTH_5	45 × number of barcode data + 240
	BARCODE_MODULE_WIDTH_6	54 × number of barcode data + 288

barcodeSymbol	*moduleSize*	**Width of Barcode Image**
BARCODE_CODE128	**BARCODE_MODULE_WIDTH_2**	22 × number of barcode data + 66
	BARCODE_MODULE_WIDTH_3	33 × number of barcode data + 99
	BARCODE_MODULE_WIDTH_4	44 × number of barcode data + 132
	BARCODE_MODULE_WIDTH_5	55 × number of barcode data + 165
	BARCODE_MODULE_WIDTH_6	66 × number of barcode data + 198
BARCODE_GS1_OMNI_DIRECTIONAL	**BARCODE_MODULE_WIDTH_2**	192
	BARCODE_MODULE_WIDTH_3	288
	BARCODE_MODULE_WIDTH_4	384
	BARCODE_MODULE_WIDTH_5	480
	BARCODE_MODULE_WIDTH_6	576
BARCODE_GS1_TRUNCATED	**BARCODE_MODULE_WIDTH_2**	192
	BARCODE_MODULE_WIDTH_3	288
	BARCODE_MODULE_WIDTH_4	384
	BARCODE_MODULE_WIDTH_5	480
	BARCODE_MODULE_WIDTH_6	576
BARCODE_GS1_LIMITED	**BARCODE_MODULE_WIDTH_2**	158
	BARCODE_MODULE_WIDTH_3	237
	BARCODE_MODULE_WIDTH_4	316
	BARCODE_MODULE_WIDTH_5	395
	BARCODE_MODULE_WIDTH_6	474
BARCODE_GS1_EXPANDED[*1]	**BARCODE_MODULE_WIDTH_2**	number of barcode module × 2
	BARCODE_MODULE_WIDTH_3	number of barcode module × 3
	BARCODE_MODULE_WIDTH_4	number of barcode module × 4
	BARCODE_MODULE_WIDTH_5	number of barcode module × 5
	BARCODE_MODULE_WIDTH_6	number of barcode module × 6

*1: The number of barcode module is determined by the barcode data to be specified.

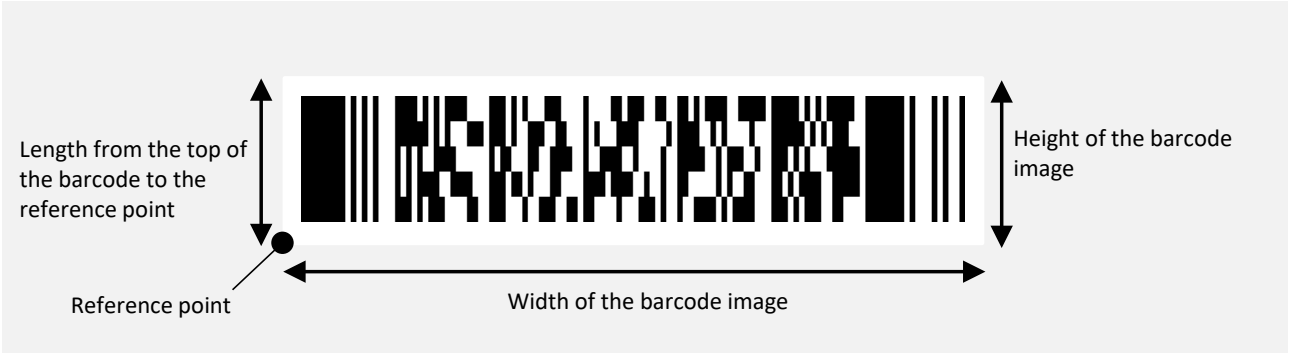
barcodeSymbol	*nwRatio*	*moduleSize*	**Width of Barcode Image**
BARCODE_CODE39	**NWRATIO_1TO2**	**BARCODE_MODULE_WIDTH_2**	26 × number of barcode data + 90
		BARCODE_MODULE_WIDTH_3	39 × number of barcode data + 135
		BARCODE_MODULE_WIDTH_4	52 × number of barcode data + 180
		BARCODE_MODULE_WIDTH_5	65 × number of barcode data + 225
		BARCODE_MODULE_WIDTH_6	78 × number of barcode data + 270
	NWRATIO_1TO2_5	**BARCODE_MODULE_WIDTH_2**	29 × number of barcode data + 96
		BARCODE_MODULE_WIDTH_3	45 × number of barcode data + 147
		BARCODE_MODULE_WIDTH_4	58 × number of barcode data + 192
		BARCODE_MODULE_WIDTH_5	74 × number of barcode data + 243
		BARCODE_MODULE_WIDTH_6	87 × number of barcode data + 288
	NWRATIO_1TO3	**BARCODE_MODULE_WIDTH_2**	32 × number of barcode data + 102
		BARCODE_MODULE_WIDTH_3	48 × number of barcode data + 153
		BARCODE_MODULE_WIDTH_4	64 × number of barcode data + 204
		BARCODE_MODULE_WIDTH_5	80 × number of barcode data + 255
		BARCODE_MODULE_WIDTH_6	96 × number of barcode data + 306
BARCODE_ITF	**NWRATIO_1TO2**	**BARCODE_MODULE_WIDTH_2**	14 × number of barcode data + 56
		BARCODE_MODULE_WIDTH_3	21 × number of barcode data + 84
		BARCODE_MODULE_WIDTH_4	28 × number of barcode data + 112
		BARCODE_MODULE_WIDTH_5	35 × number of barcode data + 140
		BARCODE_MODULE_WIDTH_6	42 × number of barcode data + 168
	NWRATIO_1TO2_5	**BARCODE_MODULE_WIDTH_2**	16 × number of barcode data + 57
		BARCODE_MODULE_WIDTH_3	25 × number of barcode data + 86
		BARCODE_MODULE_WIDTH_4	32 × number of barcode data + 114

