Xamarin SDK

PV3/PV4

BARCODE PRINTER

Ver. 1.01

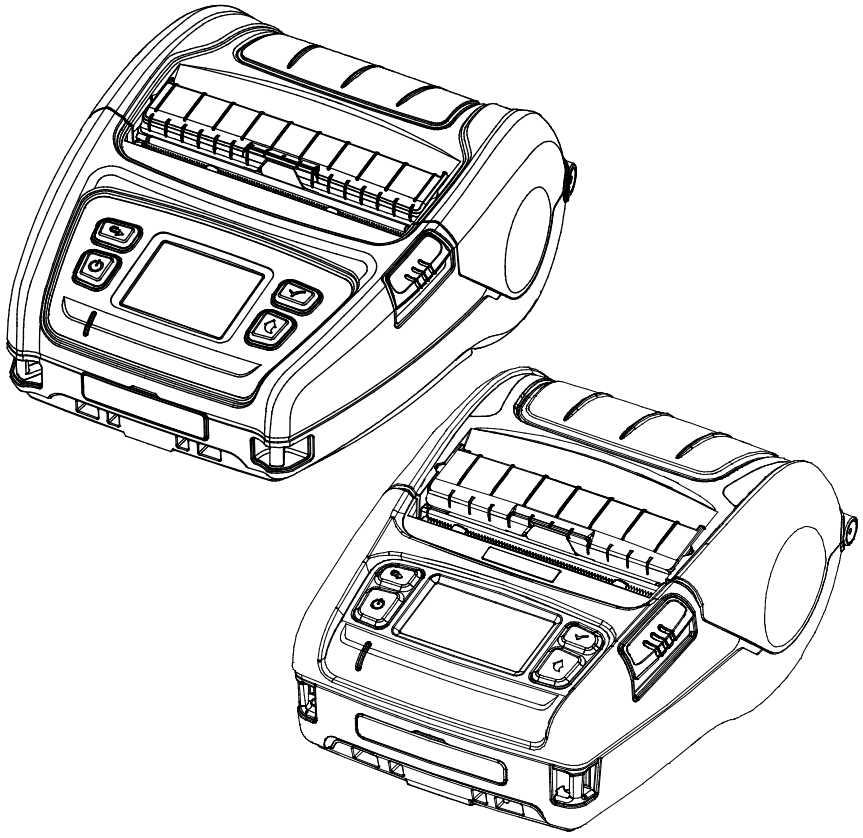


Table of Contents

1. Manual Guide

This SDK manual provides descriptions of contents necessary for developing Xamarin.Forms (Portable Class Library, NET Standard) applications. Xamarin library and Demo application were built using Visual Studio 2015 on Windows, and all the contents in this manual are explained based on Visual Studio 2015 on Windows.

2. Supported Devices

The below table summarizes the supported devices (printers) that are available in this Xamarin SDK.

Printer Name	DPI	Supported Communication	Max Printable Width
PV3	203 dpi	Bluetooth, Wi-Fi	576 dots
PV4	203 dpi	Bluetooth, Wi-Fi	832 dots

3. Xamarin SDK Installation

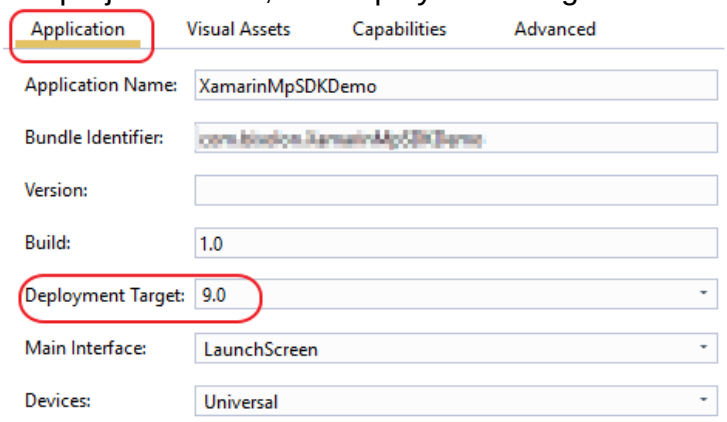
3-1 Supported Platforms

3-1-1 iOS

iOS 9.0 or later

Note

In your Xamarin project for iOS, set Deployment Target version to 9.0.

