SII

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

Rev.10

[Products]

RP-F10 Series

RP-G10 Series

Seiko Instruments Inc.

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

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.

SII reserves the right to make changes without notice to the specifications and materials contained herein and shall not be responsible for any damages (including consequential) caused by reliance on the materials presented, including but not limited to typographical, arithmetic, or listing errors.

Introduction

This manual describes "SII Print Class Library for Android^{TM}" (hereinafter referred to as "SII print class library") provided by Seiko Instruments Inc. (hereinafter referred to as "SII").

Target Printers

The printers supported by SII print class library are listed below.

Printers	Interface
RP-F10 Series	Bluetooth
	USB
	TCP/IP
RP-G10 Series	USB

Terms

The terms used in this manual are defined as below.

Terms	Description
Technical Reference	Technical References shown as follows: ・RP-F10 SERIES THERMAL PRINTER TECHNICAL REFERENCE ・RP-G10 SERIES THERMAL PRINTER TECHNICAL REFERENCE
Printer command	Command for controlling the printer described in "Technical Reference"
User's Guide	User's Guides shown as follows: ・RP-F10 SERIES Thermal Printer USER'S GUIDE ・RP-G10 SERIES Thermal Printer USER'S GUIDE

Table of Contents

Chapter 5 Sample Program 5-1

Appendix A Character Set A-1

Appendix B Barcode Size List B-1

Chapter 1
Product Overview

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

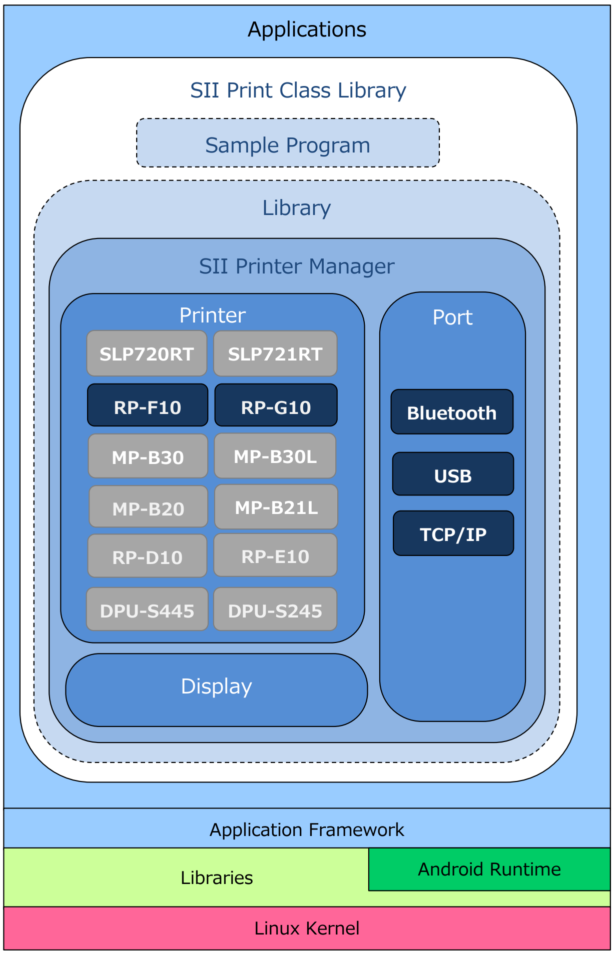
1.1 Functions Provided by SII Print Class Library

The SII print class library including the library and the sample program provides the functions to use SII printer described in "Introduction Target Printers" in Android applications.
Moreover, the SII print class library provides the library sample program in Android Studio project.

1.2 SII Print Class Library Overview

1.2.1 SII Print Class Library Configuration

The library and sample program in the 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, USB or TCP/IP) on an Android device.
Also, the applications can get the 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])
- Cutting paper
- 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 or TCP/IP
- Printer hardware reset
- Drawer operation control
- Buzzer beeping control
- Screen display control[*2]
- Registering a barcode scanner call back function
- Outputting a log file

[*1] Commands that read responses from the printer are not supported. In order to read responses from the printer, use **getStatus** or **getPrinterResponse**.

[*2] For screen display control, see "SII Print Class Library for Android™ Application Programmer's Guide" for DSP-A01 series.

(NOTE) **･RP-F10/G10 do not support the APIs relating to label printing function.**
･RP-G10 does not support the APIs relating to Display or the barcode scanner.
･See "RP-F10 SERIES Thermal Printer USER'S GUIDE" for details of the recommended barcode scanner and the barcode scanner setting.

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.

Printer	Model		RP-F10		
					RP-G10
	Communication Interface		Bluetooth	TCP/IP	USB
Android Device	Communication Port		Bluetooth[*1]	TCP/IP[*2]	USB[*3]
	OS	Android 7.0 (API 24)	Supported	Supported	Supported
		Android 7.1 (API 25)			
		Android 8.0 (API 26)			
		Android 8.1 (API 27)			
		Android 9.0 (API 28)			
		Android 10.0 (API 29)			
		Android 11.0 (API 30)			
		Android 12.0 (API 31)			
		Android 12.1 (API 32)			
		Android 13.0 (API 33)			
		Android 14.0 (API 34)			
Supported Language			Japanese, English		

*1: Bluetooth connection needs to be established by SPP (Serial Port Profile).

*2: The wireless LAN access point that the Android device is connected and the printer need to be connected to the same network.

*3: The Android device needs to support USB host function.

2.2 Printer Settings

Set the memory switches of the printer to [Value] in the following table when using the library.
See "User's Guide" for details about the memory switches and the factory default settings.

- For Bluetooth connection

MS	Function	Value
5-1	Automatic Status Response Selection (Auto Status Back)	0 : Enable
5-2	Initialized Response Selection (Init. Response)	0 : Enable
5-3	Data Discard Selection When Error Occurs (Error Through)	0 : Enable
5-4	Data Discard Selection When Output Buffer Full Occurs (Response Data Discarding)	1 : Disable
13-3	Realtime Command Selection (Realtime Command)	1 : Enable
38-1	Scanner Automatic Status Response Selection*1 (Scanner Auto Status Back)	0 : Enable*2

*1: The firmware of the printer to support the barcode scanner is Ver.1.10 or later.

*2: Select "Enable" when using the barcode scanner.
See "RP-F10 SERIES Thermal Printer USER'S GUIDE" for details about the combination of peripherals.

- For USB connection

MS	Function	Value
5-1	Automatic Status Response Selection (Auto Status Back)	0 : Enable
5-2	Initialized Response Selection (Init. Response)	0 : Enable
5-3	Data Discard Selection When Error Occurs (Error Through)	0 : Enable
5-4	Data Discard Selection When Output Buffer Full Occurs (Response Data Discarding)	1 : Disable
13-3	Realtime Command Selection (Realtime Command)	1 : Enable
38-1	Scanner Automatic Status Response Selection*1 (Scanner Auto Status Back)	0 : Enable*2

*1: The firmware of the printer to support the barcode scanner is Ver.1.10 or later.

*2: Select "Enable" when using the barcode scanner.
See "RP-F10 SERIES Thermal Printer USER'S GUIDE" for details about the combination of peripherals.

- For TCP/IP connection

MS	Function	Value
5-1	Automatic Status Response Selection (Auto Status Back)	0 : Enable
5-2	Initialized Response Selection (Init. Response)	0 : Enable
5-3	Data Discard Selection When Error Occurs (Error Through)	0 : Enable
5-4	Data Discard Selection When Output Buffer Full Occurs (Response Data Discarding)	1 : Disable
13-3	Realtime Command Selection (Realtime Command)	1 : Enable
38-1	Scanner Automatic Status Response Selection*1 (Scanner Auto Status Back)	0 : Enable*2

*1: The firmware of the printer to support the barcode scanner is Ver.1.10 or later.

*2: Select "Enable" when using the barcode scanner.
See "RP-F10 SERIES Thermal Printer USER'S GUIDE" for details about the combination of peripherals.

2.3 Precautions

Communication ports cannot be shared with the printer driver and other libraries when using TCP/IP.

Chapter 3
How to Use Library

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

3.1 Android Application Development Environment

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

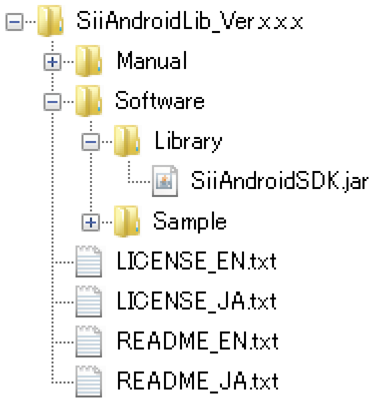
- USB driver for Windows (When developing in Windows environment)

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

The description in and after this chapter is on the premise that the environment where each tool is available is prepared.

3.2 Provided Files

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



```
SiiAndroidLib_Verx.x.x
    Manual
    Software
        Library
            SiiAndroidSDK.jar
        Sample
    LICENSE_EN.txt
    LICENSE_JA.txt
    README_EN.txt
    README_JA.txt
```

Figure 3-1

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

3.3 Build Library into Android Studio Project

This section describes how to build the SII print class library into Android Studio project.

See "Chapter 5 Sample Program" for the sample program included in the 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.