barcodeSymbol	*nwRatio*	*moduleSize*	**Width of Barcode Image**
PM_BARCODE_ITF	**NWRATIO_1TO2_5**	**BARCODE_MODULE_WIDTH_5**	41 × number of barcode data + 143
		BARCODE_MODULE_WIDTH_6	48 × number of barcode data + 171
	NWRATIO_1TO3	**BARCODE_MODULE_WIDTH_2**	18 × number of barcode data + 58
		BARCODE_MODULE_WIDTH_3	27 × number of barcode data + 87
		BARCODE_MODULE_WIDTH_4	36 × number of barcode data + 116
		BARCODE_MODULE_WIDTH_5	45 × number of barcode data + 145
		BARCODE_MODULE_WIDTH_6	54 × number of barcode data + 174
BARCODE_CODABAR*1	**NWRATIO_1TO2**	**BARCODE_MODULE_WIDTH_2**	20 × number of data + 2 × (2 + number of wide data) + 38
		BARCODE_MODULE_WIDTH_3	30 × number of data + 3 × (2 + number of wide data) + 57
		BARCODE_MODULE_WIDTH_4	40 × number of data + 4 × (2 + number of wide data) + 76
		BARCODE_MODULE_WIDTH_5	50 × number of data + 5 × (2 + number of wide data) + 95
		BARCODE_MODULE_WIDTH_6	60 × number of data + 6 × (2 + number of wide data) + 114
	NWRATIO_1TO2_5	**BARCODE_MODULE_WIDTH_2**	22 × number of data + 3 × (2 + number of wide data) + 38
		BARCODE_MODULE_WIDTH_3	34 × number of data + 5 × (2 + number of wide data) + 57
		BARCODE_MODULE_WIDTH_4	44 × number of data + 6 × (2 + number of wide data) + 76
		BARCODE_MODULE_WIDTH_5	56 × number of data + 8 × (2 + number of wide data) + 95
		BARCODE_MODULE_WIDTH_6	66 × number of data + 9 × (2 + number of wide data) + 114
	NWRATIO_1TO3	**BARCODE_MODULE_WIDTH_2**	24 × number of data + 4 × (2 + number of wide data) + 38
		BARCODE_MODULE_WIDTH_3	36 × number of data + 6 × (2 + number of wide data) + 57
		BARCODE_MODULE_WIDTH_4	48 × number of data + 8 × (2 + number of wide data) + 76
		BARCODE_MODULE_WIDTH_5	60 × number of data + 10 × (2 + number of wide data) + 95
		BARCODE_MODULE_WIDTH_6	72 × number of data + 12 × (2 + number of wide data) + 114

*1: The number of data is the number of all characters except for the start and stop characters.
The wide data is the number of ": / . +".