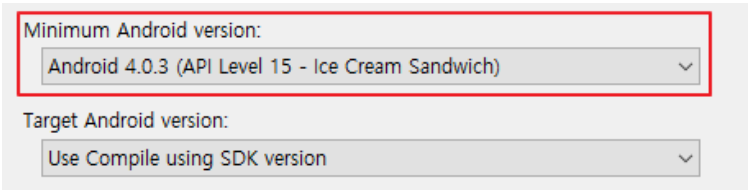


3-1-2 Android

Android 4.0.3(API Level 15: ice cream sandwich) or later

Note

In your Xamarin project for android, set minimum android version to API Level 15.



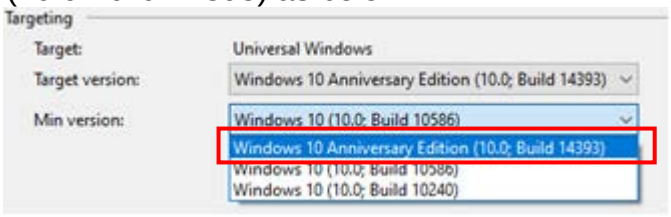
You should leave the target Android version set to Use Compile using SDK version (as shown) so that its value automatically matches the target framework setting. In general, the Target Android Version should be bounded by the Minimum Android Version and the Target Framework. That is: **Minimum Android Version < Target Android Version < Target Framework.**

3-1-3 Windows

Windows 10 version Build 14393 or later.

Note

In your Xamarin project for Universal Windows, set minimum version to Winddows 10 (10.0 Build 14393) as below.



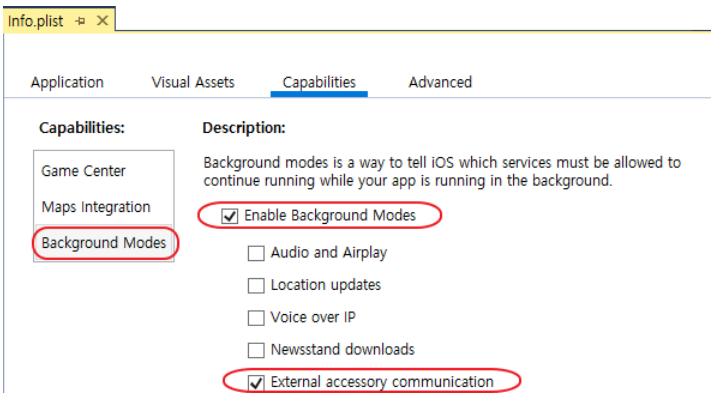
3-2 Supported Communications

3-2-1 Bluetooth

HOST must be paired with Bluetooth device for Bluetooth communication in advance. Make sure to include necessary permissions (or capabilities) to each platform before using APIs in this manual. Please refer to the table following for permission settings.

3-2-2 Wi-Fi

HOST and device (printer) can only communicate over TCP / IP on the same wired / wireless network. Make sure to include necessary permissions (or capabilities) to each platform before using APIs in this manual. Please refer to “3-3” in this manual for permission settings.

<table>
<tr><th>Platform</th><th>Required permission / capabilities</th></tr>
<tr><td rowspan="2">iOS</td><td>1. Update the Info.plist file by opening it in XML editor or text editor. Add the following to the file. This is to allow your app to access the Bluetooth port.
<key>UISupportedExternalAccessoryProtocols</key>
<array><string>com.sato.protocol</string></array>
<key>NSBluetoothAlwaysUsageDescription</key>
<string>Communication with the printer</string></td></tr>
<tr><td>2. Put a checkmark on “External Accessory communication” checkbox as a below.
 
</td></tr>
<tr><td>Android</td><td>Make sure to include following capabilities in the AndroidManifest.xml file.
- BLUETOOTH, BLUETOOTH_ADMIN, BLUETOOTH_PRIVILEGED
- ACCESS_WI-FI_STATE, CHANGE_WI-FI_STATE
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
android:installLocation="auto">
<uses-sdk android:minSdkVersion="15" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.BLUETOOTH_PRIVILEGED" />
<uses-permission android:name="android.permission. ACCESS_WI-FI_STATE " />
<uses-permission android:name="android.permission. CHANGE_WI-FI_STATE " />
</manifest></td></tr>
<tr><td>Windows</td><td>Make sure to include following capabilities in the “Package.appxmanifest” file.
- Bluetooth, Internet (Client), Private Networks (Client & Server)</td></tr>
</table>