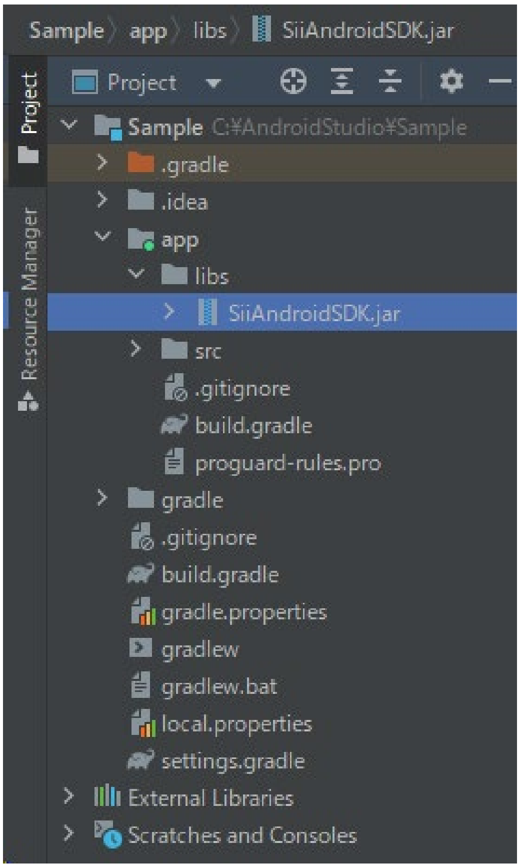


Figure 3-2

(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"/>
```

[When using TCP/IP]

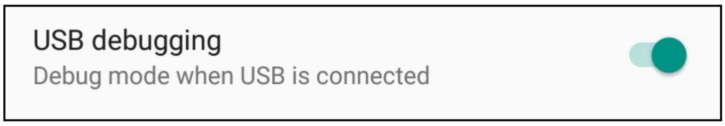
```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
```

By completing these procedures, the library function becomes available.

3.4 Use Developed Android Application on Android Device

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

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



USB debugging
Debug mode when USB is connected

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 Library

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

4.1 Standard Mode and Page Mode

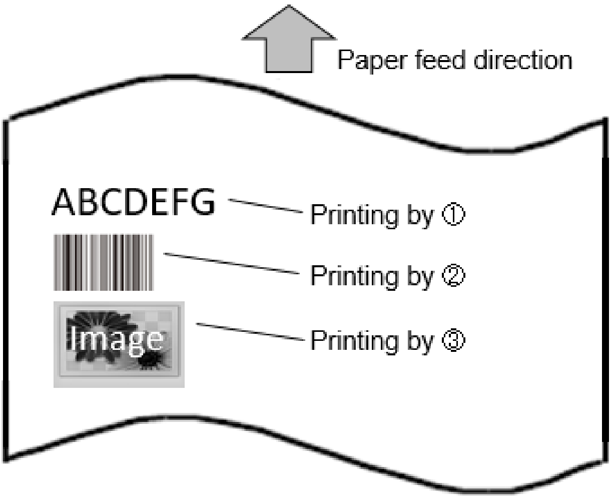
4.1.1 Basic Operation

There are two printing modes "Standard mode" and "Page mode" in the library.
The "Standard mode" and "Page mode" are described below.

(1) Standard mode

Standard mode is the mode to perform the printing in sequence.

Sample print command
① Send text data
② Print barcode
③ Send specified file (Specify an image file)



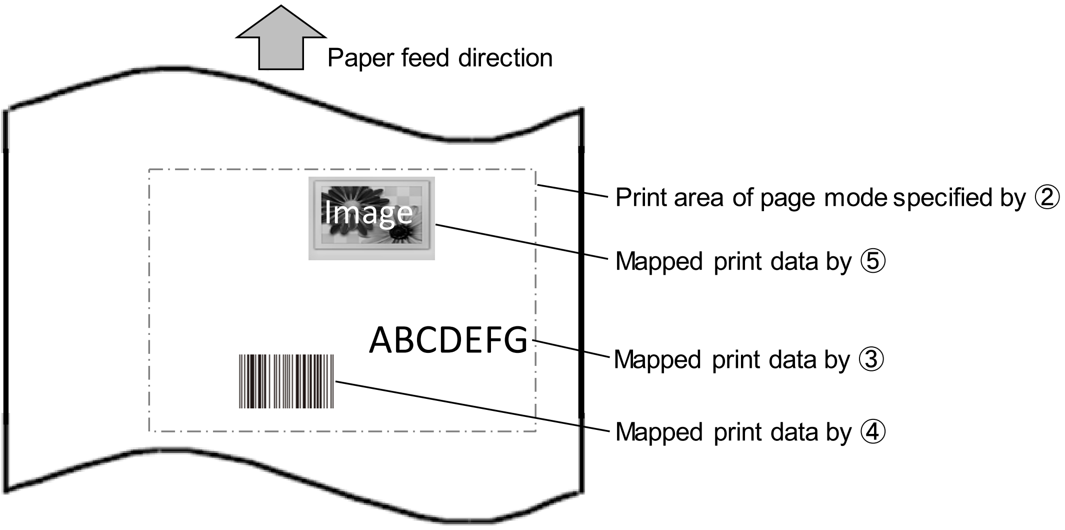
Standard mode suits the printing with an unfixed length such as a receipt.

(2) Page mode

Page mode is the mode to perform the printing on a per-page basis.

In page mode, the print area of page mode is allocated at first, and then print data is mapped on an arbitrary position of the print area.
The mapped print data is printed by the print method of page mode.

Sample print command
① Start page mode
② Specify print area of page mode
③ Send text data of page mode
④ Print barcode of page mode
⑤ Draw image file of page mode
⑥ Print page mode (print the data of ③④⑤ on the print area of ②)
⑦ End page mode



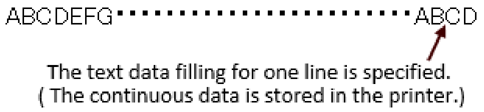
Page mode suits the printing for the followings.
・The printing with a fixed length.
・The printing with the coordinate determination of the character starting position or the ruled line printing position.

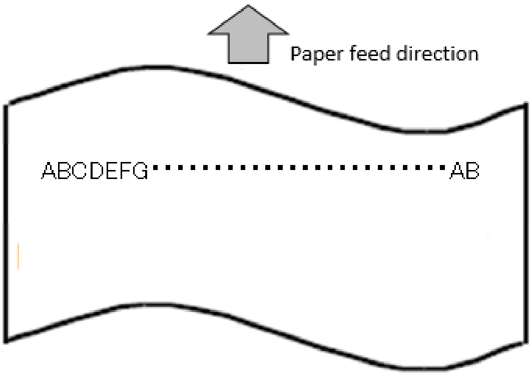
4.1.2 Text Data Printing in Standard Mode

The text data in standard mode is printed each one line.

The text data is stored in the printer when the text data less than one line is specified.
The stored text data is printed by either the following conditions.
・The text data filling for one line is specified.
・Line Feed code is specified.

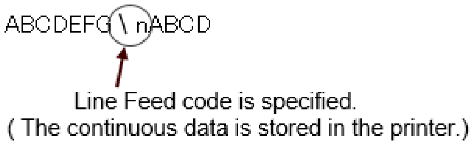
- **The print process when the text data filling for one line is specified.**

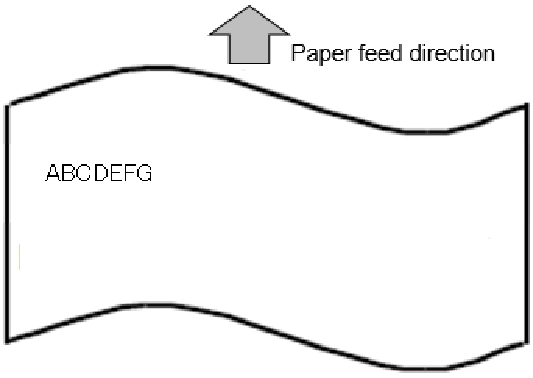

ABCDEFG・・・・・・・・・・・・・・・・・・・・・・・・ABCD
The text data filling for one line is specified.
(The continuous data is stored in the printer.)



Input data **Print result**

- **The print process when Line Feed code is specified.**


ABCDEFG\nABCD
Line Feed code is specified.
(The continuous data is stored in the printer.)



Input data **Print result**

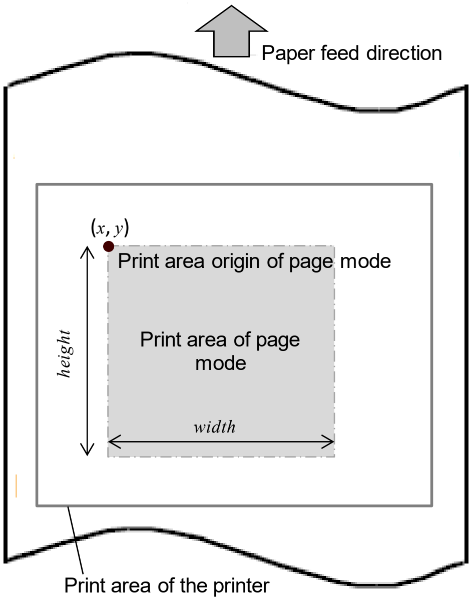
4.1.3 Mapping Position of Print Data in Page Mode

In page mode, the mapping position of print data is determined by print area, print direction, and reference point.
This section describes the print area, print direction, and reference point.

(1) Print area of page mode

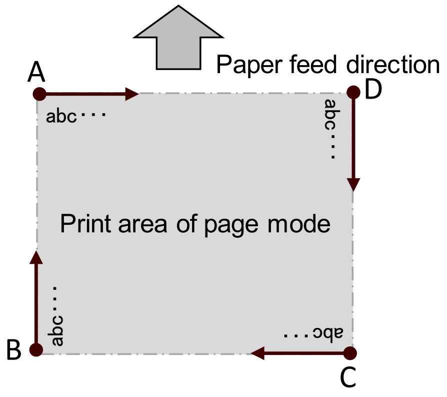
The print area of page mode is specified against the print area of the printer by the print area origin, and the width and the height of page mode. The view of the print area is shown in the following figures.

The print area of page mode can be specified more than one.



(2) Print direction

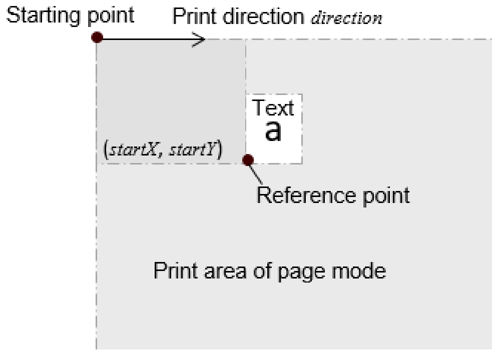
Specify the print direction at setting the print area of page mode.
The starting point is changed depending on specifying the print direction for each direction.
The relation between the print direction and the starting point is shown in the figure below.

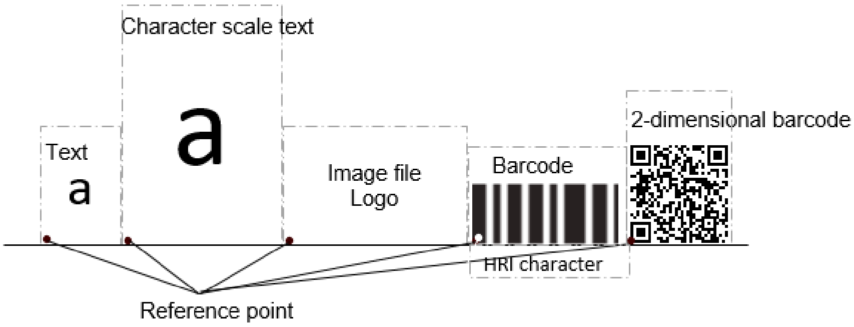


・Starting point: Upper left (A on the figure), Print direction: Left to Right
・Starting point: Left below (B on the figure), Print direction: Below to Upper
・Starting point: Right below (C on the figure), Print direction: Right to Left
・Starting point: Upper right (D on the figure), Print direction: Upper to Below

(3) Reference point

The relation between the reference point for mapping data and each print element (text, image file, logo, and barcode, etc.) is shown in the figures below.





(NOTE) The reference point cannot be specified out of the print area of page mode.

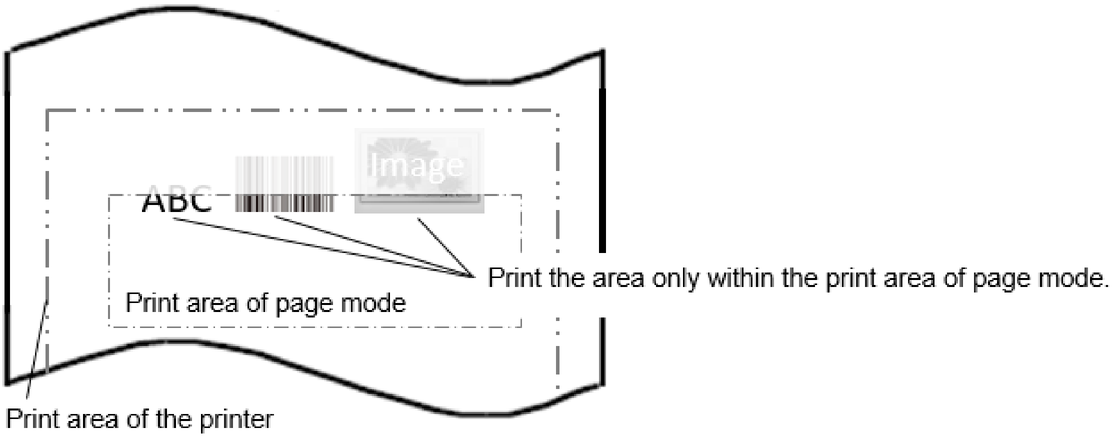
4.1.4 Print Data Process at Out of Print Area of Page Mode

This section describes the process when mapped data is to be mapped on out of the print area of page mode.

Type of Print Data

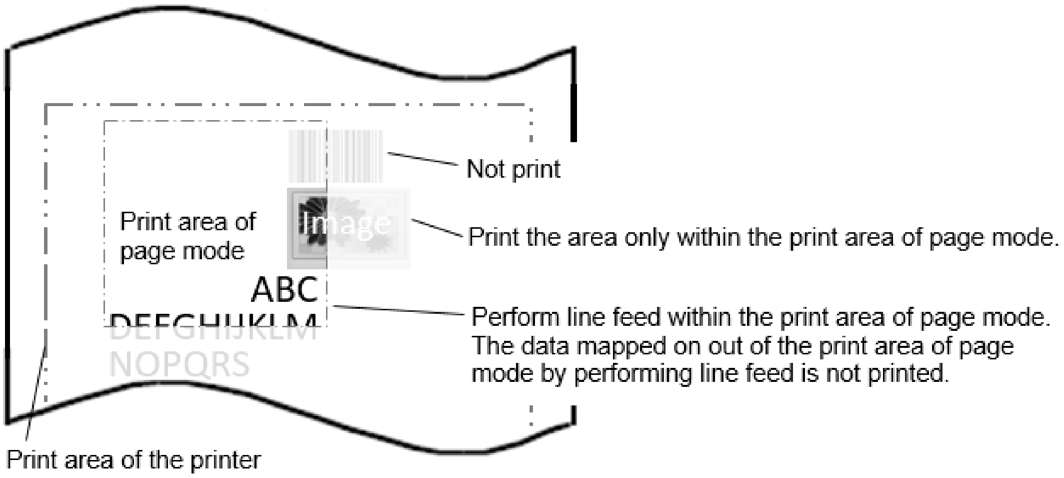
Text	Barcode, 2-dimensional Barcode	Image File, Logo, Rectangle, Ruled Line
ABC		

(1) The print data is mapped on the upper of the print area of page mode.



ABC
Image
Print the area only within the print area of page mode.
Print area of page mode
Print area of the printer

(2) The print data is mapped on the right of print area of page mode.



Not print
Print area of page mode
Image
Print the area only within the print area of page mode.
ABC
DEFGHIJKLM
NOPQRS
Perform line feed within the print area of page mode.
The data mapped on out of the print area of page mode by performing line feed is not printed.
Print area of the printer

(NOTE)	**Read error or incorrect reading may occur when the part of mapped barcode data is on out of the print area of page mode.**

4.2 Log File Output Function

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

4.2.1 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 "4.2.2 Log Output Settings" for details on the settings for x.

4.2.2 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

4.2.3 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.

4.3 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 APIs used for communication with the printer and for printing. See "**4.3.1 PrinterManager Class**" for more details.	✓
PrinterEvent	Provides the API that gets the end event when **startDiscoveryPrinter** is terminated. See "**4.3.2 PrinterEvent Class**" for more details.	✓
PrinterListener	Interface for getting the end event when **startDiscoveryPrinter** is terminated. See "**4.3.3 PrinterListener Interface**" for more details.	✓
PrinterInfo	Stores the printer information found by **startDiscoveryPrinter**. See "**4.3.4 PrinterInfo Class**".	✓
PrinterException	Exception class that is thrown at API call. See "**4.3.5 PrinterException Class**" for more details.	✓
CallbackFunctionListener	Interface for getting the change event of printer status. See "**4.3.6 CallbackFunctionListener Interface**" for more details.	✓
BarcodeScannerListener	Interface for getting barcode scanner connection or barcode scanner disconnection, or received barcode data. See "**4.3.7 BarcodeScannerListener Interface**" for more details.	✓[*2]
SmartLabelManager	Provides the API to specify label files or replace data.	−

[*1] ✓: Supported, -: Not supported in RP-F10/G10
[*2] Supported only by RP-F10

(NOTE)
- **RP-F10/G10 do not support the APIs relating to label printing function.**
- **RP-G10 does not support the APIs relating to Display or the barcode scanner.**

4.3.1 **PrinterManager** Class

(1) Method List

Methods provided by the **PrinterManager** class are shown in the following table.
"Standard mode" or "Page mode" can be selected in the **PrinterManager** class.

Method	Description
Common method to standard mode and page mode	The valid methods in standard mode and page mode. See "4.3.1(1)① Common method to standard mode and page mode" for the methods.
Dedicated method for standard mode	The valid methods in standard mode. See "4.3.1(1)② Dedicated method for standard mode" for the methods.
Dedicated method for page mode	The valid methods in page mode. See "4.3.1(1)③ Dedicated method for page mode" for the methods.

① Common method to standard mode and page mode

Methods provided by the common method to standard mode and page mode are shown in the following table. See "4.3.1(4)① Common method to standard mode and page mode" for details of the common methods.

Name	Description	Supported [*1]
PrinterManager	Constructor	✓
connect	Start communicating with printer (Bluetooth)	✓
connect	Start communicating with printer (USB)	✓
connect	Start communicating with printer (TCP/IP)	✓
disconnect	Stop communicating with printer	✓
setBarcodeScannerListener	Start/End callback of barcode scanner	✓[*2]
openDrawer	Open cash drawer	✓
buzzer	Sound buzzer	-
externalBuzzer	Sound external buzzer	✓
getStatus	Get printer status	✓
setCallbackFunctionListener	Start/End callback of printer status change	✓
abort	Abort waiting state of printer	✓
registerLogo	Register logo	✓
unregisterLogo	Delete registered logo	✓
registerStyleSheet	Register style sheet	-
unregisterStyleSheet	Delete registered style sheet	-
resetPrinter	Reset printer	✓
getPrinterResponse	Get various responses from printer	✓
startDiscoveryPrinter	Start printer search (Bluetooth)	✓
startDiscoveryPrinter	Start printer search (USB)	✓

Name	Description	Supported [*1]
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 RP-F10/G10

*2: Supported only by RP-F10

② Dedicated method for standard mode

Methods provided by the dedicated method for standard mode are shown in the following table. See "4.3.1(4)② Dedicated method for standard mode" for details of the dedicated methods.

Name	Description	Supported [*1]
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	✓
feedPosition	Paper form feed	−
sendBinary	Send binary data	✓
sendDataFile	Send specified file	✓

Name	Description	Supported *1
printPDF	Print PDF page	✓
printLogo	Print logo	✓

*1: ✓: Supported, -: Not supported in RP-F10/G10

③ Dedicated method for page mode

Methods provided by the dedicated method for page mode are shown in the following table.
See "4.3.1(4)③ Dedicated method for page mode" for details of the dedicated methods.

Name	Description	Supported *1
enterPageMode	Start page mode	✓
exitPageMode	End page mode	✓
setPageModeArea	Specify print area of page mode	✓
setPageModeDirection	Specify print direction of page mode	✓
setPageModeLineSpacing	Specify line spacing of page mode	✓
printPageMode	Print page mode	✓
printPageModeText	Send text data of page mode	✓
printPageModeTextEx	Send format specified text data of page mode	✓
printPageModeBarcode	Print barcode of page mode	✓
printPageModePDF417	Print PDF417 of page mode	✓
printPageModeQRcode	Print QR Code of page mode	✓
printPageModeDataMatrix	Print Data Matrix of page mode	✓
printPageModeMaxiCode	Print MaxiCode of page mode	✓
printPageModeGS1DataBarStacked	Print GS1 Databar Stacked of page mode	✓
printPageModeGS1DataBarStackedOmnidirectional	Print GS1 Databar Stacked Omni-directional of page mode	✓
printPageModeGS1DataBarExpandedStacked	Print GS1 Databar Expanded Stacked of page mode	✓
printPageModeAztecCode	Print Aztec Code of page mode	−
sendPageModeBinary	Send binary data of page mode	✓
printPageModeImageFile	Draw image file of page mode	✓
printPageModeRectangle	Draw rectangle image of page mode	✓
printPageModeLine	Print ruled line of page mode	✓
printPageModeLogo	Print logo of page mode	✓

*1: ✓: Supported, -: Not supported in RP-F10/G10

(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_RP-FG10	RP-F10/G10	301
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 of user area	1
PRINTER_RESPONSE_ARRANGE_USER_AREA	Send remaining capacity of 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

③ 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

④ Codepage

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

Constant Name	Description	Value
CODE_PAGE_437	USA, Standard Europe (Code Page437)	0
CODE_PAGE_KATAKANA	Katakana	1
CODE_PAGE_850	Multilingual (Code Page850)	2
CODE_PAGE_860	Portuguese (Code Page860)	3
CODE_PAGE_863	Canadian-French (Code Page863)	4
CODE_PAGE_865	Nordic (Code Page865)	5
CODE_PAGE_857	Turkish (Code Page857)	13
CODE_PAGE_737	Greek (Code Page737)	14
CODE_PAGE_1252	Latin (Code Page1252)	16
CODE_PAGE_866	Russian (Code Page866)	17
CODE_PAGE_852	Eastern Europe (Code Page852)	18
CODE_PAGE_858	Euro (Code Page858)	19
CODE_PAGE_855	Cyrillic (Code Page855)	34
CODE_PAGE_864[*1]	Arabic (Code Page864)	37
CODE_PAGE_1250	Central European (Code Page1250)	45
CODE_PAGE_1251	Cyrillic (Code Page1251)	46
CODE_PAGE_1253	Greek (Code Page1253)	47
CODE_PAGE_1254	Turkish (Code Page1254)	48

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

⑤ 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
PRINTER_TYPE_TCP	TCP/IP	2

⑥ 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
PDF417_COLUMN_AUTO	Automatic selection of the number of columns	0

(3) Enumerated Constant List

① Drawer number (DrawerNum)

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

Constant Name	Description
DRAWER_1	Drawer 1
DRAWER_2	Drawer 2

② Pulse width (PulseWidth)

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

Constant Name	Description
ON_OFF_TIME_100	ON/OFF time 100 milliseconds
ON_OFF_TIME_200	ON/OFF time 200 milliseconds
ON_OFF_TIME_300	ON/OFF time 300 milliseconds
ON_OFF_TIME_400	ON/OFF time 400 milliseconds
ON_OFF_TIME_500	ON/OFF time 500 milliseconds
ON_OFF_TIME_600	ON/OFF time 600 milliseconds
ON_OFF_TIME_700	ON/OFF time 700 milliseconds
ON_OFF_TIME_800	ON/OFF time 800 milliseconds

③ Buzzer pattern (BuzzerPattern)

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

Constant Name	Description
BUZZER_PATTERN_1	Pattern 1
BUZZER_PATTERN_2	Pattern 2
BUZZER_PATTERN_3	Pattern 3
BUZZER_PATTERN_4	Pattern 4

④ 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

⑤ 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

⑥ 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

⑦ 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

⑧ 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

⑨ 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

⑩ 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)

⑪ Character scale (CharacterScale)

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

Constant Name	Description
VARTICAL_1_HORIZONTAL_1	Height × 1 and width × 1
VARTICAL_1_HORIZONTAL_2	Height × 1 and width × 2
VARTICAL_1_HORIZONTAL_3	Height × 1 and width × 3
VARTICAL_1_HORIZONTAL_4	Height × 1 and width × 4
VARTICAL_2_HORIZONTAL_1	Height × 2 and width × 1
VARTICAL_2_HORIZONTAL_2	Height × 2 and width × 2
VARTICAL_2_HORIZONTAL_3	Height × 2 and width × 3
VARTICAL_2_HORIZONTAL_4	Height × 2 and width × 4
VARTICAL_2_HORIZONTAL_6	Height × 2 and width × 6
VARTICAL_3_HORIZONTAL_1	Height × 3 and width × 1
VARTICAL_3_HORIZONTAL_2	Height × 3 and width × 2
VARTICAL_3_HORIZONTAL_3	Height × 3 and width × 3
VARTICAL_3_HORIZONTAL_4	Height × 3 and width × 4
VARTICAL_4_HORIZONTAL_1	Height × 4 and width × 1
VARTICAL_4_HORIZONTAL_2	Height × 4 and width × 2
VARTICAL_4_HORIZONTAL_3	Height × 4 and width × 3
VARTICAL_4_HORIZONTAL_4	Height × 4 and width × 4
VARTICAL_4_HORIZONTAL_6	Height × 4 and width × 6
VARTICAL_4_HORIZONTAL_8	Height × 4 and width × 8
VARTICAL_6_HORIZONTAL_2	Height × 6 and width × 2
VARTICAL_6_HORIZONTAL_4	Height × 6 and width × 4
VARTICAL_6_HORIZONTAL_6	Height × 6 and width × 6
VARTICAL_6_HORIZONTAL_8	Height × 6 and width × 8
VARTICAL_8_HORIZONTAL_4	Height × 8 and width × 4
VARTICAL_8_HORIZONTAL_6	Height × 8 and width × 6
VARTICAL_8_HORIZONTAL_8	Height × 8 and width × 8

⑫ 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

⑬ 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

⑭ 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 **printBarcode** or **printPageModeBarcode** for details of syntax.

⑮ 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** ● **printPageModeBarcode**
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** ● **printPageModePDF417**
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** ● **printPageModeQRcode**
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** ● **printPageModeDataMatrix**
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** ● **printPageModeGS1DataBarStacked** ● **printPageModeGS1DataBarStackedOmnidirectional** ● **printPageModeGS1DataBarExpandedStacked**
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	

⑯ 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)

⑰ 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

⑱ 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** ● **printPageModePDF417**
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** ● **printPageModeQRcode**
QR_ERROR_CORRECTION_M	Error correction level M	
QR_ERROR_CORRECTION_H	Error correction level H	
QR_ERROR_CORRECTION_Q	Error correction level Q	

⑲ 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

⑳ QR Code Model (QrModel)

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

Constant Name	Description
QR_MODEL_2	QR Code Model 2

㉑ 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

㉒ 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

㉓ Cutting method (CuttingMethod)

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

Constant Name	Description	
	Paper Feed to Cut Position	Cutting Method
CUT_FULL	Enabled	Full cut
CUT_FULL_NO_FEED	Disabled	
CUT_PARTIAL	Enabled	Partial cut
CUT_PARTIAL_NO_FEED	Disabled	
CUT_NONE[*1]	Disabled	No cut

*1: Supported only by **printPageMode**.

㉔ 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

㉕ Image scaling (ImageScale)

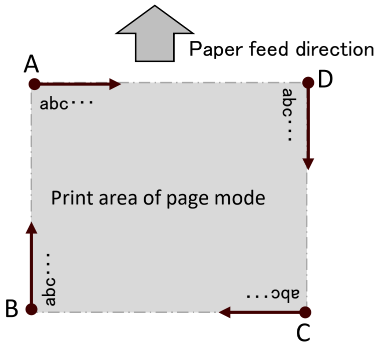
Constants of enumerated type used for image scaling are shown in the following table.

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

㉖ Print direction (Direction)

Constants of enumerated type used for the print direction in page mode are shown in the following table.

Constant Name	Description
DIRECTION_LEFT_TO_RIGHT	Starting point: Upper left (A on the figure), Print drectioin: Left to Right
DIRECTION_BOTTOM_TO_TOP	Starting point: Left below (B on the figure), Print drectioin: Below to Upper
DIRECTION_RIGHT_TO_LEFT	Starting point: Right below (C on the figure), Print drectioin: Righ to Left
DIRECTION_TOP_TO_BOTTOM	Starting point: Upper right (D on the figure), Print drectioin: Upper to Below



㉗ Line style (LineStyle)

Constants of enumerated type used for the line style in page mode are shown in the following table.

Constant Name	Description
LINESTYLE_THIN	Thin solid line (2 dots)
LINESTYLE_MEDIUM	Medium solid line (4 dots)
LINESTYLE_THICK	Thick solid line (8 dots)

(4) Method Details

① Common method to standard mode and page mode

The following methods are valid in standard mode and page mode. Standard mode is set immediately after **connect** is executed.

PrinterManager — Constructor

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

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

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

connect — Start communicating with printer (Bluetooth)

Starts communication with a printer by Bluetooth connection.
The method of syntax (a) always communicates with a printer in secure mode.
The method of syntax (b) communicates with a printer by specifying secure mode or 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 "4.3.1(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 this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Call this method before using other **Printer Manager** 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 or barcode scanner 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**.
The receipt of the barcode data from the barcode scanner can be notified as an event by **onBarcodeScannerReadData** and **setBarcodeScannerListener.**
The connection of the barcode scanner can be notified as an event by **onBarcodeScannerChangedOnline** and **setBarcodeScannerListener**.