barcodeSymbol	**Number of Data**	*moduleSize*	**Width of Barcode Image**
BARCODE_EAN13_ ADDON	14 or 15	**BARCODE_MODULE_WIDTH_2**	276
		BARCODE_MODULE_WIDTH_3	414
		BARCODE_MODULE_WIDTH_4	552
		BARCODE_MODULE_WIDTH_5	690
		BARCODE_MODULE_WIDTH_6	828
	17 or 18	**BARCODE_MODULE_WIDTH_2**	330
		BARCODE_MODULE_WIDTH_3	495
		BARCODE_MODULE_WIDTH_4	660
		BARCODE_MODULE_WIDTH_5	825
		BARCODE_MODULE_WIDTH_6	990
BARCODE_JAN13_ ADDON	14 or 15	**BARCODE_MODULE_WIDTH_2**	276
		BARCODE_MODULE_WIDTH_3	414
		BARCODE_MODULE_WIDTH_4	552
		BARCODE_MODULE_WIDTH_5	690
		BARCODE_MODULE_WIDTH_6	828
	17 or 18	**BARCODE_MODULE_WIDTH_2**	330
		BARCODE_MODULE_WIDTH_3	495
		BARCODE_MODULE_WIDTH_4	660
		BARCODE_MODULE_WIDTH_5	825
		BARCODE_MODULE_WIDTH_6	990

B.1.2 printPDF417



(1) Height of the barcode image

moduleSize	**Height of Barcode Image**[*1]
PDF417_MODULE_WIDTH_2	*moduleHeight* × *row*[*2] + 8
PDF417_MODULE_WIDTH_3	*moduleHeight* × *row*[*2] + 12
PDF417_MODULE_WIDTH_4	*moduleHeight* × *row*[*2] + 16
PDF417_MODULE_WIDTH_5	*moduleHeight* × *row*[*2] + 20
PDF417_MODULE_WIDTH_6	*moduleHeight* × *row*[*2] + 24
PDF417_MODULE_WIDTH_7	*moduleHeight* × *row*[*2] + 28
PDF417_MODULE_WIDTH_8	*moduleHeight* × *row*[*2] + 32

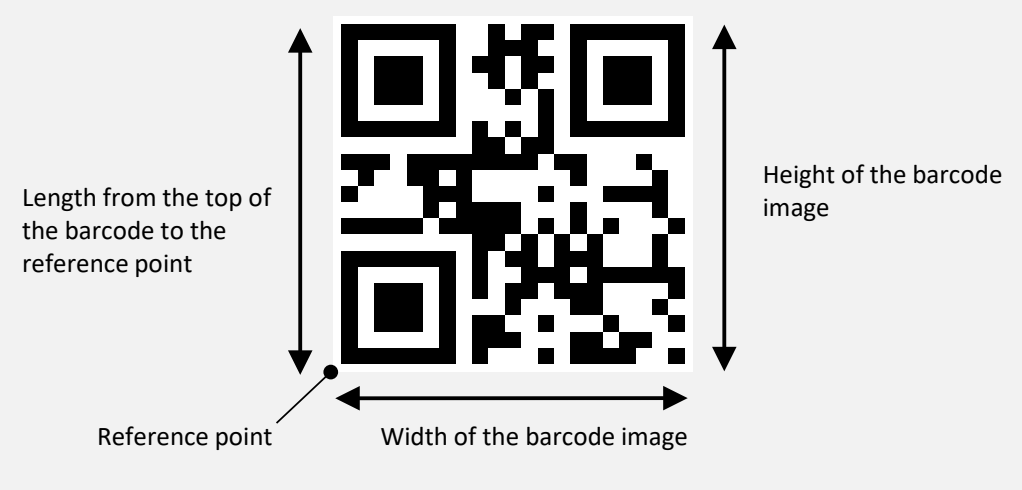
*1: Height of the barcode image = Length from the top of the barcode to the reference point
*2: Parameter $row \neq 0$