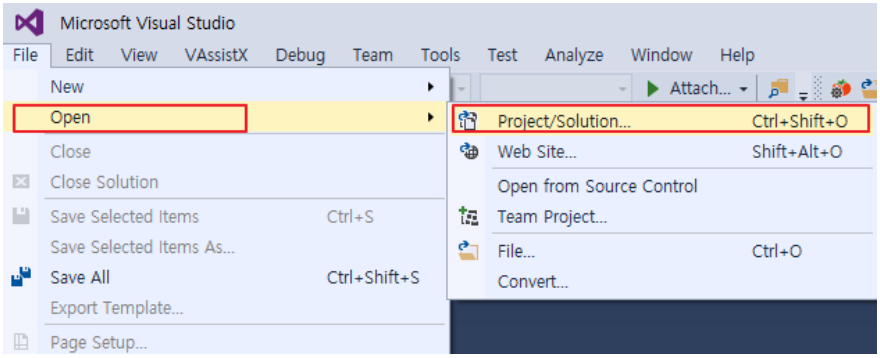
Note	When using Bluetooth and Wi-Fi, there are a few things to modify your project's settings and configuration information related to permissions for each platform. Demo application provided helps you modifying the necessary settings of your own Xamarin application.

3-3 Installation Instruction

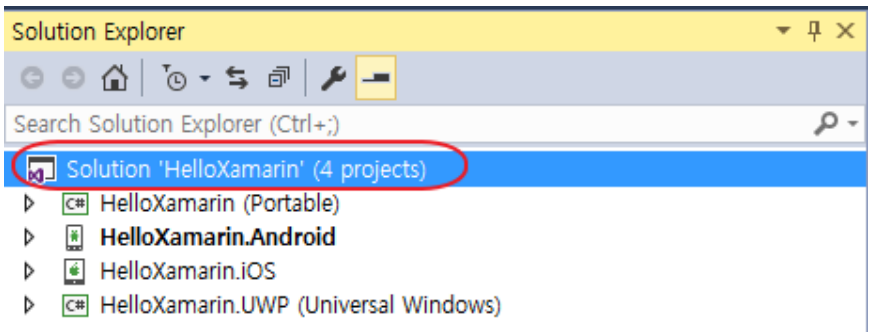
• Xamarin SDK can be installed as a **NuGet** package into MS Visual Studio.
When you use NuGet to install a package, it copies the library files to your solution and automatically updates your project (add references, change config files, etc). If you remove a package,**NuGet** reverses whatever changes it maded. If you need more information about NuGet, visit at https://www.nuget.org/

3-3-1 Installation steps from a local nupkg file

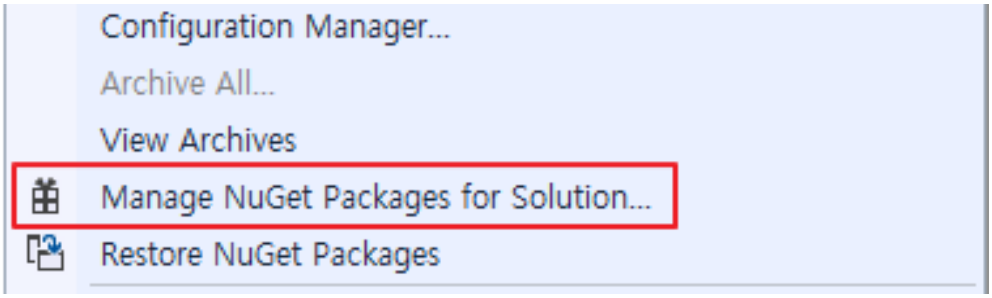
1) Open Visual Studio, and open your own Xamarin.Forms solution.



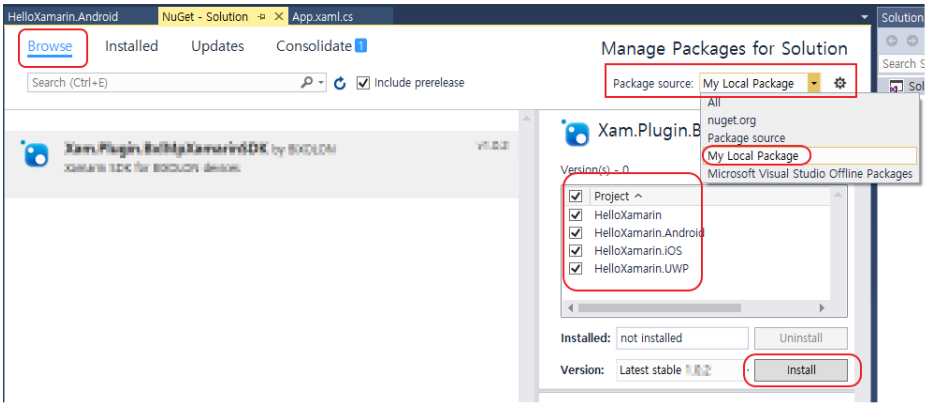
2) From the Solution Explorer, right click the top level solution.