The disconnection of the barcode scanner can be notified as an event by **onBarcodeScannerChangedOffline** and **setBarcodeScannerListener**.

connect Start communicating with printer (USB)

Starts communication with a printer by USB connection.

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

Parameter *printerModel* Printer model constant for USB connection
See "4.3.1(2)① Printer model" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Call this method before using other **Printer Manager** 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 or barcode scanner 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**.
The receipt of the barcode data from the barcode scanner can be notified as events by **onBarcodeScannerReadData** and **setBarcodeScannerListener.**
The connection of the barcode scanner can be notified as an event by **onBarcodeScannerChangedOnline** and **setBarcodeScannerListener**.
The disconnection of the barcode scanner can be notified as an event by **onBarcodeScannerChangedOffline** and **setBarcodeScannerListener**.

connect Start communicating with printer (TCP/IP)

Starts communication with a printer by TCP/IP connection.

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

Parameter *printerModel* Printer model constant for Ethernet connection
See "4.3.1(2)① Printer model" for available constants.

address IP address
Example: "192.168.0.190"

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

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

Starts communication with a printer connected to the same network as the Android device by TCP/IP connection.
Connects to the IP address specified by *address*. TCP ports 9100 and 26100 are used for communication. Also, printer initial setting is performed at the connection based on *printerModel* specified.

- **Creating/discarding of socket in TCP/IP connection of the library**

  After **connect**, the library retains the created socket until **disconnect**.
  And connecting to the same printer from other applications is not possible until **disconnect**.
  Based on the completion of data transmission to the printer, the socket is once discarded after elapsing the socket keeping time set by **setSocketKeepingTime**. Then the new socket is created immediately and used for the next connection.

Monitoring of the printer status or barcode scanner 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**.
The receipt of the barcode data from the barcode scanner can be notified as events by **onBarcodeScannerReadData** and **setBarcodeScannerListener.**
The connection of the barcode scanner can be notified as an event by **onBarcodeScannerChangedOnline** and **setBarcodeScannerListener**.
The disconnection of the barcode scanner can be notified as an event by **onBarcodeScannerChangedOffline** and **setBarcodeScannerListener**.

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 this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description This method discards the print data kept by **controlTransaction**.
The instance of **CallbackFunctionListener** interface kept by **setCallbackFunctionListener** is discarded and the callback is stopped.
The instance of **BarcodeScannerListener** interface kept by **setBarcodeScannerListener** is discarded and also callback is stopped.

Note It is recommended to get the execution response by **PRINTER_RESPONSE_REQUEST** of **getPrinterResponse** 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

Starts or ends callback of barcode scanner.
Supported only by RP-F10.

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

Parameter *listener* Instance of **BarcodeScannerListener** interface

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description The targeting callbacks are as follows.

- ･Receipt of barcode data
- ･Connection of barcode scanner
- ･Disconnection of barcode scanner

When specify the instance of **BarcodeScannerListener** 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 **BarcodeScannerListener** 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.

openDrawer Open cash drawer

Opens the specified cash drawer.

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

Parameter *drawerNum* Drawer number
See "4.3.1(3)① Drawer number (DrawerNum)" for available constants.

onOffTime Pulse width
See "4.3.1(3)② Pulse width (PulseWidth)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

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

Sounds the external buzzer.

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

Parameter *buzzerPattern* Buzzer pattern
See "4.3.1(3)③ Buzzer pattern (BuzzerPattern)" for available constants.
The external buzzer sound stops under one of the following conditions:

- ･Sounding for the number of times set by *buzzerCount*
- ･Opening the cover
- ･Executing the printer command "Stop External Buzzer"

buzzerCount Buzzer sound count (times)
The external buzzer sounds for the number of times set by *buzzerCount*.
The valid range is 1 to 255.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

getStatus Get printer status

Gets the latest printer status.
The method of syntax (a) returns the printer status with return value.
The method of syntax (b) stores the printer status in an array of int type.

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 this method is being called.
See "**4.3.5 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 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	Autocutter error	No error	Error
4	Out-of-paper error	No error	Error
5	Reserved	Fixed	-
6	Reserved	Fixed	-
7	Cover open error	No error	Error
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	Drawer switch input status	Low	High
16	FLASH memory rewriting	Not rewriting	Rewriting
17	Peripheral device selection	Printer	Others

Bit	Function	Value	
		0	1
18	Reserved	Fixed	-
19 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 this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

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

The keeping 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 this method is being called.
See "**4.3.5 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 file by **sendDataFile** is aborted, the printer does not accept other processes until the specified image file is received completely. (Methods and transmission data are misinterpreted and recognized as a part of the image file.) To solve this situation, use this method to abort the waiting state of the printer.
Note that when this method is executed, a part of unprinted image file may be printed.

registerLogo Register logo

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

Syntax

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

(a) public void registerLogo(String fileName,
                             String id,
                             Dithering dithering) 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 supporting image file are .bmp, .jpg, .jpeg, and .png. When the image file 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 "4.3.1(3)④ Dithering (Dithering)" for available constants.

Exception

PrinterException
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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 this method is being called.
See "**4.3.5 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*fileName*, 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 this method is being called.
See "**4.3.5 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.

For TCP/IP connection:
The reset is performed to the connected printer by our proprietary command (reset request) to TCP port 26100.

The connection with the printer is retained even after this method is executed.

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 "4.3.1(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.
PRINTER_RESPONSE_ARRANGE_USER_AREA (Send remaining capacity of user area 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 stored as a string array. Example: *buf.size()* = 3, *buf[0]* = "22", *buf[1]* = "23", *buf[2]* = "24", etc.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

Syntax public void **startDiscoveryPrinter**(PrinterListener *listener*) 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*.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

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 this method is being called.
See "**4.3.5 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)

Searches for SII printer connecting to the same network.

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

retry Retry count (times)
Sends the local broadcast packet the number of times set by *retry*.
The valid range is 1 to 5.
When the value is specified less than 1, the number is set to 1.
When the value is specified more than 5, the number is set to 5.

timeout Search timeout period (millisecond: ms)
Sets the timeout period per search. Each time the local broadcast packet is sent, this method waits for a response from the printer until the period specified by *timeout* elapses.
The valid range is 3000 to 60000.
When the value is specified less than 3000, the period is set to 3000 ms.
When the value is specified more than 60000, the period is set to 60000 ms.

Error **PrinterException** is thrown when an error occurs while this method is being called.

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

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.
When the value out of the valid range is specified, the value is set to 10000 ms.

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.
When the value out of the valid range is specified, the value is set to 10000 ms.

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 "4.3.1(2)③ International character set" for details of the value.

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

Character codes with the varying print result depending on the configuration of the international character:
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 "4.3.1(2)③ 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 than Japanese:
COUNTRY_USA

getCodePage — Get codepage

Gets the value of codepage.

Syntax public int **getCodePage**()

Return value See "4.3.1(2)④ Codepage" for details of the value.

Description The encoder used for sending the text data by **sendText**, **sendTextEx**, **sendDataFile,** **printPageModeText**, or **printPageModeTextEx** 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 "4.3.1(2)④ Codepage" for the values available for setting.
When an invalid value is specified, 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 than Japanese:
CODE_PAGE_1252

getPrinterModel — Get printer model

Gets the value of the connecting printer model.

Syntax public int **getPrinterModel**()

Return value See "4.3.1(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 "4.3.1(2)⑤ 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

Gets the socket keeping time.

Syntax public int **getSocketKeepingTime**()

Return value Socket keeping time (millisecond: ms)

Description This method can get the socket time keeping time regardless of whether **isConnect** is true or false.

setSocketKeepingTime — Set socket keeping time

Sets the socket keeping time.

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

Valid range 60000 to 300000 (millisecond: ms)
When the value is specified less than 60000, the time is set to 60000 ms.
When the value is specified more than 300000, the time is set to 300000 ms.

Default 300000

Description This method can set the socket time keeping time regardless of whether **isConnect** is true or false.

For the socket keeping time, specify a time equal to Receive Timeout of the printer to be connected. The setting of Receive Timeout can be changed in "SII RP Utility" on Android application for Google Play.

The set socket keeping time becomes effective at the next **connect** execution.

getVersion Get SDK version

Gets the SDK version as a character string.

Syntax public String **getVersion**()

Return value 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 "4.3.1(3)⑤ Batch processing selection (TransactionFunction)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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**
- **cutPaper**
- **openDrawer**
- **externalBuzzer**
- **sendBinary**
- **sendDataFile**
- **printPDF**
- **printLogo**[*1]
- **enterPageMode**
- **exitPageMode**
- **setPageModeArea**
- **setPageModeDirection**
- **setPageModeLineSpacing**
- **printPageMode**
- **printPageModeText**
- **printPageModeTextEx**
- **printPageModeBarcode**
- **printPageModePDF417**
- **printPageModeQRcode**
- **printPageModeDataMatrix**
- **printPageModeMaxiCode**
- **printPageModeGS1DataBarStacked**
- **printPageModeGS1DataBarStackedOmnidirectional**
- **printPageModeGS1DataBarExpandedStacked**
- **sendPageModeBinary**
- **printPageModeImageFile**
- **printPageModeRectangle**
- **printPageModeLine**
- **printPageModeLogo**[*1]

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

② Dedicated method for standard mode

The following methods are valid in standard mode. **PrinterException** is thrown when the dedicated method for standard mode are executed in page mode.

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 this method is being called.
See "**4.3.5 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 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.

sendTextEx — Send format specified text data

Sends format specified text data to the printer.
The method of syntax (a) or (c) outputs the pending data at first and starts processing.
The method of syntax (b) starts processing according to the constants of the pending data output specifying.

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,
                           CharacterReverse reverse,
                           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 "4.3.1(3)⑥ Bold print (CharacterBold)" for available constants.

underline Underline
See "4.3.1(3)⑦ Underline (CharacterUnderline)" for available constants.

reverse Reverse print
See "4.3.1(3)⑧ Reverse print (CharacterReverse)" for available constants.

inversion Inversion print
See "4.3.1(3)⑨ Inversion print (CharacterInversion)" for available constants.

font Font
See "4.3.1(3)⑩ Character font (CharacterFont)" for available constants.

scale Character scale
See "4.3.1(3)⑪ Character scale (CharacterScale)" for available constants.

alignment Alignment
See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

output Pending data output specifying
See "4.3.1(3)⑬ Pending data output specifying (OutputPendingData)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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 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 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.

printBarcode Print barcode

Prints barcode.
The method of syntax (a) specifies the barcode data by character string.
The method of syntax (b) specifies the barcode data by character string and specifies the alignment and N:W ratio of the barcode.
The method of syntax (c) specifies the barcode data by the array of bytes and specifies the alignment of the barcode.
The method of 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 "4.3.1(3)⑭ Barcode symbol (BarcodeSymbol)" for available constants and corresponding syntax.

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.

Barcode	Number of Data	Inputtable Data Character String (Data)	Remarks
CODE93	1 to 150 bytes	(0x00 to 0x2E)	Input data with 0x2F or more at the end.
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 "4.3.1(3)⑮ 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 "4.3.1(3)⑯ HRI character print position (HriPosition)" for available constants.

hriFont
HRI character font
See "4.3.1(3)⑩ Character font (CharacterFont)" for available constants.

alignment
Alignment
See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

nwRatio N:W ratio
See "4.3.1(3)⑰ N:W ratio (NwRatio)" for available constants.
Depending on specified *nwRatio* and *moduleSize*, the wide element width is set as shown 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 this method is being called.
See "**4.3.5 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.

Note The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printPDF417 Print PDF417

Prints PDF417.
The method of syntax (a) specifies PDF417 symbol.
The method of syntax (b) 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 "4.3.1(3)⑱ Error correction level (ErrorCorrection)" for available constants.
row	The number of rows (row) The valid range is 0, 3 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 "4.3.1(3)⑮ 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 "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.
pdf417Symbol	Symbol of PDF417 See "4.3.1(3)⑲ PDF417 symbol (Pdf417Symbol)" for available constants.

Exception **PrinterException**

PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

Note The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printQRcode Print QR Code

Prints QR Code.
The method of syntax (a) is fixed to QR Code Model 2.
The method of syntax (b) specifies QR Code Model.

Syntax

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

(b) public void **printQRcode**(String *text*,
ErrorCorrection *errorCorrection*,
ModuleSize *moduleSize*,
PrintAlignment *alignment*,
QrModel *model*) 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 "4.3.1(3)⑱ Error correction level (ErrorCorrection)" for available constants.
	moduleSize	Module size See "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.
	alignment	Alignment See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.
	model	QR Code Model See "4.3.1(3)⑳ QR Code Model (QrModel)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

Note The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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 "4.3.1(3)㉑ Data Matrix module (DataMatrixModule)" for available constants.
	moduleSize	Module size See "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.
	alignment	Alignment See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

Note The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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 "4.3.1(3)㉒ MaxiCode Mode (MaxiCodeMode)" for available constants.

alignment Alignment
See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

Note The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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 "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

alignment Alignment
See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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 "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

alignment Alignment
See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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 following:
' ', '!', '"', '%', '&', ''', '(', ')', '*', '+', ',', '-', '.', '/', ':', ';', '<', '=', '>', '?', '_', '0' to '9', 'A' to 'Z', 'a' to 'z'.
Enter '{1’ to FNC1.

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 "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.
alignment	Alignment See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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 printAztecCode(String text,
                           int layer,
                           int errorCorrection,
                           ModuleSize moduleSize,
                           AztecSymbol aztecSymbol,
                           PrintAlignment alignment) throws PrinterException
```