(2) Width of the barcode image

pdf417Symol	*moduleSize*	**Width of Barcode Image**
PDF417_STANDARD	**PDF417_MODULE_WIDTH_2**	(17 × *column*[*1] + 69) × 2 + 8
	PDF417_MODULE_WIDTH_3	(17 × *column*[*1] + 69) × 2 + 12
	PDF417_MODULE_WIDTH_4	(17 × *column*[*1] + 69) × 2 + 16
	PDF417_MODULE_WIDTH_5	(17 × *column*[*1] + 69) × 2 + 20
	PDF417_MODULE_WIDTH_6	(17 × *column*[*1] + 69) × 2 + 24
	PDF417_MODULE_WIDTH_7	(17 × *column*[*1] + 69) × 2 + 28
	PDF417_MODULE_WIDTH_8	(17 × *column*[*1] + 69) × 2 + 32
PDF417_COMPACT	**PDF417_MODULE_WIDTH_2**	(17 × *column*[*1] + 35) × 2 + 8
	PDF417_MODULE_WIDTH_3	(17 × *column*[*1] + 35) × 2 + 12
	PDF417_MODULE_WIDTH_4	(17 × *column*[*1] + 35) × 2 + 16
	PDF417_MODULE_WIDTH_5	(17 × *column*[*1] + 35) × 2 + 20
	PDF417_MODULE_WIDTH_6	(17 × *column*[*1] + 35) × 2 + 24
	PDF417_MODULE_WIDTH_7	(17 × *column*[*1] + 35) × 2 + 28
	PDF417_MODULE_WIDTH_8	(17 × *column*[*1] + 35) × 2 + 32

*1: Parameter $column \neq 0$

B.1.3 printQRCode



(1) Height and width of the barcode image

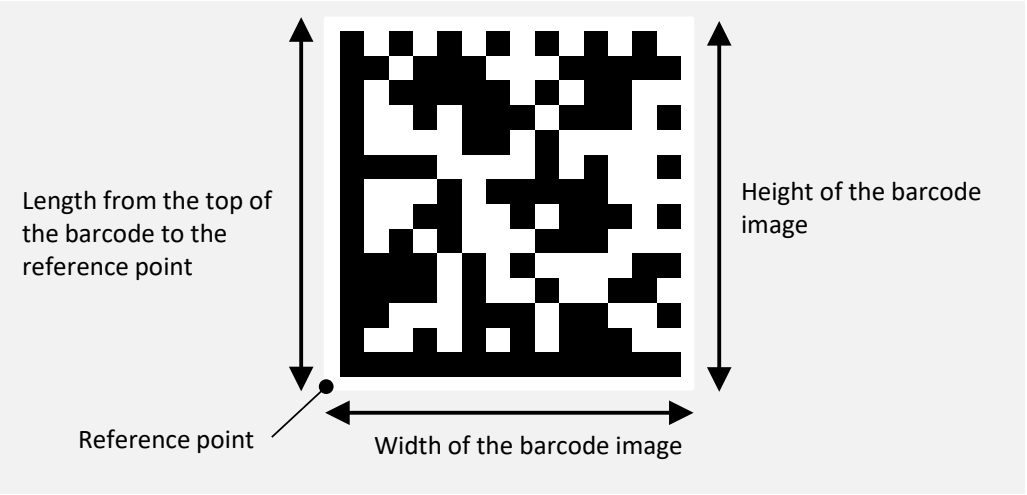
Height[*1] and width of the barcode image = (4 × version[*2] + 17 + 8) × module size value

*1: Height of the barcode image = Length from the top of the barcode to the reference point
*2: The version is determined by the content of the barcode data and the error correction level.

Module Size Value

moduleSize	**Module Size Value**
QR_MODULE_SIZE_2	2
QR_MODULE_SIZE_3	3
QR_MODULE_SIZE_4	4
QR_MODULE_SIZE_5	5
QR_MODULE_SIZE_6	6
QR_MODULE_SIZE_7	7
QR_MODULE_SIZE_8	8
QR_MODULE_SIZE_9	9
QR_MODULE_SIZE_10	10
QR_MODULE_SIZE_11	11
QR_MODULE_SIZE_12	12
QR_MODULE_SIZE_13	13
QR_MODULE_SIZE_14	14
QR_MODULE_SIZE_15	15
QR_MODULE_SIZE_16	16

B.1.4 printDataMatrix



(1) Height and width of the barcode image

Height of the barcode image = (number of vertical module + 2) × module size value

Width of the barcode image = (number of horizontal module + 2) × module size value