3) Click "**Manage NuGet Packages for Solution…**"



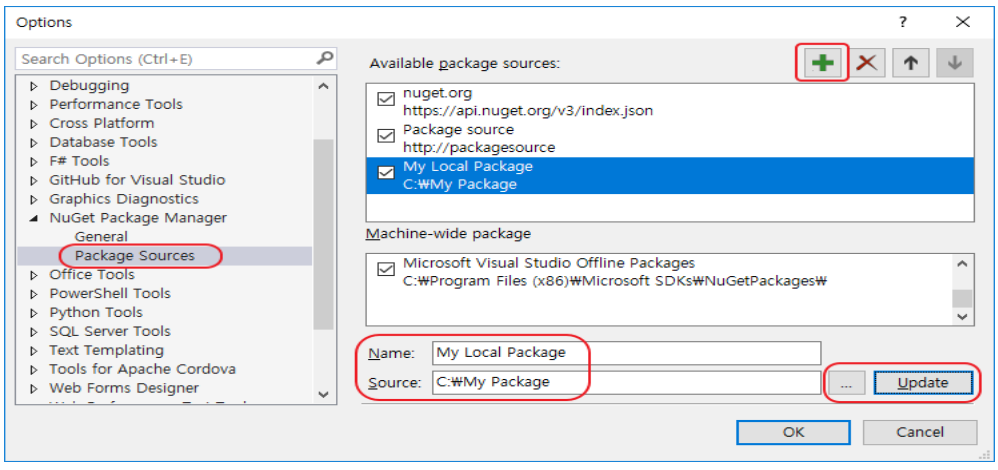
4) On the top of the right side, click "**Setting**" button in red.



5) Click "**Package Sources**", and then click "**+**" button.
6) Browse to the directory where "**Xam.Plugin.SatoMpXamarinSDK.x.x.x.nupkg**" file exists, and then enter any name you want.

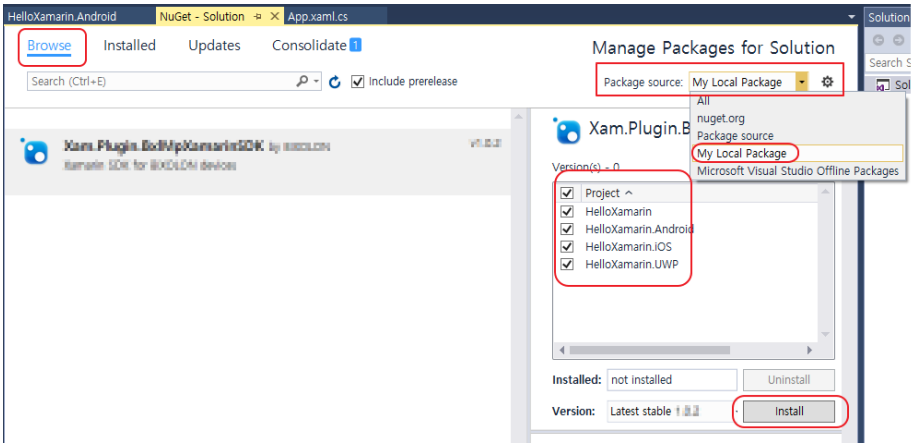
Note	"x.x.x" is the version number for the SDK.

7) Click "**Update**"

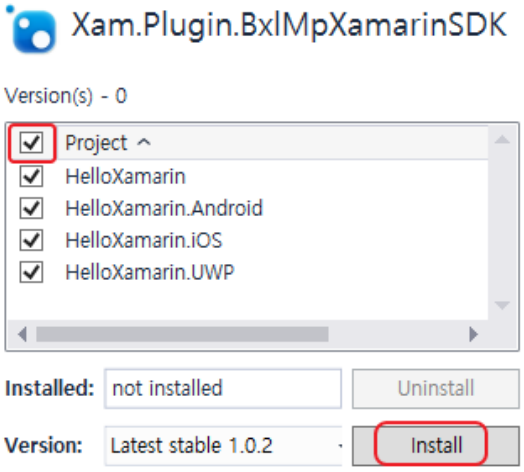


8) Click “**OK**”.