cutPaper Cut paper

Selects enabled/disabled of the paper feed to the cut position and cuts the paper.

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

Parameter	*cuttingMethod*	Cutting method See "4.3.1(3)㉓ Cutting method (CuttingMethod)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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
```

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 16 KB (16384 bytes).

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.
The method of syntax (a), dithering is fixed to be disabled.
The method of syntax (b), dithering can be specified.

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

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

Parameter *fileName* File path of the data 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 1 MB (1048576 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 file. Colored image file is converted to monochrome image by binarization and registered. Printing is performed at one time after mapping the image file 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 **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 "4.3.1(3)⑫ 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 "4.3.1(3)④ Dithering (Dithering)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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* Name of the PDF file
The file extension for supporting PDF is .pdf.
A maximum of 1 MB (1048576 bytes) of file size can be specified.
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 "4.3.1(3)㉔ Image rotation direction (Rotate)" for available constants.

imageScale

Image scaling
See "4.3.1(3)㉕ 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 "4.3.1(3)④ Dithering (Dithering)" for available constants.

alignment

Alignment
See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

printLogo Print logo

Prints the registered logo.

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

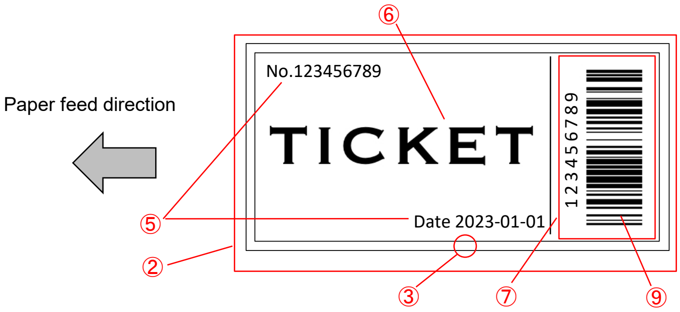
Parameter	*id*	Logo ID to print (key code) Specify the ID of the registered logo as a character string.
	alignment	Alignment See "4.3.1(3)⑫ Alignment (PrintAlignment)" for available constants.

Exception **PrinterException**

PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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.

③ Dedicated method for page mode

The following methods are dedicated methods to use page mode. An example for the print process in page mode is shown below.



① Start page mode

```
mPrinterManager.enterPageMode();
```

② Specify print area of page mode

```
mPrinterManager.setPageModeArea(0, 0, 355, 576);
```

③ Specify a rectangle and a ruled line

```
mPrinterManager.printPageModeRectangle(0, 0, 344, 575, LineStyle.LINESTYLE_THIN);
mPrinterManager.printPageModeRectangle(7, 7, 336, 567, LineStyle.LINESTYLE_THIN);
mPrinterManager.printPageModeLine(11, 404, 334, 404, LineStyle.LINESTYLE_THIN);
```

④ Specify print direction of page mode

```
mPrinterManager.setPageModeDirection(Direction.DIRECTION_TOP_TO_BOTTOM );
```

⑤ Specify a character

```
mPrinterManager.printPageModeText(21, 47, "NO.123456789");
mPrinterManager.printPageModeText(212, 340, "Date 2023-01-01");
```

⑥ Specify an image file

```
mPrinterManager.printPageModeImageFile(
        10,
        222,
        Environment.getExternalStorageDirectory().getPath() + "/TicketImage.jpg",
        Dithering.DITHERING_DISABLE);
```

⑦ Specify print area of page mode

```
mPrinterManager.setPageModeArea(0, 404, 345, 163);
```

⑧ Specify print direction

```
mPrinterManager.setPageModeDirection(Direction.DIRECTION_LEFT_TO_RIGHT);
```

⑨ Specify a barcode

```
mPrinterManager.printPageModeBarcode(
        20,
        132,
        BarcodeSymbol.BARCODE_SYMBOL_CODE128,
        new byte[]{0x67, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19x, 0x68},
        ModuleSize.BARCODE_MODULE_WIDTH_2,
        80,
        HriPosition.HRI_POSITION_ABOVE,
        CharacterFont.FONT_A);
```

⑩ Print in page mode

```
mPrinterManager.printPageMode(CuttingMethod.CUT_PARTIAL);
```

⑪ Ends page mode

```
mPrinterManager.exitPageMode();
```

enterPageMode — Start page mode

Starts page mode.

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

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description This method starts page mode. The dedicated method for page mode and common methods to standard mode and page mode can be used after this method execution.

Executing **exitPageMode** discards the print data kept in the page data buffer and changes the mode to standard mode.
Executing **printPageMode** prints the print data kept in the page data buffer.

exitPageMode — End page mode

Ends page mode and changes the mode to standard mode.

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

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Discards the print data kept in the page data buffer and changes the mode to standard mode.

setPageModeArea — Specify print area of page mode

Specifies print area of page mode.

Syntax

```
public void setPageModeArea(int x,
                            int y,
                            int width,
                            int height) throws PrinterException
```

Parameter

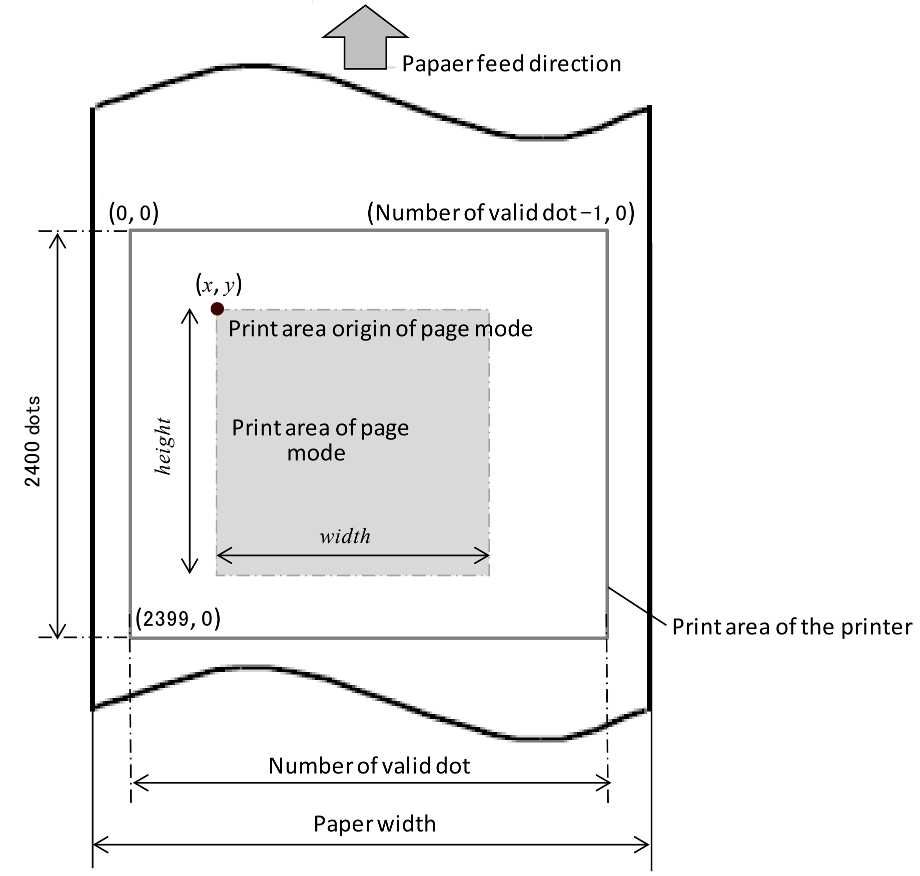
x The horizontal origin (dot) of the print area of page mode
0 represents the left edge on the print area of the printer.

y The vertical origin (dot) of the print area of page mode
The valid range is 0 to 2399.
0 represents the position where paper feed has not been performed.

width The print area width (dot) of page mode

height The print area height (dot) of page mode
The valid range is 1 to (2400-*y*).

The valid range of *x* and *width* are shown in figure below.



Memory Switch Setting of Printer		Number of Valid Dots	**setPageModeArea**	
MS4-4 (Paper Width)	**MS4-5 (Number of Effective Dots)**		*x*	*width*
80 mm	576	576	0 to 575	1 to (576 - *x*)
	512	512	0 to 511	1 to (512 - *x*)
58 mm	432	432	0 to 431	1 to (432 - *x*)
	360	360	0 to 359	1 to (360 - *x*)

The number of valid dots varies depending on the memory switch setting of the printer. See "User's Guide" for details of the memory switch and the factory default setting.

Exception **PrinterException**

PrinterException is thrown when an error occurs while this method is being called. See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

The print area of page mode can be specified when page mode is started by **enterPageMode** and then this method is executed after executing the dedicated method for page mode. The data that has been mapped is kept.
The data of the dedicated method for page mode is mapped to the print area of page mode added by this method after executing this method.

The print area of page mode is *x* = 0, *y* = 0, *width* = number of a valid dot, *height* = 2400 after executing **enterPageMode**.

setPageModeDirection Specify print direction of page mode

Specifies print direction of page mode.

Syntax public void **setPageModeDirection**(Direction *direction*) throws **PrinterException**

Parameter *direction* Print direction
See "4.3.1(3)㉖ Print direction (Direction)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

The print direction is left to right after executing **enterPageMode**.

setPageModeLineSpacing Specify line spacing of page mode

Specifies line spacing of page mode.

Syntax public void **setPageModeLineSpacing**(int *lineSpacing*) throws **PrinterException**

Parameter *linespacing* Line spacing (dot) of page mode
The valid range is 0 to 255.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

The line spacing is 34 dots after executing **enterPageMode**.

printPageMode Print page mode

Prints the print data kept in page data buffer.

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

Parameter *cuttingMethod* Cutting method
See "4.3.1(3)㉓ Cutting method (CuttingMethod)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 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 The print data is kept after printing. The print data is discarded at the timing of the following:
- Execute **enterPageMode**
- Execute **disconnect**
- Execute **exitPageMode**

printPageModeText Send text data of page mode

Maps the text data on the print area of page mode.

Syntax public void **printPageModeText**(int *startX,* int *startY,* String *text*) throws **PrinterException**

Parameter *startX* The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text Text data
Data size that can be specified at 1 time is 16 KB (16384 bytes).

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description This method encodes the specified text data to printable text data based on **setInternationalCharacter** and **setCodePage**.

Start page mode by **enterPageMode** before executing this method.

printPageModeTextEx Send format specified text data of page mode

Maps the format specified text data on the print area of page mode.

Syntax public void **printPageModeTextEx**(int *startX,*
int *startY,*
String *text,*
CharacterBold *bold,*
CharacterUnderline *underline,*
CharacterReverse *reverse,*
CharacterFont *font,*
CharacterScale *scale*) throws **PrinterException**

Parameter *startX* The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text Text data
Data size that can be specified at 1 time is 16 KB (16384 bytes).

bold Bold print
See "4.3.1(3)⑥ Bold print (CharacterBold)" for available constants.

underline Underline
See "4.3.1(3)⑦ Underline (CharacterUnderline)" for available constants.

reverse Reverse print
See "4.3.1(3)⑧ Reverse print (CharacterReverse)" for available constants.

font Font
See "4.3.1(3)⑩ Character font (CharacterFont)" for available constants.

scale Character scale
See "4.3.1(3)⑪ Character scale (CharacterScale)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description This method encodes the specified text data to printable text data based on **setInternationalCharacter** and **setCodePage**.

Start page mode by **enterPageMode** before executing this method.

printPageModeBarcode Print barcode of page mode

Maps the barcode on the print area of page mode.
The method of syntax (a) specifies the barcode data by character string.
The method of syntax (b) specifies the barcode data by character string and specifies N:W ratio of the barcode.
The method of syntax (c) specifies the barcode data by the array of bytes.
The method of syntax (d) is not supported.

Syntax

```
(a) public void printPageModeBarcode(int startX,
                                     int startY,
                                     BarcodeSymbol barcodeSymbol,
                                     String text,
                                     ModuleSize moduleSize,
                                     int moduleHeight,
                                     HriPosition hriPosition,
                                     CharacterFont hriFont) throws PrinterException

(b) public void printPageModeBarcode(int startX,
                                     int startY,
                                     BarcodeSymbol barcodeSymbol,
                                     String text,
                                     ModuleSize moduleSize,
                                     int moduleHeight,
                                     HriPosition hriPosition,
                                     CharacterFont hriFont,
                                     NwRatio nwRatio) throws PrinterException

(c) public void printPageModeBarcode(int startX,
                                     int startY,
                                     BarcodeSymbol barcodeSymbol,
                                     byte[] data,
                                     ModuleSize moduleSize,
                                     int moduleHeight,
                                     HriPosition hriPosition,
                                     CharacterFont hriFont) throws PrinterException