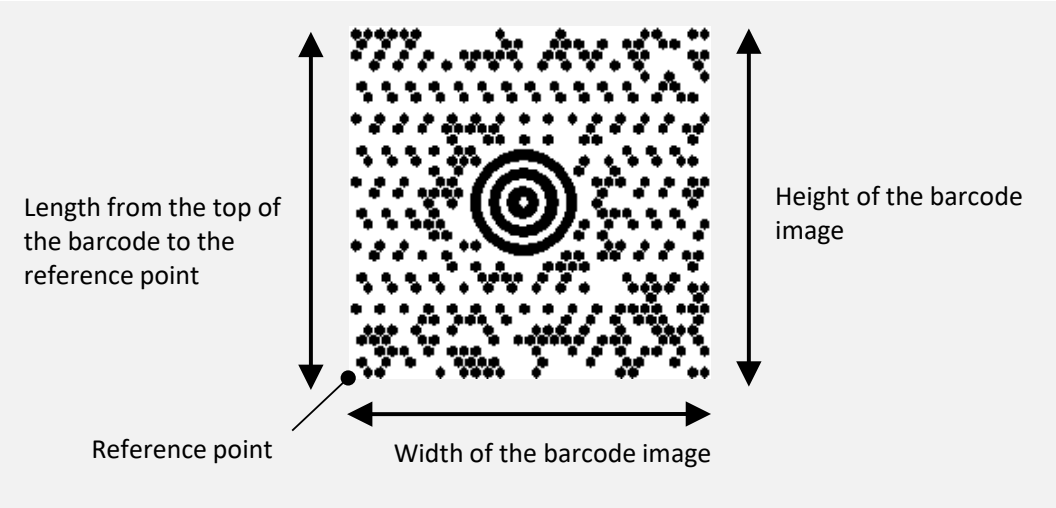
dataMatrixModule	**Number of Vertical Module**	**Number of Horizontal Module**
DATA_MATRIX_10_10	10	10
DATA_MATRIX_12_12	12	12
DATA_MATRIX_14_14	14	14
DATA_MATRIX_16_16	16	16
DATA_MATRIX_18_18	18	18
DATA_MATRIX_20_20	20	20
DATA_MATRIX_22_22	22	22
DATA_MATRIX_24_24	23	23
DATA_MATRIX_26_26	26	26
DATA_MATRIX_32_32	32	32
DATA_MATRIX_36_36	36	36
DATA_MATRIX_40_40	40	40
DATA_MATRIX_44_44	44	44
DATA_MATRIX_48_48	48	48
DATA_MATRIX_52_52	52	52
DATA_MATRIX_64_64	64	64
DATA_MATRIX_72_72	72	72
DATA_MATRIX_80_80	80	80
DATA_MATRIX_88_88	88	88
DATA_MATRIX_96_96	96	96
DATA_MATRIX_104_104	104	104
DATA_MATRIX_120_120	120	120

dataMatrixModule	Number of Vertical Module	Number of Horizontal Module
DATA_MATRIX_132_132	132	132
DATA_MATRIX_144_144	144	144
DATA_MATRIX_8_18	8	18
DATA_MATRIX_8_32	8	32
DATA_MATRIX_12_26	12	26
DATA_MATRIX_12_36	12	36
DATA_MATRIX_16_36	16	36
DATA_MATRIX_16_48	16	48

Module Size Value

moduleSize	Module Size Value
DATAMATRIX_MODULE_SIZE_2	2
DATAMATRIX_MODULE_SIZE_3	3
DATAMATRIX_MODULE_SIZE_4	4
DATAMATRIX_MODULE_SIZE_5	5
DATAMATRIX_MODULE_SIZE_6	6
DATAMATRIX_MODULE_SIZE_7	7
DATAMATRIX_MODULE_SIZE_8	8
DATAMATRIX_MODULE_SIZE_9	9
DATAMATRIX_MODULE_SIZE_10	10
DATAMATRIX_MODULE_SIZE_11	11
DATAMATRIX_MODULE_SIZE_12	12
DATAMATRIX_MODULE_SIZE_13	13
DATAMATRIX_MODULE_SIZE_14	14
DATAMATRIX_MODULE_SIZE_15	15
DATAMATRIX_MODULE_SIZE_16	16