9) “**Xam.Plugin.SatoMpXamarinSDK**” package can be found when you select the name you entered in step 6 in the “**Package source**” list.



10) Click “**Project**” checkbox to install the entire package.



11) Click “**Install**” to start installing all the appropriate references to the libraries for each platform project.

4. ISPVPrinterDeviceFactory Reference

4-1 Overview

• "**ISPVPrinterDeviceFactory**" interface can be used to create an instance implementing "**IPrinterDevice**" interface for communication with a specific device. All instances for device (printer) control are created using the "**createDevice**" method.

4-2 Methods

Note	Use the "**current**" property in "**SPVPrinterDeviceFactory**" class to call the methods defined in this interface. eg) SPVPrinterDeviceFactory.Current.createDevice(10)

4-2-1 createDevice

Gets an instance for communication with a specific device.

[Syntax]
IPrinterDevice createDevice(DeviceType deviceType);

[Parameters]
Device deviceType: A value of the device type to create.

DeviceType Enumeration	Value	Description
LABEL_PRINTER	1	Label Prniter

[Returns]
A reference to the instance implementing "**IPrinterDevice**" interface specified or null.

5. IPrinterDevice Reference

5-1 Overview

• "**IPrinterDevice**" interface provides common member methods with regards to the device, and the properties and methods can be used in all the classes implementing this interface.

5-2 Properties

5-2-1 IsOpen

The device in the usable status or not.

5-2-2 InterfaceType

Communication Interfacetype (Bluetooth, Wi-Fi).

5-3 Methods

5-3-1 selectInterface

Sets the communication interface type and address of the device(printer) to connect.

Note	This method must be used before calling the "**openService**" method.

[Syntax]
int selectInterface(int interfaceType, string address);

[Parameters]
• int interface type: communication type to connect the device.

InterfaceType Enumeration	Value	Description
WI-FI	1	Wi-Fi.
BLUETOOTH	4	Bluetooth.

• string address:
In case of TCP (Wi-Fi) communication, it should be used in the form of "IP address [: port number]". Port number is 9100 by default and can be omitted. In case of Bluetooth communication, it should be entered as the value described in the table below according to platform.

Platform	Address value
iOS	Bluetooth Serial Number
Android	Bluetooth MAC Address
Windows	Bluetooth MAC Address

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

5-3-2 openService

Establishes a communication connection with the device.

[Syntax]
Task<int> **openService**(uint timeout = 3);

[Parameters]
uint timeout: Timeout value for connecting a device.

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

5-3-3 closeService

Disconnects the communication with the device.

[Syntax]
Task<int> closeService (int closeTimeout = 0);

[Parameters]
int closeTimeout: Timeout value for disconnecting a device.

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

5-3-4 setTransaction

Enters or Leaves 'Transaction' mode, and then sends the data in the buffer.

Note	It is recommended to use the Transaction mode when using output devices such as printers.

[Syntax]
Task<int> setTransaction(int mode);

[Parameters]
int mode: Configure the transaction mode

TransactionMode Enumeration	Value	Description
TRANSACTION_OUT	0	Cancel transaction mode after data transmission
TRANSACTION_IN	1	Enter transaction mode

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

6. IPrinterLookup Reference

6-1 Overview

• The "**IPrinterLookup**" interface provides useful methods for searching and retrieving the information needed to connect devices (printers).

6-2 Methods

Note	Use the "**current**" property in "**SPVPrinterLookupUtil**" class to call the methods defined in this interface. eg) SPVPrinterLookupUtil.Current.refreshDevicesList() eg) SPVPrinterLookupUtil.Current.getDeviceList()

6-2-1 refreshDevicesList

Starts discovering devices available to connect with HOST.

[Syntax]
Task<int> refreshDevicesList (int interfaceType)

[Parameters]
int interface type: communication type searching for devices.

InterfaceType Enumeration	Value	Description
WI-FI	1	Wi-Fi (wireless LAN)
BLUETOOTH	4	Bluetooth

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

6-2-2 getDevicesList

Gets the data collctions containing the connection information discovered by "**refreshDeviceList**" method.