(d) public void printPageModeBarcode(int startX,
                                     int startY,
                                     BarcodeSymbol barcodeSymbol,
                                     String text,
                                     ModuleSize moduleSize) throws PrinterException
```

Parameter *startX* The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

barcodeSymbol BarcodeSymbol
See "4.3.1(3)⑭ Barcode symbol (BarcodeSymbol)" for available constants and correspondent syntax.

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' ' ', '$', '%', '+', '-', '.', '/'	Start code and stop code ('*') are automatically added.
CODE93	1 to 150 bytes	(0x00 to 0x2E)	Input data with 0x2F or more at the end.
CODE128	2 to 150 bytes	(0x00 to 0x66)	When inputting the start code (0x67 to 0x69) of the CODE128 code set. Input data with 0x67 or more at the end.
		(0x00 to 0x7F)	When starting with a CODE128 special code start code ("{A", "{B", "{C").
ITF	2 to 150 characters (However, an even number)	'0' to '9'	
CODABAR	1 to 150 characters	'0' to '9' '$', '+', '-', '.', '/', ':'	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.

Barcode	Number of Data	Inputtable Data Character String (Data)	Remarks
GS1 Databar Expanded	2 to 255 characters	' ' to '"' '%' to '?' 'A' to 'Z' '_' 'a' to 'z' '{'	

moduleSize Barcode width
See "4.3.1(3)⑮ 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

barcodeSymbol		
	moduleSize	**Valid Range**
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 "4.3.1(3)⑯ HRI character print position (HriPosition)" for available constants.

hriFont
HRI character font
See "4.3.1(3)⑩ Character font (CharacterFont)" for available constants.

nwRatio
N:W ratio
See "4.3.1(3)⑰ N:W ratio (NwRatio)" for available constants.
Depending on specified *nwRatio* and *moduleSize*, the wide element width is set as shown 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 this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.
The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printPageModePDF417 Print PDF417 of page mode

Maps PDF417 on the print area of page mode.
The method of syntax (a) specifies PDF417 symbol.
The method of syntax (b) is fixed to standard PDF417.

Syntax (a) public void **printPageModePDF417**(int *startX,*
int *startY,*
String *text,*
ErrorCorrection *ErrorCorrection,*
int *row*,
int *column*,
ModuleSize *moduleSize*,
int *moduleHeight*,
Pdf417Symbol *pdf417Symbol*) throws **PrinterException**

(b) public void **printPageModePDF417**(int *startX,*
int *startY,*
String *text,*
ErrorCorrection *ErrorCorrection,*
int *row*,
int *column*,
ModuleSize *moduleSize*,
int *moduleHeight*) throws **PrinterException**

Parameter *startX* The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text Barcode data

errorCorrection Error correction level
See "4.3.1(3)⑱ Error correction level (ErrorCorrection)" for available constants.

row The number of rows (row)
The valid range is 0, 3 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 "4.3.1(3)⑮ 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.

pdf417Symbol Symbol of PDF417
See "4.3.1(3)⑲ PDF417 symbol (Pdf417Symbol)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.
The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printPageModeQRcode Print QR Code of page mode

Maps QR Code on the print area of page mode.
The method of syntax (a) is fixed to QR Code Model 2.
The method of syntax (b) specifies QR Code Model.

Syntax (a) public void **printPageModeQRcode** (int *startX,*
int *startY,*
String *text,*
ErrorCorrection *ErrorCorrection,*
ModuleSize *moduleSize*) throws **PrinterException**

(b) public void **printPageModeQRcode** (int *startX,*
int *startY,*
String *text,*
ErrorCorrection *ErrorCorrection,*
ModuleSize *moduleSize,*
QrModel *model*) throws **PrinterException**

Parameter *startX* The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text Barcode data
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 "4.3.1(3)⑱ Error correction level (ErrorCorrection)" for available constants.

moduleSize Module size
See "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

model QR Code Model
See "4.3.1(3)⑳ QR Code Model (QrModel)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.
The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printPageModeDataMatrix Print Data Matrix of page mode

Maps Data Matrix on the print area of page mode.

Syntax

```
public void printPageModeDataMatrix(int startX,
                                    int startY,
                                    String text,
                                    DataMatrixModule dataMatrixModule,
                                    ModuleSize moduleSize) throws PrinterException
```

Parameter

startX The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text Barcode data

dataMatrixModule The number of Data Matrix modules
See "4.3.1(3)㉑ Data Matrix module (DataMatrixModule)" for available constants.

moduleSize Module size
See "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.
The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printPageModeMaxiCode Print MaxiCode of page mode

Maps MaxiCode on the print area of page mode.

Syntax

```
public void printPageModeMaxiCode(int startX,
                                  int startY,
                                  String text,
                                  MaxiCodeMode maxiCodeMode) throws PrinterException
```

Parameter

startX The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text Barcode data

- 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 "4.3.1(3)㉒ MaxiCode Mode (MaxiCodeMode)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.
The quiet zone is not secured. Set the quiet zone in accordance with the standard of the barcode symbol.

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

printPageModeGS1DataBarStacked Print GS1 Databar Stacked of page mode

Maps GS1 Databar Stacked on the print area of page mode.

Syntax

```
public void printPageModeGS1DataBarStacked(int startX,
                                           int startY,
                                           String text,
                                           ModuleSize moduleSize) throws PrinterException
```

Parameter	*startX*	The horizontal reference point (dot) from the starting point The valid range is 0 to 2399.
	startY	The vertical reference point (dot) from the starting point The valid range is 0 to 2399.
	text	Barcode data 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 "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.

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

printPageModeGS1DataBarStackedOmnidirectional Print GS1 Databar Stacked Omni-directional of page mode

Maps GS1 Databar Stacked Omni-directional on the print area of page mode.

Syntax

```
public void printPageModeGS1DataBarStackedOmnidirectional(int startX,
                                                          int startY,
                                                          String text,
                                                          int moduleHeight,
                                                          ModuleSize moduleSize) throws PrinterException
```

Parameter *startX*

The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY

The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text

Barcode data
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 "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.

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

printPageModeGS1DataBarExpandedStacked
Print GS1 Databar Expanded Stacked of page mode

Maps GS1 Databar Expanded Stacked on the print area of page mode.

Syntax public void **printPageModeGS1DataBarExpandedStacked**(int *startX,*
int *startY,*
String *text,*
int *column*,
ModuleSize *moduleSize*) throws **PrinterException**

Parameter *startX*

The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY

The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

text

Barcode data
Enter any number of characters using the following:
' ', '!', '"', '%', '&', ''', '(', ')', '*', '+', ',', '-', '.', '/', ':', ';', '<', '=', '>', '?', '_', '0' to '9', 'A' to 'Z', 'a' to 'z'.
Enter '{1’ to FNC1.

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 "4.3.1(3)⑮ Module size (ModuleSize)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

Note Map the print data of the barcode not to overlap the other print data.

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

printPageModeAztecCode Print Aztec Code of page mode

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

Syntax
```
public void printPageModeAztecCode(int startX,
                                   int startY,
                                   String text,
                                   int layer,
                                   int errorCorrection,
                                   ModuleSize moduleSize,
                                   AztecSymbol aztecSymbol) throws PrinterException
```

sendPageModeBinary Send binary data of page mode

Maps binary data on the print area of page mode.

Syntax
```
public void sendPageModeBinary(byte [] binary) throws PrinterException
```

Parameter	*binary*	Binary data Data size that can be specified at 1 time is 16 KB (16384 bytes).

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

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.

Note This method may execute unexpected performance depending on the data to send. Please ensure the performance with your actual device in advance.

printPageModeImageFile Draw Image file of page mode

Maps the image file on the print area of page mode.

Syntax
```
public void printPageModeImageFile(int startX,
                                   int startY,
                                   String fileName,
                                   Dithering dithering) throws PrinterException
```

Parameter	*startX*	The horizontal reference point (dot) from the starting point The valid range is 0 to 2399.
	startY	The vertical reference point (dot) from the starting point The valid range is 0 to 2399.
	fileName	File path of the data 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 1 MB (1048576 bytes).
The image files that can be sent are .bmp, .jpg, .jpeg, .png.
Colored image file is converted to monochrome image by binarization and registered.

dithering Dithering
See "4.3.1(3)④ Dithering (Dithering)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

Description Start page mode by **enterPageMode** before executing this method.

printPageModeRectangle Draw rectangle image of page mode

Maps the rectangle image on the print area of page mode.

Syntax

```
public void printPageModeRectangle(int startX,
                                   int startY,
                                   int endX,
                                   int endY,
                                   LineStyle LineStyle) throws PrinterException
```

Parameter

startX The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

endX The horizontal reference point (dot) from the ending point
The valid range is 0 to 2399.

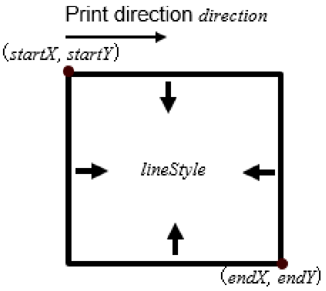
endY The vertical reference point (dot) from the ending point
The valid range is 0 to 2399.

lineStyle Line style
See "4.3.1(3)㉗ Line style (LineStyle)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

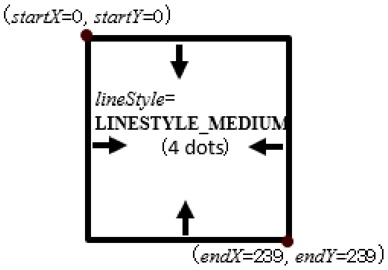
Description Start page mode by **enterPageMode** before executing this method.

The rectangle is mapped to *direction* of **setPageModeDirection** as shown in the figure below.



The example of the parameter setting to the image is shown below.

Example: Draw a square with a medium solid line (4 dots) at 240 dots (30 mm) from the starting point.

Image	Parameter
	*startX* 0 *startY* 0 *endX* 239 *endY* 239 *lineStyle* **LINESTYLE_MEDIUM**

printPageModeLine Print ruled line of page mode

Maps the ruled line on the print area of page mode.

Syntax