B.1.5 printMaxicode



(1) Height of the barcode image

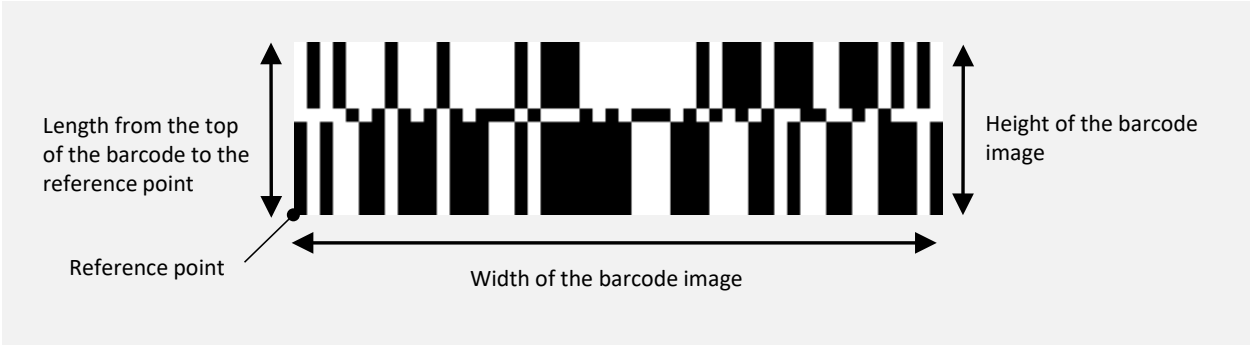
Height of the barcode image[*1] = 215

*1: Height of the barcode image = Length from the top of the barcode to the reference point

(2) Width of the barcode image

Width of the barcode image = 225

B.1.6 printGS1DataBarStacked



(1) Height and width of the barcode image

Height of the barcode image[*1] = 13 × module size value

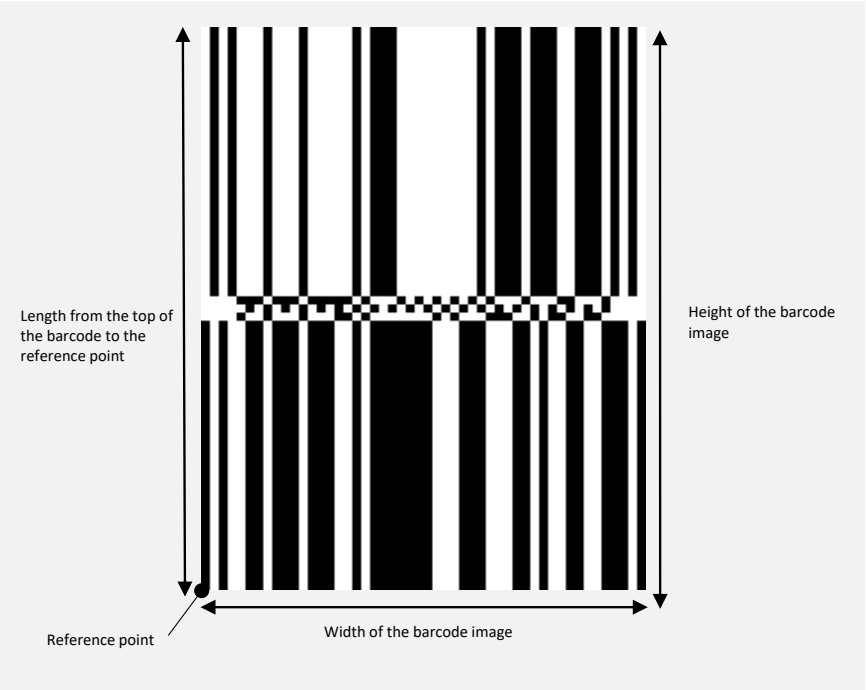
*1: Height of the barcode image = Length from the top of the barcode to the reference point

Width of the barcode image = 50 × module size value

Module Size Value

moduleSize	**Module Size Value**
GS1DATABAR_MODULE_SIZE_2	2
GS1DATABAR_MODULE_SIZE_3	3
GS1DATABAR_MODULE_SIZE_4	4
GS1DATABAR_MODULE_SIZE_5	5
GS1DATABAR_MODULE_SIZE_6	6
GS1DATABAR_MODULE_SIZE_7	7
GS1DATABAR_MODULE_SIZE_8	8
GS1DATABAR_MODULE_SIZE_9	9
GS1DATABAR_MODULE_SIZE_10	10
GS1DATABAR_MODULE_SIZE_11	11
GS1DATABAR_MODULE_SIZE_12	12
GS1DATABAR_MODULE_SIZE_13	13
GS1DATABAR_MODULE_SIZE_14	14
GS1DATABAR_MODULE_SIZE_15	15
GS1DATABAR_MODULE_SIZE_16	16