[Parameters]
int interface type: communication type for getting device list

InterfaceType Enumeration	Value	Description
WI-FI	1	Wi-Fi (wireless LAN)
BLUETOOTH	4	Bluetooth

[Syntax]
ObservableCollection<PrinterConnectionInformation> getDevicesList (int interfaceType)

[Returns]
ObservableCollection<PrinterConnectionInformation>: data collctions containing the connection information.

```
class PrinterConnectionInformation
{
    public InterfaceType IntefaceType { get; set; }
    public string Name { get; set; }
    public string Address
    public string MacAddress { get; set; }
    public string IpAddress { get; set; }
    public string PortNumber { get; set; }
}
```

Properties
• InterfaceType: The communication type supported.
• Name: The display name for the device.
• Address: The address for the device (IP Address, S/N, or MAC address)
• MACAddress: MAC address or S/N for Bluetooth device.
• IPAddress: IP address for TCP communication.
• PortNumber: Port number for TCP communication.

7. ILabelPrinterDevice Reference

7-1 Overview

• The ILabelPrinterDevice interface provides methods for controlling a Label printer.

7-2 Methods

7-2-1 setTextEncoding

Sets the codepage used for encoding characters.

[Syntax]
int setTextEncoding(int textEncoding);

[Parameters]
• int textEncoding: Code page to use when encoding a character string.

LabelCodePage Enumeration	Value	Description
CP437	437	USA, Standard Europe
CP850	850	Western European
CP852	852	Latin 2
CP860	860	Portuguese
CP863	863	Canadian-French
CP865	865	Nordic
WPC1252	1252	Latin 1
CP857	857	Turkish
CP737	737	Greek (PC 737)
WPC1250	1250	Central European
WPC1253	1253	Greek (ANSI 1253)
WPC1254	1254	Turkish
CP855	855	Cyrillic
CP862	862	Hebrew DOS code
CP866	866	Cyrillic #2
WPC1251	1251	Cyrillic
WPC1255	1255	Hebrew New code
CP928	928	Greek
CP775	775	Baltic
WPC1257	1257	Baltic
CP858	858	Euro
GB2312	936	Simplified Chinese
BIG5	950	Traditional Chinese
KS5601	949	Korean
SHIFTJIS	932	Japanese

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-2 setCharacterset

Sets the codepage and international characterset used for encoding characters.

[Syntax]
Task<int> setCharacterset(int characterset, int internationalCharset);

[Parameters]
• int characterset: Code page to use when encoding a character string

LabelCodePage Enumeration	Value	Description
CP437	437	USA, Standard Europe
CP850	850	Western European
CP852	852	Latin 2
CP860	860	Portuguese
CP863	863	Canadian-French
CP865	865	Nordic
WPC1252	1252	Latin 1
CP857	857	Turkish
CP737	737	Greek (PC 737)
WPC1250	1250	Central European
WPC1253	1253	Greek (ANSI 1253)
WPC1254	1254	Turkish
CP855	855	Cyrillic
CP862	862	Hebrew DOS code
CP866	866	Cyrillic #2
WPC1251	1251	Cyrillic
WPC1255	1255	Hebrew New code
CP928	928	Greek
CP775	775	Baltic
WPC1257	1257	Baltic
CP858	858	Euro
GB2312	936	Simplified Chinese
BIG5	950	Traditional Chinese
KS5601	949	Korean
SHIFTJIS	932	Japanese

• int internationalCharset: International character set

ICS Enumeration	Value	Description
USA	0	USA Code
FRANCE	1	FRANCE Code
GERMANY	2	GERMANY Code
UK	3	UK Code
DENMARK_I	4	DENMARK1 Code
SWEDEN	5	SWEDEN Code
ITALY	6	ITALY Code
SPAIN_I	7	SPAIN1 Code
JAPAN	8	JAPAN Code
NORWAY	9	NORWAY Code
DENMARK_II	10	DENMARK 2 Code
SPAIN_II	11	SPAIN 2 Code
LATIN_AMERICA	12	LATIN AMERICA Code

KOREA	13	KOREA Code
SLOVENIA_CROATIA	14	SLOVENIA Code
CHINA	15	CHINA Code

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-3 setLength

Sets the label length, gap, media type, and offset.

[Syntax]
Task<int> setLength(int labelLength, int gapLength, char mediaType, int offsetLength)

[Parameters]
• int labelLength: Media length
• int gapLength: Gap length of the media
• char mediaType: Media type

LabelMediaType Enumeration	Value	Description
GAP	'G'	Gap paper
CONTINUOUS	'C'	Continuous paper
BLACK_MARK	'B'	Black mark paper

• int offsetLength: Offset length

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-4 setSpeed

Sets the printing speed.

[Syntax]
Task<int> setSpeed(int speed)

[Parameters]
• int speed: printing speed

Printer Name	Speed (inch per ips)	Range
PV3	1 ~ 5 ips	0 ~ 4
PV4	1 ~ 5 ips	0 ~ 4

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-5 setWidth

Sets the print width.