```
public void printPageModeLine(int startX,
                              int startY,
                              int endX,
                              int endY,
                              LineStyle LineStyle) throws PrinterException
```

Parameter

startX The horizontal reference point (dot) from the starting point
The valid range is 0 to 2399.

startY The vertical reference point (dot) from the starting point
The valid range is 0 to 2399.

endX The horizontal reference point (dot) from the ending point
The valid range is 0 to 2399.

endY The vertical reference point (dot) from the ending point
The valid range is 0 to 2399.

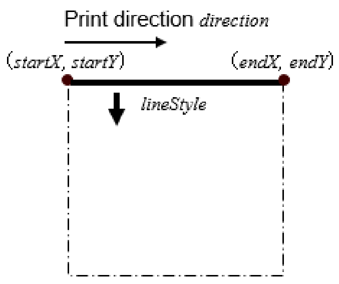
lineStyle Line style
See "4.3.1(3)㉗ Line style (LineStyle)" for available constants.

Exception **PrinterException**
PrinterException is thrown when an error occurs while this method is being called.
See "**4.3.5 PrinterException Class**" for details of the error.

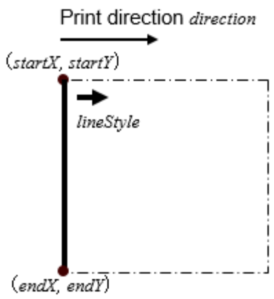
Description Start page mode by **enterPageMode** before executing this method.

A diagonal stroke cannot be drawn by this method.

The ruled line is mapped to the *direction* of **setPageModeDirection** as shown in the figure below.



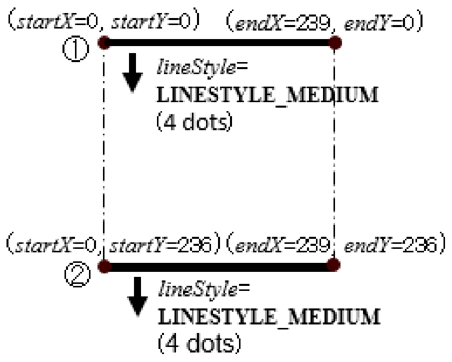
Mapping direction of horizontal ruled line



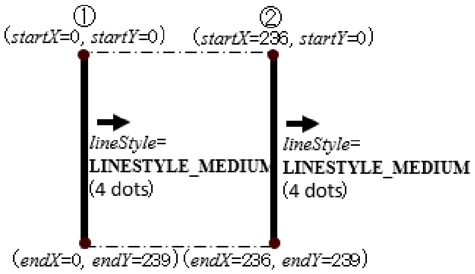
Mapping direction of vertical ruled line

The setting example of the parameter to the image is shown below.

Example: Draw a horizontal ruled line of a square with a medium solid line (4 dots) at 240 dots (30 mm) from the starting point.

Image	Parameter
 (*startX*=0, *startY*=0) (*endX*=239, *endY*=0) ① *lineStyle*= LINESTYLE_MEDIUM (4 dots) (*startX*=0, *startY*=236) (*endX*=239, *endY*=236) ② *lineStyle*= LINESTYLE_MEDIUM (4 dots)	① *startX* 0 *startY* 0 *endX* 239 *endY* 0 *lineStyle* **LINESTYLE_MEDIUM** ② *startX* 0 *startY* 236 *endX* 239 *endY* 236 *lineStyle* **LINESTYLE_MEDIUM**

Example: Draw a vertical ruled line of a square with a medium solid line (4 dots) at 240 dots (30 mm) from the starting point.

Image	Parameter
 ① (*startX*=0, *startY*=0) ② (*startX*=236, *startY*=0) *lineStyle*= LINESTYLE_MEDIUM (4 dots) *lineStyle*= LINESTYLE_MEDIUM (4 dots) (*endX*=0, *endY*=239) (*endX*=236, *endY*=239)	① *startX* 0 *startY* 0 *endX* 0 *endY* 239 *lineStyle* **LINESTYLE_MEDIUM** ② *startX* 236 *startY* 0 *endX* 236 *endY* 239 *lineStyle* **LINESTYLE_MEDIUM**

printPageModeLogo Print logo of page mode

Maps the registered logo on the print area of page mode.

Syntax	public void **printPageModeLogo**(int *startX,* int *startY,* String *id*) throws **PrinterException**	
Parameter	*startX*	The horizontal reference point (dot) from the starting point The valid range is 0 to 2399.
	startY	The vertical reference point (dot) from the starting point The valid range is 0 to 2399.
	id	Logo ID to print (key code) Specify the ID of the registered logo as a character string
Exception	**PrinterException** **PrinterException** is thrown when an error occurs while this method is being called. See "**4.3.5 PrinterException Class**" for details of the error.	
Description	Start page mode by **enterPageMode** before executing this method.	

4.3.2 **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 "4.3.2(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.

4.3.3 **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 printer search

(2) Method Details

finishEvent End event of 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.

4.3.4 **PrinterInfo** Class

PrinterInfo class stores the information of the printer found by **startDiscoveryPrinter**.

(1) Method List

Printer model name (Printer name), Bluetooth address, MAC address, IP 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
getMacAddress	Get MAC address
getIpAddress	Get IP 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 (Printer name) from the printer information found by **startDiscoveryPrinter**.

Syntax public String **getPrinterModelName**()

Return value Printer model name (Printer 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

getMacAddress — Get MAC address

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

Syntax public String **getMacAddress**()

Return value MAC address

getIPAddress Get IP address

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

Syntax	public String **getIpAddress**()
Return value	IP 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

4.3.5 **PrinterException** Class

(1) Method List

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

Name	Description
PrinterException	Constructor
getErrorCode	Get error code

(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	Port is not open.	-12
ERROR_DEVICE_NOT_CONNECTED	There is a problem with connection between the Android device and printer.	-21
ERROR_OFFLINE	Disconnected state or the printer is offline.	-22
ERROR_DEVICE_INITIALIZE_FAILED	Failed to change the printer setting. 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 being used by another process.	-302
ERROR_FILE_INVALID	The specified file is invalid.	-303
ERROR_LOW_MEMORY	Memory shortage occurred when loading image file.	-311
ERROR_OVER_ MAX_IMAGE	Either or both of width and height of image file 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_PAGE_MODE_SPECIFIED	Page mode is specified.	-511
ERROR_PAGE_MODE_NOT_SPECIFIE	Page mode is not specified.	-512
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 "4.3.5(2) Constant List" for details of the error.

4.3.6 **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.

4.3.7 **BarcodeScannerListener** Interface

BarcodeScannerListener Interface is an interface for the barcode scanner connection, barcode scanner disconnection, or received barcode data obtaining.
This interface is supported only by RP-F10.

(1) Method List

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

Name	Description
onBarcodeScannerReadData	Receipt event of barcode data
onBarcodeScannerChangedOnline	Connection event of barcode scanner
onBarcodeScannerChangedOffline	Disconnection event of barcode scanner

(2) Method Details

onBarcodeScannerReadData Receipt event of barcode data

Syntax public void **onBarcodeScannerReadData**(byte[] *data*)

Parameter *data* Barcode data received from the barcode scanner

Description This method is called when barcode data is received after starting a callback of the barcode data receipt from the barcode scanner by **setBarcodeScannerListener**.

The receipt of barcode data is notified when **isConnect** is true and the printer is connecting.

This method is an interface, so it is not implemented.
Implement the optional process in the class that receives a callback of the barcode scanner event.

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

onBarcodeScannerChangedOnline Connection event of barcode scanner

Syntax public void **onBarcodeScannerChangedOnline**()

Description This method is called at the following timing.
・When the barcode scanner is in connected status with the printer and **setBarcodeScannerListener** is executed.
・When the barcode scanner is connected to the printer.

The connection event of barcode data is notified when **isConnect** is true and the printer is connecting.

This method is an interface, so it is not implemented.
Implement the optional process in the class that receives callback of the barcode scanner event.

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

onBarcodeScannerChangedOffline Disconnection event of barcode scanner

Syntax public void **onBarcodeScannerChangedOffline**()

Description This method is called at the following timing.

・When the barcode scanner is in disconnected status with the printer and **setBarcodeScannerListener** is executed.
・When the barcode scanner is disconnected to the printer.

The disconnection event of barcode scanner 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 callback of the barcode scanner event.

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

4.3.8 **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.

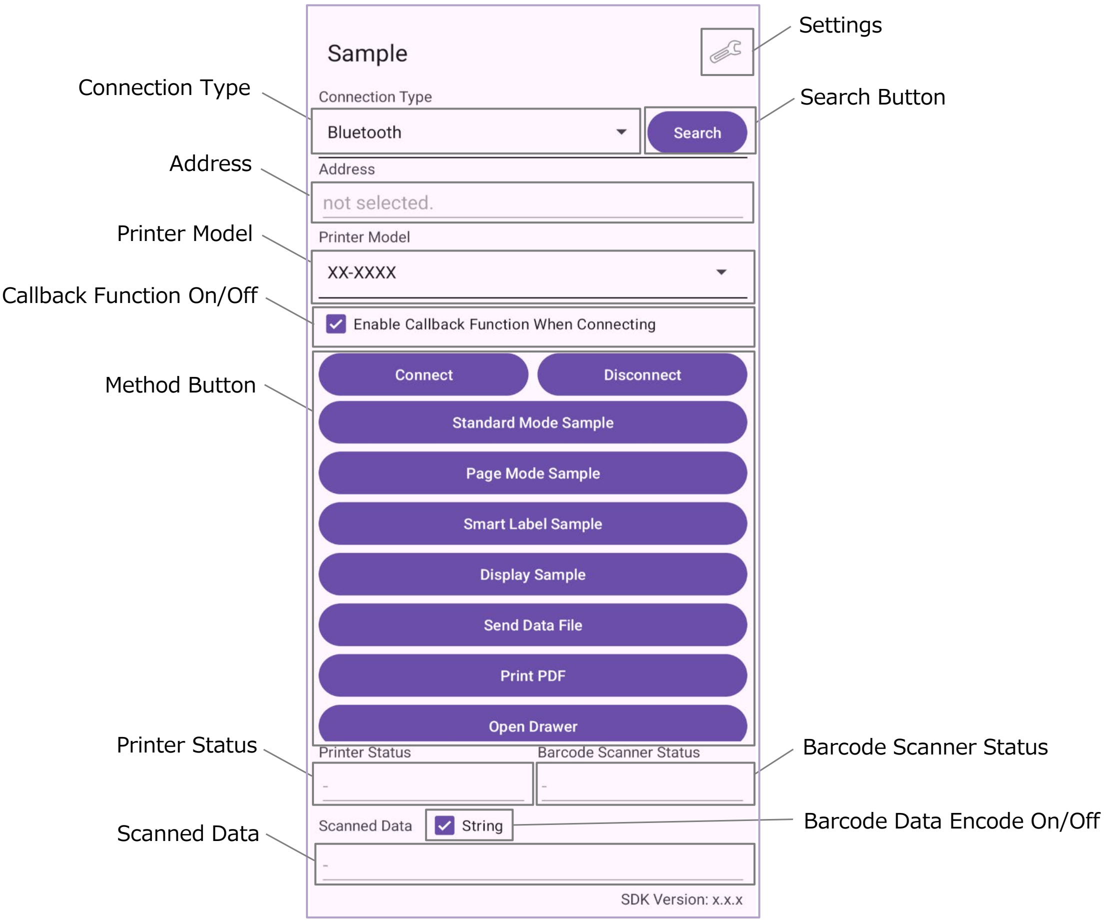
Chapter 5
Sample Program

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

5.1 Screen Layout

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



Sample

Connection Type: Bluetooth

Address: not selected.

Printer Model: XX-XXXX

☑ Enable Callback Function When Connecting

Printer Status: -

Barcode Scanner Status: -