B.1.7 printGS1DataBarStackedOmnidirectional



(1) Height and width of the barcode image

Height of the barcode image[*1] = (*moduleHeight* × 2 + 3) × module size value

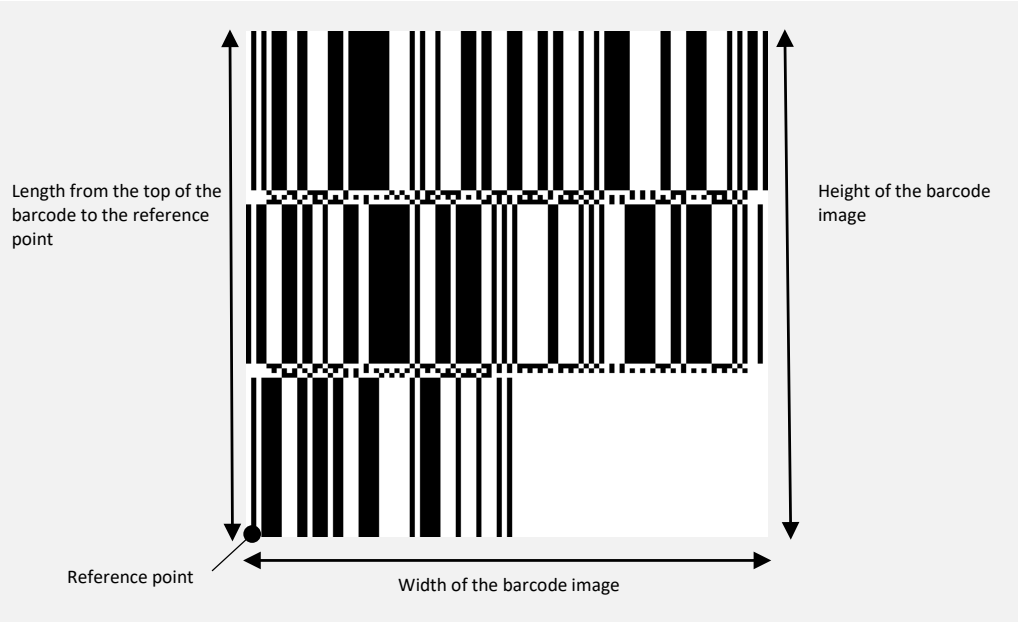
*1: Height of the barcode image = Length from the top of the barcode to the reference point

Width of the barcode image = 50 × module size value

Module Size Value

moduleSize	**Module Size Value**
GS1DATABAR_MODULE_SIZE_2	2
GS1DATABAR_MODULE_SIZE_3	3
GS1DATABAR_MODULE_SIZE_4	4
GS1DATABAR_MODULE_SIZE_5	5
GS1DATABAR_MODULE_SIZE_6	6
GS1DATABAR_MODULE_SIZE_7	7
GS1DATABAR_MODULE_SIZE_8	8
GS1DATABAR_MODULE_SIZE_9	9
GS1DATABAR_MODULE_SIZE_10	10
GS1DATABAR_MODULE_SIZE_11	11
GS1DATABAR_MODULE_SIZE_12	12
GS1DATABAR_MODULE_SIZE_13	13
GS1DATABAR_MODULE_SIZE_14	14
GS1DATABAR_MODULE_SIZE_15	15
GS1DATABAR_MODULE_SIZE_16	16

B.1.8 printGS1DataBarExpandedStacked



(1) Height and width of the barcode image

Height of the barcode image[*1] = ((34 + 3) × number of row[*2] + 34) × module size value

*1: Height of the barcode image = Length from the top of the barcode to the reference point
*2: The number of row is determined by the barcode data.

Width of the barcode image = (4 + 49 × *column* / 2) × module size value

Module Size Value

moduleSize	**Module Size Value**
GS1DATABAR_MODULE_SIZE_2	2
GS1DATABAR_MODULE_SIZE_3	3
GS1DATABAR_MODULE_SIZE_4	4
GS1DATABAR_MODULE_SIZE_5	5
GS1DATABAR_MODULE_SIZE_6	6
GS1DATABAR_MODULE_SIZE_7	7
GS1DATABAR_MODULE_SIZE_8	8
GS1DATABAR_MODULE_SIZE_9	9
GS1DATABAR_MODULE_SIZE_10	10
GS1DATABAR_MODULE_SIZE_11	11
GS1DATABAR_MODULE_SIZE_12	12
GS1DATABAR_MODULE_SIZE_13	13
GS1DATABAR_MODULE_SIZE_14	14
GS1DATABAR_MODULE_SIZE_15	15
GS1DATABAR_MODULE_SIZE_16	16