[Syntax]
Task<int> setWidth(int labelWidth)

[Parameters]
• int labelWidth: print width

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-6 setDensity

Sets the print density.

[Syntax]
Task<int> setDensity(int density)

[Parameters]
• int density: print density (Range of 1 to 20)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-7 setMargin

Sets the margins in the printing area.

[Syntax]
Task<int> setMargin(int horizontalMargin, int verticalMargin)

[Parameters]
• int horizontalMargin: Left margin
• int verticalMargin: Top margin

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-8 setOffset

Sets offset length between black mark(or gap) and dotted lines.

[Syntax]
Task<int> setOffset(int offset)

[Parameters]
• int offset: offset value (Range of -100 to 100)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-9 setOrientation

Sets the printing orientation of the label.

[Syntax]
Task<int> setOrientation(char orientation)

[Parameters]
• char orientation: Printing orientation

LabelOrientation Enumeration	Value	Description
TOP_TO_BOTTOM	'T'	Print from top to bottom
BOTTOM_TO_TOP	'B'	Print from bottom to top

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-10 setCuttingPosition

Sets the cutting position or tear-off position after printing labels.

[Syntax]
Task<int> setCuttingPosition(int cuttingPosition)

[Parameters]
• int position: Cutting position value(Range of -100 to 100).

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-11 checkPrinterStatus

Gets the printer's current status.

[Syntax]
Task<uint> checkPrinterStatus(void);

[Returns]

LabelPrinterStatus Enumeration	Value	Description
IDLE	0	Printing is possible
PAPER_EMPTY	1	No paper
COVER_OPEN	2	Cover is open
TPH_OVERHEAT	8	TPH overheating
GAP_ERROR	16	Gap Detection Error(Auto-sensing failure)
BUILDING_LABEL_TO_BE_PRINTED	64	On building label to be printed in the buffer
PRINTING_LABEL	128	On printing label in image buffer
ISSUED_LABEL_ISPAUSED	256	Issued label is paused in peeler unit
MOTOR_OVERHEAT	512	Motor overheat
BOARD_OVERHEAT	1024	Board overheat
WAIT_FOR_PAPER_TO_BE_TAKEN	2048	Wait for paper to be taken

7-2-12 getModelName

Gets the printer's model name.

[Syntax]
Task<int> getModelName (Action<string> modelDelegate)

[Parameters]
Action<string> modelDelegate: A delegate with a parameter used for getting a model name.

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-13 getFirmwareVersion

Gets the printer's firmware version.

[Syntax]
Task<int> getFirmwareVersion(Action<string> versionDelegate)

[Parameters]
Action<string> versionDelegate: A delegate with a parameter used for getting a firmware version.

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-14 printBuffer

Starts printing the contents saved in the printer

[Syntax]
Task<int> printBuffer(int numberOfCopies);

[Parameters]
• int numberOfCopies: The number of copies (Range: 1 to 65535)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-15 printRawData

Sends user-defined data to the printer.

[Syntax]
Task<int> printRawData(byte[] data);

[Parameters]
• byte[] data: Data to be sent

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-16 drawTextDeviceFont

Saves data of the character strings to the printer buffer using the bitmap font.

[Syntax]
Task<int> drawTextDeviceFont(string data, int xPos, int yPos, char fontSelection,
int fontWidth, int fontHeight, int rightSpace, int rotation,
bool reverse, bool bold, bool rightToLeft,
int alignment);

[Parameters]
• string data: data for characters to be printed
• int xPos: x axis coordinates of the character string
• int yPos: y axis axis coordinates of the character string
• char fontSelection: font selection