Scanned Data ☑ String

-

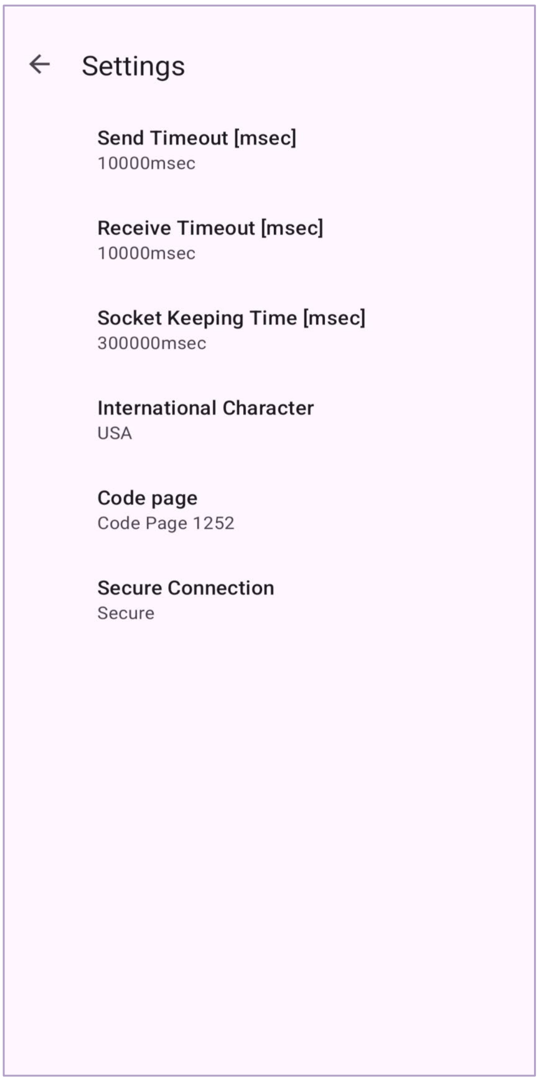
SDK Version: x.x.x

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. When [Callback Function On/Off] is On, the connection status of the barcode scanner is displayed.
Barcode Data Encode On/Off	Selects the conversion of barcode data read by the barcode scanner. On: Converts the read binary data to the character strings and displays. Off: Displays the read binary data as it is.
Scanned Data	Displays the barcode data read by the barcode scanner. When [Callback Function On/Off] is On, the barcode data read by the barcode scanner is displayed.

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

5.1.2 [Settings] Screen

Various setting functions are displayed in [Settings].



Settings

Send Timeout [msec]
10000msec

Receive Timeout [msec]
10000msec

Socket Keeping Time [msec]
300000msec

International Character
USA

Code page
Code Page 1252

Secure Connection
Secure

5.2 Precaution

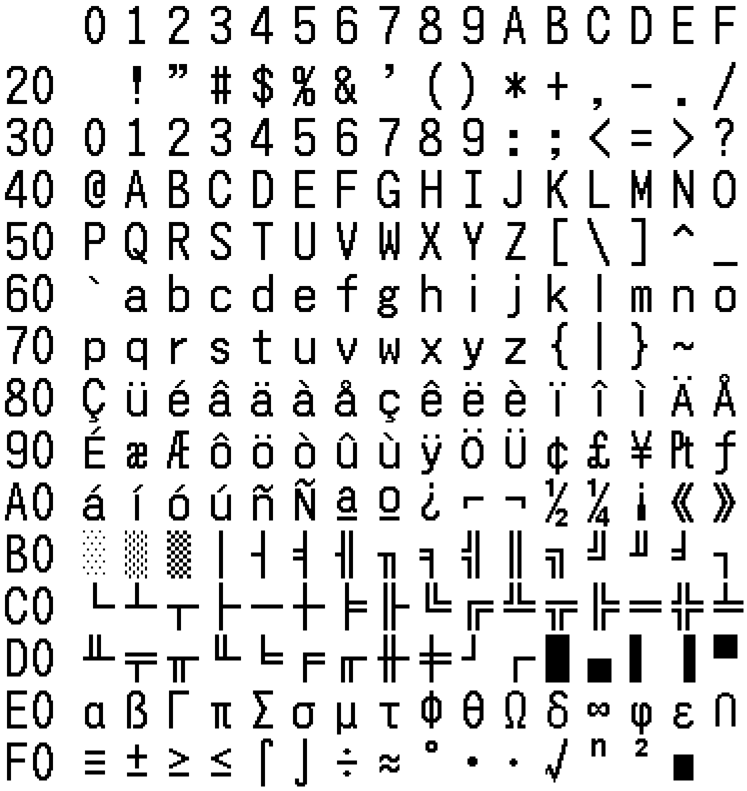
The sample program is subject to change without notice.

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

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.



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	₧	ƒ
A0	á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	»
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	φ	ε	∩
F0	≡	±	≥	≤	⌠	⌡	÷	≈	°	•	·	√	ⁿ	²	■	

Figure A-1 **CODE_PAGE_437 (USA, Standard Europe)**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	‾	
80																
90																
A0		｡	｢	｣	､	･	ｦ	ｧ	ｨ	ｩ	ｪ	ｫ	ｬ	ｭ	ｮ	ｯ
B0	ｰ	ｱ	ｲ	ｳ	ｴ	ｵ	ｶ	ｷ	ｸ	ｹ	ｺ	ｻ	ｼ	ｽ	ｾ	ｿ
C0	ﾀ	ﾁ	ﾂ	ﾃ	ﾄ	ﾅ	ﾆ	ﾇ	ﾈ	ﾉ	ﾊ	ﾋ	ﾌ	ﾍ	ﾎ	ﾏ
D0	ﾐ	ﾑ	ﾒ	ﾓ	ﾔ	ﾕ	ﾖ	ﾗ	ﾘ	ﾙ	ﾚ	ﾛ	ﾜ	ﾝ	ﾞ	ﾟ
E0																
F0																

Figure A-2 CODE_PAGE_KATAKANA

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ø	£	Ø	×	ƒ
A0	á	í	ó	ú	ñ	Ñ	ª	º	¿	®	¬	½	¼	¡	«	»
B0	░	▒	▓	│	┤	Á	Â	À	©	╣	║	╗	╝	¢	¥	┐
C0	└	┴	┬	├	─	┼	ã	Ã	╚	╔	╩	╦	╠	═	╬	¤
D0	ð	Ð	Ê	Ë	È	ı	Í	Î	Ï	┘	┌	█	▄	¦	Ì	▀
E0	ó	ß	Ô	ò	õ	Õ	µ	þ	Þ	Ú	Û	Ù	ý	Ý	¯	´
F0	-	±	‗	¾	¶	§	÷	¸	°	¨	·	¹	³	²	■	

Figure A-3 CODE_PAGE_850 (Multilingual)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ã	à	Á	ç	ê	Ê	è	Í	Ô	ì	Ã	Â
90	É	À	È	ô	õ	ò	Ú	ù	Ì	Õ	Ü	¢	£	Ù	₧	Ó
A0	á	í	ó	ú	ñ	Ñ	ª	º	¿	Ò	¬	½	¼	¡	«	»
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	φ	ε	∩
F0	≡	±	≥	≤	⌠	⌡	÷	≈	°	•	·	√	ⁿ	²	■	

Figure A-4 CODE_PAGE_860 (Portuguese)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	Â	à	¶	ç	ê	ë	è	ï	î	‗	À	§
90	É	È	Ê	ô	Ë	Ï	û	ù	¤	Ô	Ü	¢	£	Ù	Û	ƒ
A0	¦	´	ó	ú	¨	¸	³	¯	Î	⌐	¬	½	¼	¾	«	»
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	φ	ε	∩
F0	≡	±	≥	≤	⌠	⌡	÷	≈	°	•	·	√	ⁿ	²	■	

Figure A-5 CODE_PAGE_863 (Canadian-French)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ø	£	Ø	₧	ƒ
A0	á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	¤
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	φ	ε	∩
F0	≡	±	≥	≤	⌠	⌡	÷	≈	°	∙	·	√	ⁿ	²	■	

Figure A-6 CODE_PAGE_865 (Nordic)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ı	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	İ	Ö	Ü	ø	£	Ø	Ş	ş
A0	á	í	ó	ú	ñ	Ñ	Ğ	ğ	¿	®	¬	½	¼	¡	«	»
B0	░	▒	▓	│	┤	Á	Â	À	©	╣	║	╗	╝	¢	¥	┐
C0	└	┴	┬	├	─	┼	ã	Ã	╚	╔	╩	╦	╠	═	╬	¤
D0	º	ª	Ê	Ë	È	€	Í	Î	Ï	┘	┌	█	▄	¦	Ì	▀
E0	Ó	ß	Ô	Ò	õ	Õ	µ		×	Ú	Û	Ù	ì	ÿ	¯	´
F0	-	±		¾	¶	§	÷	¸	°	¨	·	¹	³	²	■	

Figure A-7 CODE_PAGE_857 (Turkish)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο	Π
90	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω	α	β	γ	δ	ε	ζ	η	θ
A0	ι	κ	λ	μ	ν	ξ	ο	π	ρ	σ	ς	τ	υ	φ	χ	ψ
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	ω	ά	έ	ή	ϊ	ί	ό	ύ	ϋ	ώ	Ά	Έ	Ή	Ί	Ό	Ύ
F0	Ώ	±	≥	≤	Ϊ	Ϋ	÷	≈	°	•	·	√	^{n}	^{2}	■	

Figure A-8 CODE_PAGE_737 **(Greek)**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	€		‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ		Ž	
90		‘	’	“	”	•	–	—	˜	™	š	›	œ		ž	Ÿ
A0		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
B0	°	±	^{2}	^{3}	´	µ	¶	·	¸	^{1}	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Figure A-9 CODE_PAGE_1252 **(Latin)**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
90	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
A0	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
F0	Ё	ё	Є	є	Ї	ї	Ў	ў	°	•	·	√	№	¤	■	

Figure A-10 CODE_PAGE_866 (Russian)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ä	ů	ć	ç	ł	ë	Ő	ő	î	Ź	Ä	Ć
90	É	Ĺ	ĺ	ô	ö	Ľ	ľ	Ś	ś	Ö	Ü	Ť	ť	Ł	×	č
A0	á	í	ó	ú	Ą	ą	Ž	ž	Ę	ę	¬	ź	Č	ş	«	»
B0	░	▒	▓	│	┤	Á	Â	Ě	Ş	╣	║	╗	╝	Ż	ż	┐
C0	└	┴	┬	├	─	┼	Ă	ă	╚	╔	╩	╦	╠	═	╬	¤
D0	đ	Đ	Ď	Ë	ď	Ň	Í	Î	ě	┘	┌	█	▄	Ţ	Ů	▀
E0	Ó	ß	Ô	Ń	ń	ň	Š	š	Ŕ	Ú	ŕ	Ű	ý	Ý	ţ	´
F0	-	˝	˛	ˇ	˘	§	÷	¸	°	¨	˙	ű	Ř	ř	■	

Figure A-11 CODE_PAGE_852 (Eastern Europe)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ø	£	Ø	×	ƒ
A0	á	í	ó	ú	ñ	Ñ	ª	º	¿	®	¬	½	¼	¡	«	»
B0	░	▒	▓	│	┤	Á	Â	À	©	╣	║	╗	╝	¢	¥	┐
C0	└	┴	┬	├	─	┼	ã	Ã	╚	╔	╩	╦	╠	═	╬	¤
D0	ð	Ð	Ê	Ë	È	€	Í	Î	Ï	┘	┌	█	▄	¦	Ì	▀
E0	Ó	ß	Ô	Ò	õ	Õ	µ	þ	Þ	Ú	Û	Ù	ý	Ý	¯	´
F0	-	±	‗	¾	¶	§	÷	¸	°	¨	·	¹	³	²	■	

Figure A-12 CODE_PAGE_858 (Euro)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	ђ	Ђ	ѓ	Ѓ	ё	Ё	є	Є	ѕ	Ѕ	і	І	ї	Ї	ј	Ј
90	љ	Љ	њ	Њ	ћ	Ћ	ќ	Ќ	ў	Ў	џ	Џ	ю	Ю	ъ	Ъ
A0	а	А	б	Б	ц	Ц	д	Д	е	Е	ф	Ф	г	Г	«	»
B0	░	▒	▓	│	┤	х	Х	и	И	╣	║	╗	╝	й	Й	┐
C0	└	┴	┬	├	─	┼	к	К	╚	╔	╩	╦	╠	═	╬	¤
D0	л	Л	м	М	н	Н	о	О	п	┘	┌	█	▄	П	я	▀
E0	Я	р	Р	с	С	т	Т	у	У	ж	Ж	в	В	ь	Ь	№
F0	-	ы	Ы	з	З	ш	Ш	э	Э	щ	Щ	ч	Ч	§	■	

Figure A-13 CODE_PAGE_855 (Cyrillic)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	°	·	•	√	▒	─	│	┼	┤	┬	├	┴	┐	┌	└	┘
90	β	∞	φ	±	½	¼	≈	«	»	ﻷ	ﻸ			ﻻ	ﻼ	
A0		–	ﺂ	£	¤	ﺄ			ﺎ	ﺏ	ﺕ	ﺙ	،	ﺝ	ﺡ	ﺥ
B0	٠	١	٢	٣	٤	٥	٦	٧	٨	٩	ﻑ	؛	ﺱ	ﺵ	ﺹ	؟
C0	¢	ﺀ	ﺁ	ﺃ	ﺅ	ﻊ	ﺋ	ﺍ	ﺑ	ﺓ	ﺗ	ﺛ	ﺟ	ﺣ	ﺧ	ﺩ
D0	ﺫ	ﺭ	ﺯ	ﺳ	ﺷ	ﺻ	ﺿ	ﻁ	ﻅ	ﻋ	ﻏ	¦	¬	÷	×	ﻉ
E0	ـ	ﻓ	ﻗ	ﻛ	ﻟ	ﻣ	ﻧ	ﻫ	ﻭ	ﻯ	ﻳ	ﺽ	ﻌ	ﻎ	ﻍ	ﻡ
F0	ﹽ		ﻥ	ﻩ	ﻬ	ﻰ	ﻲ	ﻐ	ﻕ	ﻵ	ﻶ	ﻝ	ﻙ	ﻱ	■	

Figure A-14 CODE_PAGE_864 (Arabic)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	€		‚		„	…	†	‡		‰	Š	‹	Ś	Ť	Ž	Ź
90		‘	’	“	”	•	–	—		™	š	›	ś	ť	ž	ź
A0		ˇ	˘	Ł	¤	Ą	¦	§	¨	©	Ş	«	¬	–	®	Ż
B0	°	±	˛	ł	´	µ	¶	·	¸	ą	ş	»	Ľ	˝	ľ	ż
C0	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D0	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E0	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F0	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

Figure A-15 CODE_PAGE_1250 (Central European)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Ђ	Ѓ	‚	ѓ	„	…	†	‡	€	‰	Љ	‹	Њ	Ќ	Ћ	Џ
90	ђ	‘	’	“	”	•	–	—		™	љ	›	њ	ќ	ћ	џ
A0		Ў	ў	Ј	¤	Ґ	¦	§	Ё	©	Є	«	¬	-	®	Ї
B0	°	±	І	і	ґ	µ	¶	·	ё	№	є	»	ј	Ѕ	ѕ	ї
C0	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
D0	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
E0	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F0	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я

Figure A-16 CODE_PAGE_1251 **(Cyrillic)**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	€		‚	ƒ	„	…	†	‡		‰		‹				
90		‘	’	“	”	•	–	—		™		›				
A0		΅	Ά	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	―
B0	°	±	²	³	΄	µ	¶	·	Έ	Ή	Ί	»	Ό	½	Ύ	Ώ
C0	ΐ	Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο
D0	Π	Ρ		Σ	Τ	Υ	Φ	Χ	Ψ	Ω	Ϊ	Ϋ	ά	έ	ή	ί
E0	ΰ	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
F0	π	ρ	ς	σ	τ	υ	φ	χ	ψ	ω	ϊ	ϋ	ό	ύ	ώ	

Figure A-17 CODE_PAGE_1253 **(Greek)**

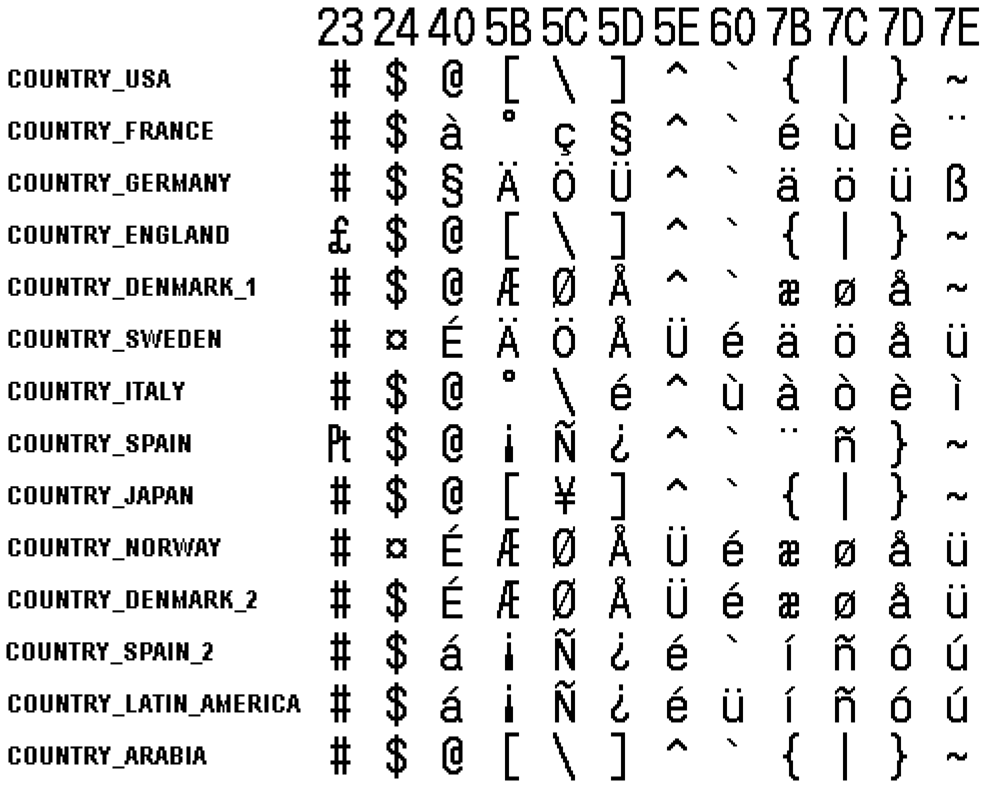
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	”	#	$	%	&	’	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	€		‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ			
90		‘	’	“	”	•	–	—	˜	™	š	›	œ			Ÿ
A0		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
B0	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ğ	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	İ	Ş	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ğ	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ı	ş	ÿ

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.



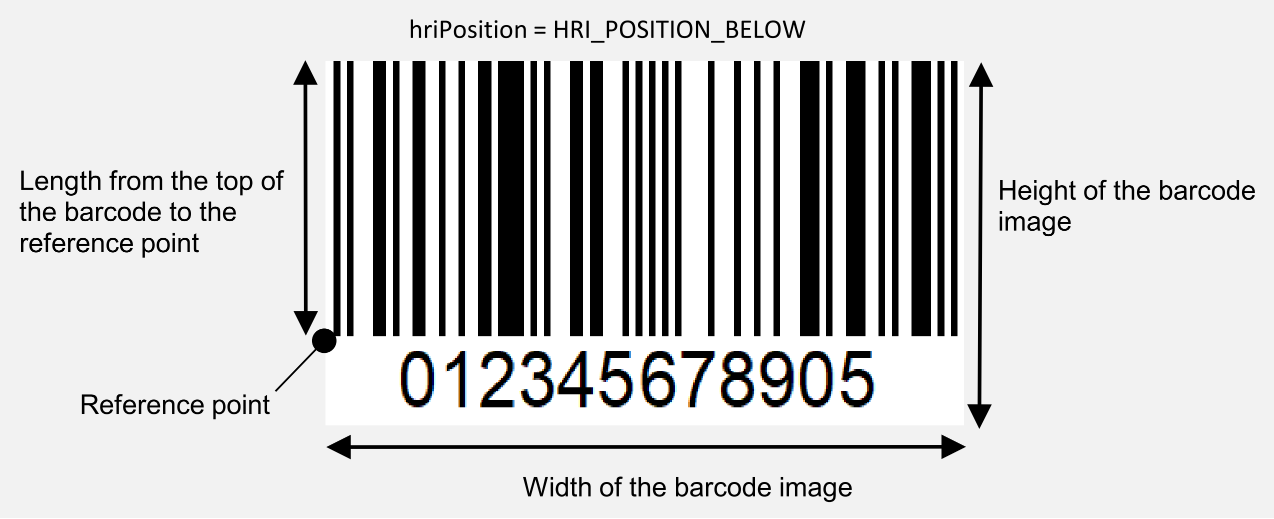
	23	24	40	5B	5C	5D	5E	60	7B	7C	7D	7E
COUNTRY_USA	#	$	@	[	\	]	^	`	{	|	}	~
COUNTRY_FRANCE	#	$	à	°	ç	§	^	`	é	ù	è	¨
COUNTRY_GERMANY	#	$	§	Ä	Ö	Ü	^	`	ä	ö	ü	ß
COUNTRY_ENGLAND	£	$	@	[	\	]	^	`	{	|	}	~
COUNTRY_DENMARK_1	#	$	@	Æ	Ø	Å	^	`	æ	ø	å	~
COUNTRY_SWEDEN	#	¤	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
COUNTRY_ITALY	#	$	@	°	\	é	^	ù	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， printPageModeBarcode


hriPosition = HRI_POSITION_BELOW
Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
012345678905
Width of the barcode image

(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**	190
	BARCODE_MODULE_WIDTH_3	285
	BARCODE_MODULE_WIDTH_4	380
	BARCODE_MODULE_WIDTH_5	475
	BARCODE_MODULE_WIDTH_6	570
BARCODE_UPC_E	**BARCODE_MODULE_WIDTH_2**	102
	BARCODE_MODULE_WIDTH_3	153
	BARCODE_MODULE_WIDTH_4	204
	BARCODE_MODULE_WIDTH_5	255
	BARCODE_MODULE_WIDTH_6	306
BARCODE_EAN13	**BARCODE_MODULE_WIDTH_2**	190
	BARCODE_MODULE_WIDTH_3	285
	BARCODE_MODULE_WIDTH_4	380
	BARCODE_MODULE_WIDTH_5	475
	BARCODE_MODULE_WIDTH_6	570
BARCODE_JAN13	**BARCODE_MODULE_WIDTH_2**	190
	BARCODE_MODULE_WIDTH_3	285
	BARCODE_MODULE_WIDTH_4	380
	BARCODE_MODULE_WIDTH_5	475
	BARCODE_MODULE_WIDTH_6	570
BARCODE_EAN8	**BARCODE_MODULE_WIDTH_2**	134
	BARCODE_MODULE_WIDTH_3	201
	BARCODE_MODULE_WIDTH_4	268
	BARCODE_MODULE_WIDTH_5	335
	BARCODE_MODULE_WIDTH_6	402
BARCODE_JAN8	**BARCODE_MODULE_WIDTH_2**	134
	BARCODE_MODULE_WIDTH_3	201
	BARCODE_MODULE_WIDTH_4	268
	BARCODE_MODULE_WIDTH_5	335
	BARCODE_MODULE_WIDTH_6	402
BARCODE_CODE93	**BARCODE_MODULE_WIDTH_2**	18 × number of barcode data + 56
	BARCODE_MODULE_WIDTH_3	27 × number of barcode data + 84
	BARCODE_MODULE_WIDTH_4	36 × number of barcode data + 112
	BARCODE_MODULE_WIDTH_5	45 × number of barcode data + 140
	BARCODE_MODULE_WIDTH_6	54 × number of barcode data + 168

barcodeSymbol	*moduleSize*	**Width of Barcode Image**
BARCODE_CODE128	**BARCODE_MODULE_WIDTH_2**	22 × number of barcode data + 26
	BARCODE_MODULE_WIDTH_3	33 × number of barcode data + 39
	BARCODE_MODULE_WIDTH_4	44 × number of barcode data + 52
	BARCODE_MODULE_WIDTH_5	55 × number of barcode data + 65
	BARCODE_MODULE_WIDTH_6	66 × number of barcode data + 78
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.

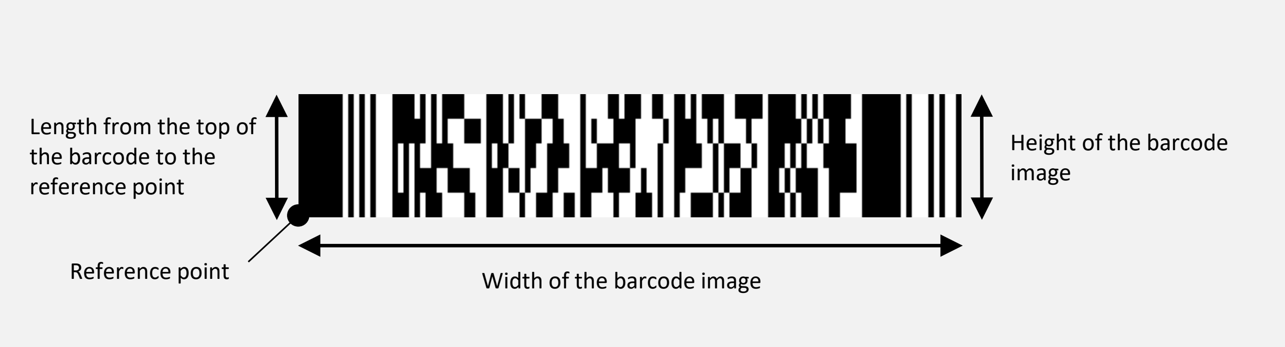
barcodeSymbol	*nwRatio*	*moduleSize*	**Width of Barcode Image**
BARCODE_CODE39	**NWRATIO_1TO2**	**BARCODE_MODULE_WIDTH_2**	26 × number of barcode data + 50
		BARCODE_MODULE_WIDTH_3	39 × number of barcode data + 75
		BARCODE_MODULE_WIDTH_4	52 × number of barcode data + 100
		BARCODE_MODULE_WIDTH_5	65 × number of barcode data + 125
		BARCODE_MODULE_WIDTH_6	78 × number of barcode data + 150
	NWRATIO_1TO2_5	**BARCODE_MODULE_WIDTH_2**	29 × number of barcode data + 56
		BARCODE_MODULE_WIDTH_3	45 × number of barcode data + 87
		BARCODE_MODULE_WIDTH_4	58 × number of barcode data + 112
		BARCODE_MODULE_WIDTH_5	74 × number of barcode data + 143
		BARCODE_MODULE_WIDTH_6	87 × number of barcode data + 168
	NWRATIO_1TO3	**BARCODE_MODULE_WIDTH_2**	32 × number of barcode data + 62
		BARCODE_MODULE_WIDTH_3	48 × number of barcode data + 93
		BARCODE_MODULE_WIDTH_4	64 × number of barcode data + 124
		BARCODE_MODULE_WIDTH_5	80 × number of barcode data + 155
		BARCODE_MODULE_WIDTH_6	96 × number of barcode data + 186
BARCODE_ITF	**NWRATIO_1TO2**	**BARCODE_MODULE_WIDTH_2**	14 × number of barcode data + 16
		BARCODE_MODULE_WIDTH_3	21 × number of barcode data + 24
		BARCODE_MODULE_WIDTH_4	28 × number of barcode data + 32
		BARCODE_MODULE_WIDTH_5	35 × number of barcode data + 40
		BARCODE_MODULE_WIDTH_6	42 × number of barcode data + 48
	NWRATIO_1TO2_5	**BARCODE_MODULE_WIDTH_2**	16 × number of barcode data + 17
		BARCODE_MODULE_WIDTH_3	25 × number of barcode data + 26
		BARCODE_MODULE_WIDTH_4	32 × number of barcode data + 34

barcodeSymbol	*nwRatio*	*moduleSize*	**Width of Barcode Image**
PM_BARCODE_ITF	**NWRATIO_1TO2_5**	**BARCODE_MODULE_WIDTH_5**	41 × number of barcode data + 43
		BARCODE_MODULE_WIDTH_6	48 × number of barcode data + 51
	NWRATIO_1TO3	**BARCODE_MODULE_WIDTH_2**	18 × number of barcode data + 18
		BARCODE_MODULE_WIDTH_3	27 × number of barcode data + 27
		BARCODE_MODULE_WIDTH_4	36 × number of barcode data + 36
		BARCODE_MODULE_WIDTH_5	45 × number of barcode data + 45
		BARCODE_MODULE_WIDTH_6	54 × number of barcode data + 54
BARCODE_CODABAR[*1]	**NWRATIO_1TO2**	**BARCODE_MODULE_WIDTH_2**	20 × number of data + 2 × (2 + number of wide data) - 2
		BARCODE_MODULE_WIDTH_3	30 × number of data + 3 × (2 + number of wide data) - 3
		BARCODE_MODULE_WIDTH_4	40 × number of data + 4 × (2 + number of wide data) - 4
		BARCODE_MODULE_WIDTH_5	50 × number of data + 5 × (2 + number of wide data) - 5
		BARCODE_MODULE_WIDTH_6	60 × number of data + 6 × (2 + number of wide data) - 6
	NWRATIO_1TO2_5	**BARCODE_MODULE_WIDTH_2**	22 × number of data + 3 × (2 + number of wide data) - 2
		BARCODE_MODULE_WIDTH_3	34 × number of data + 5 × (2 + number of wide data) - 3
		BARCODE_MODULE_WIDTH_4	44 × number of data + 6 × (2 + number of wide data) - 4
		BARCODE_MODULE_WIDTH_5	56 × number of data + 8 × (2 + number of wide data) - 5
		BARCODE_MODULE_WIDTH_6	66 × number of data + 9 × (2 + number of wide data) - 6
	NWRATIO_1TO3	**BARCODE_MODULE_WIDTH_2**	24 × number of data + 4 × (2 + number of wide data) - 2
		BARCODE_MODULE_WIDTH_3	36 × number of data + 6 × (2 + number of wide data) - 3
		BARCODE_MODULE_WIDTH_4	48 × number of data + 8 × (2 + number of wide data) - 4
		BARCODE_MODULE_WIDTH_5	60 × number of data + 10 × (2 + number of wide data) - 5
		BARCODE_MODULE_WIDTH_6	72 × number of data + 12 × (2 + number of wide data) - 6

[*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**	244
		BARCODE_MODULE_WIDTH_3	366
		BARCODE_MODULE_WIDTH_4	488
		BARCODE_MODULE_WIDTH_5	610
		BARCODE_MODULE_WIDTH_6	732
	17 or 18	**BARCODE_MODULE_WIDTH_2**	298
		BARCODE_MODULE_WIDTH_3	447
		BARCODE_MODULE_WIDTH_4	596
		BARCODE_MODULE_WIDTH_5	745
		BARCODE_MODULE_WIDTH_6	894
BARCODE_JAN13_ ADDON	14 or 15	**BARCODE_MODULE_WIDTH_2**	244
		BARCODE_MODULE_WIDTH_3	366
		BARCODE_MODULE_WIDTH_4	488
		BARCODE_MODULE_WIDTH_5	610
		BARCODE_MODULE_WIDTH_6	732
	17 or 18	**BARCODE_MODULE_WIDTH_2**	298
		BARCODE_MODULE_WIDTH_3	447
		BARCODE_MODULE_WIDTH_4	596
		BARCODE_MODULE_WIDTH_5	745
		BARCODE_MODULE_WIDTH_6	894

B.1.2 printPDF417， printPageModePDF417


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(1) Height of the barcode image

Height of the barcode image[*1] = *moduleHeight* × *row*[*2]

*1: Height of the barcode image = Length from the top of the barcode to the reference point
*2: *row*≠0

(2) Width of the barcode image

When *pdf417Symbol* is **PDF417_STANDARD**:

Width of the barcode image = (17 × *column*[*1] + 69) × module size value

*1: *column*≠0

When *pdf417Symbol* is **PDF417_COMPACT**:

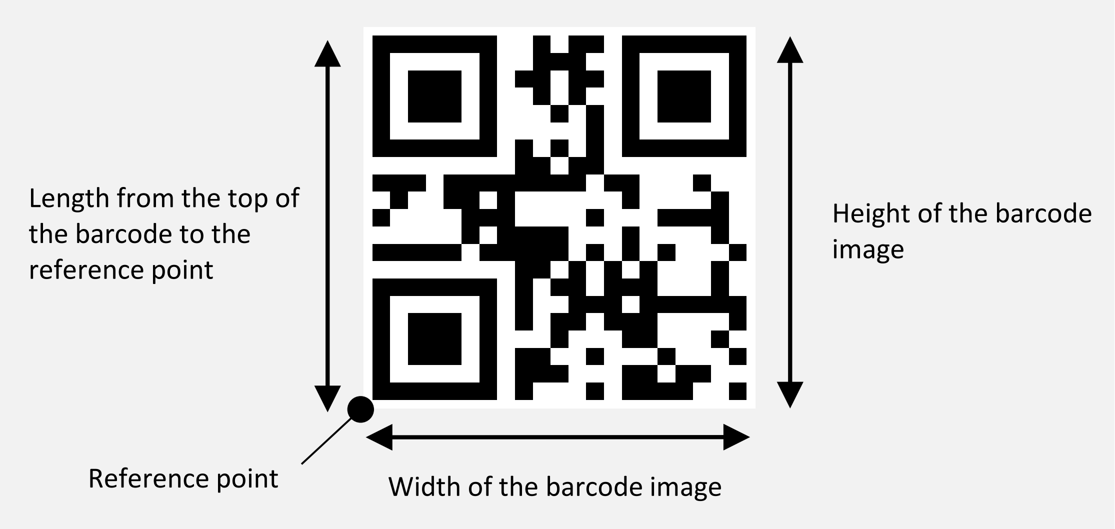
Width of the barcode image = (17 × *column*[*1] + 35) × module size value

*1: *column*≠0

Module Size Value

moduleSize	**Module Size Value**
PDF417_MODULE_WIDTH_2	2
PDF417_MODULE_WIDTH_3	3
PDF417_MODULE_WIDTH_4	4
PDF417_MODULE_WIDTH_5	5
PDF417_MODULE_WIDTH_6	6
PDF417_MODULE_WIDTH_7	7
PDF417_MODULE_WIDTH_8	8

B.1.3 printQRCode， printPageModeQRCode


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(1) Height and width of the barcode image

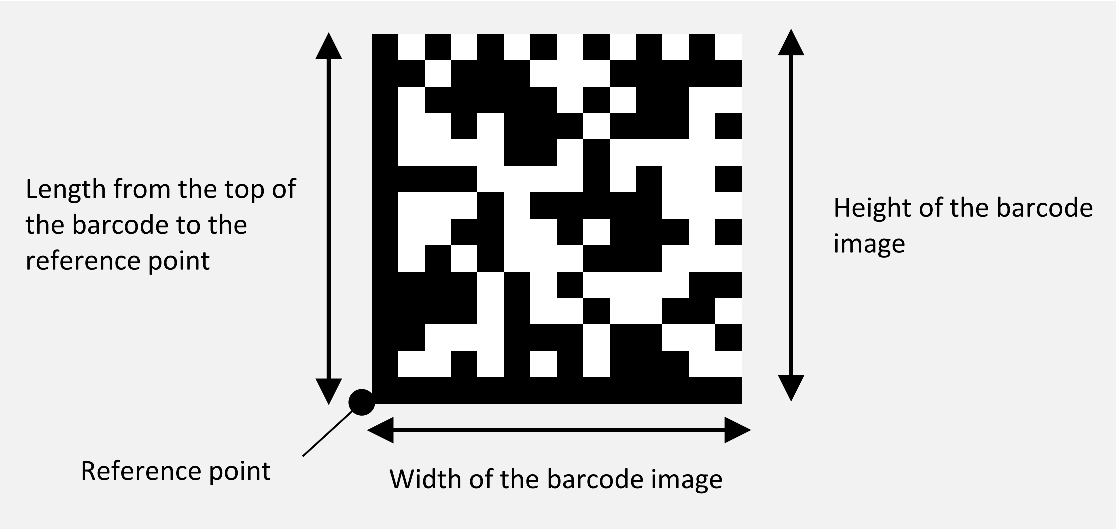
Height[*1] and width of the barcode image = (4 × version[*2] + 17) × module size value

[*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， printPageModeDataMatrix


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(1) Height and width of the barcode image

Height of the barcode image = number of vertical module × module size value

Width of the barcode image = number of horizontal module × module size value

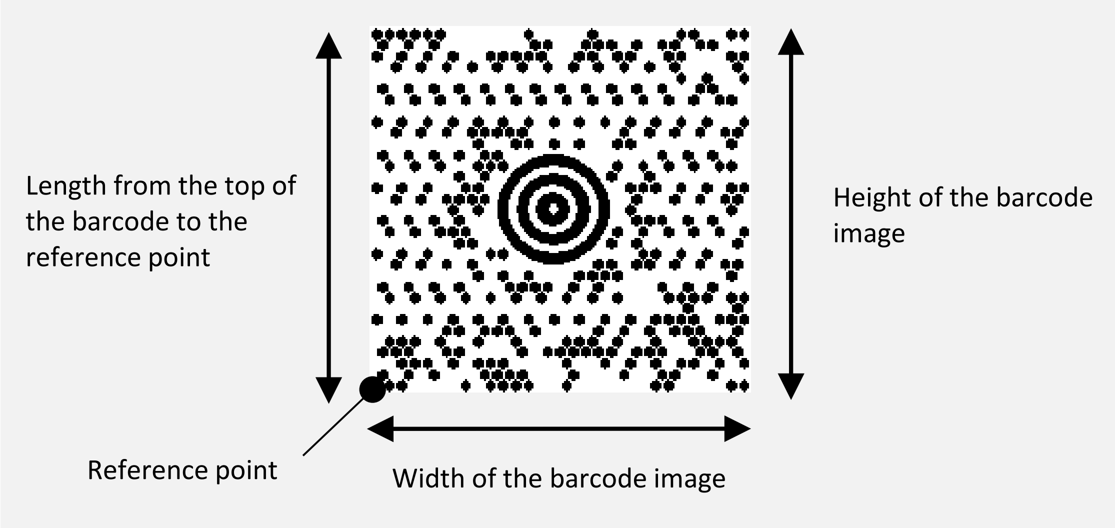
dataMatrixModule	**Number of Vertical Module**	**Number of Horizontal Module**
DATA_MATRIX_10_10	10	10
DATA_MATRIX_12_12	12	12
DATA_MATRIX_14_14	14	14
DATA_MATRIX_16_16	16	16
DATA_MATRIX_18_18	18	18
DATA_MATRIX_20_20	20	20
DATA_MATRIX_22_22	22	22
DATA_MATRIX_24_24	23	23
DATA_MATRIX_26_26	26	26
DATA_MATRIX_32_32	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， printPageModeMaxicode


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(1) Height of the barcode image

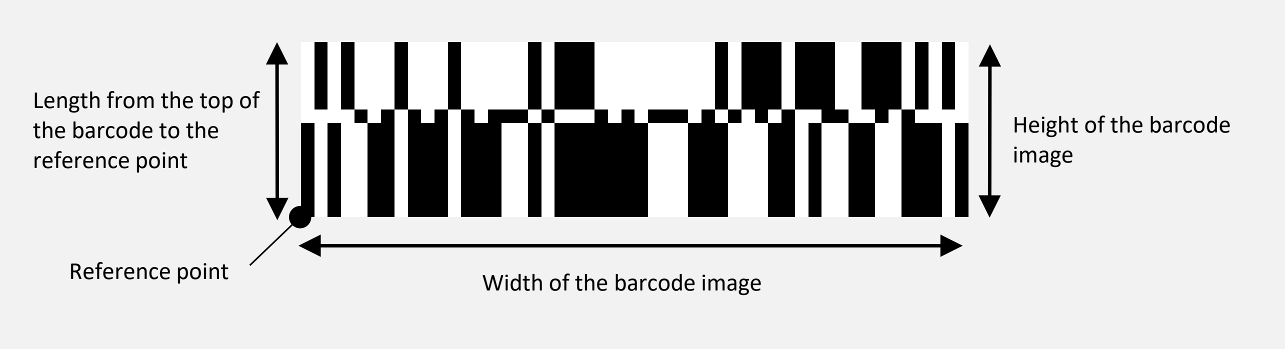
Height of the barcode image[*1] = 200

[*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 = 210

B.1.6 printGS1DataBarStacked， printPageModeGS1DataBarStacked


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(1) Height and width of the barcode image

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

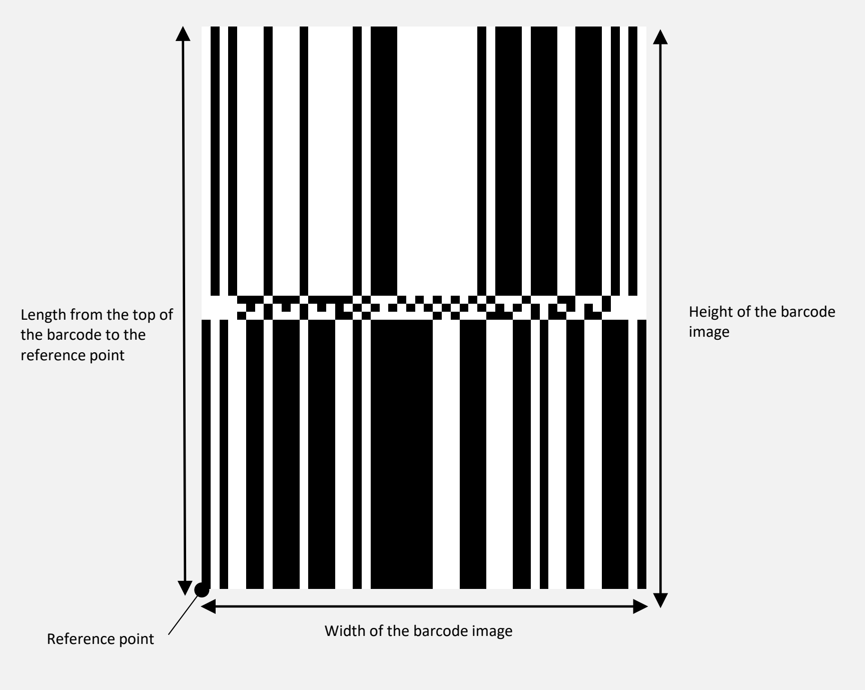
*1: Height of the barcode image = Length from the top of the barcode to the reference point

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 , printPageModeGS1DataBarStackedOmnidirectional


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(1) Height and width of the barcode image

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

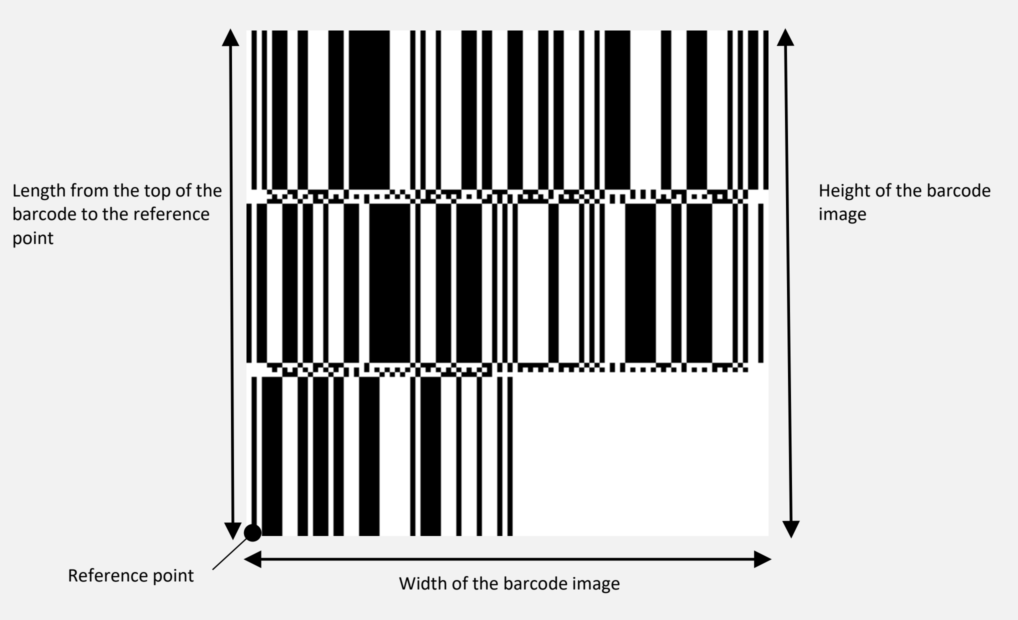
[*1]: Height of the barcode image = Length from the top of the barcode to the reference point

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 , printPageModeGS1DataBarExpandedStacked


Length from the top of the barcode to the reference point
Height of the barcode image
Reference point
Width of the barcode image

(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