LabelBitmapFont Enumeration	Value	Description
DEVICE_FONT_6PT	'0'	9 X 15 (dots)
DEVICE_FONT_8PT	'1'	12 X 20 (dots)
DEVICE_FONT_10PT	'2'	16 X 25 (dots)
DEVICE_FONT_12PT	'3'	19 X 30 (dots)
DEVICE_FONT_15PT	'4'	24 X 38 (dots)
DEVICE_FONT_20PT	'5'	32 X 40 (dots)
DEVICE_FONT_30PT	'6'	48 X 76 (dots)
DEVICE_FONT_14PT	'7'	22 X 34 (dots)
DEVICE_FONT_18PT	'8'	28 X 44 (dots)
DEVICE_FONT_24PT	'9'	37 X 58 (dots)
DEVICE_FONT_KOREAN1	'a'	16 X 16 (dots) (English. Numbers 9 X 15)
DEVICE_FONT_KOREAN2	'b'	24 X 24 (dots) (English. Numbers 12 X 24)
DEVICE_FONT_KOREAN3	'c'	20 X 20 (dots) (English. Numbers 12 X 20)
DEVICE_FONT_KOREAN4	'd'	26 X 26 (dots) (English. Numbers 16 X 30)
DEVICE_FONT_KOREAN5	'e'	20 X 26 (dots) (English. Numbers 16 X 30)
DEVICE_FONT_KOREAN6	'f'	38 X 38 (dots) (English. Numbers 22 X 34)
DEVICE_FONT_GB2312	'm'	24 X 24 (dots) (English. Numbers 12 X 24)
DEVICE_FONT_BIG5	'n'	24 X 24 (dots) (English. Numbers 12 X 24)
DEVICE_FONT_SHIFT_JIS	'j'	24 X 24 (dots) (English. Numbers 12 X 24)

• int fontWidth: The width expansion ratio of a character. (Range: 1 to 9)
• int fontHeight: The height expansion ratio of a character. (Range: 1 to 9)

• int rightSpace: The space on the right of a character (default value: 0)
• int rotation: The printing direction of characters

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

• bool reverse: prints characters with reverse property or not
• bool bold: prints characters with bold property or not
• bool rightToLeft: direction of printing characters (left to right or right to left)
• int alignment: alignment

LabelAlignment Enumeration	Value	Description
LEFT	0	Align to the left
RIGHT	1	Align to the right

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-17 drawTextVectorFont

Saves data of the character strings to the printer buffer using the vector font.

[Syntax]

Task<int> drawTextVectorFont (string data, int xPos, int yPos, char fontSelection,
int fontWidth, int fontHeight, int rightSpace, int rotation,
bool reverse, bool bold, bool italic,
bool rightToLeft, int alignment)

[Parameters]

• string data: data for characters to be printed
• int xPos: x axis coordinates of the character string
• int yPos: y axis axis coordinates of the character string
• char fontSelection: font selection

LabelVectorFont Enumeration	Value	Description
VECTOR_FONT_ASCII	'U'	ASCII (1Byte code)
VECTOR_FONT_KS5601	'K'	KS5601 (2Byte code)
VECTOR_FONT_BIG5	'B'	BG5 (2Byte code)
VECTOR_FONT_GB2312	'G'	GB2312 (2Byte code)
VECTOR_FONT_SHIFT_JIS	'J'	Shift-JIS (2Byte code)
VECTOR_FONT_OCR_A	'a'	OCR-A (1Byte code)
VECTOR_FONT_OCR_B	'b'	OCR-B (1Byte code)

• int fontWidth: character width
• int fontHeight: character height
• int rightSpace: right space of character
• int rotation: printing direction

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

• bool reverse: prints characters with reverse property or not
• bool bold: prints characters with bold property or not
• bool italic: prints characters with italic property or not
• bool rightToLeft: direction of printing characters (left to right or right to left)
• int alignment: alignment

LabelAlignment Enumeration	Value	Description
LEFT	0	Align to the left
RIGHT	1	Align to the right
CENTER	2	Align at the center

[Returns]

If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-18 drawBarcode1D

Saves data of the 1d barcode to the printer buffer.

[Syntax]
Task<int> drawBarcode1D(string data, int xPos, int yPos, int barcodeType, int widthNarrow, int widthWide, int height, int hri, int quietZoneWidth, int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int barcodeType: Type of barcode
• int widthNarrow: Narrow bar width.
• int widthWide: Wide bar width
• int height: Barcode height
• int hri: HRI (Human Readable Interface) printing position

LabelHRI Enumeration	Value	Description
TEXTNONE	0	No HRI print
TEXTBELOW	1	above barcode
TEXTABOVE	2	below barcode
TEXTBELOW_FONTSIZE2	3	above barcode (font size 2)
TEXTABOVE_FONTSIZE2	4	below barcode (font size 2)
TEXTBELOW_FONTSIZE3	5	above barcode (font size 3)
TEXTABOVE_FONTSIZE3	6	below barcode (font size 3)
TEXTBELOW_FONTSIZE4	7	above barcode (font size 4)
TEXTABOVE_FONTSIZE4	8	below barcode (font size 4)

• int quietZoneWidth: Quiet zone width (Range of 0 to20)
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-19 drawBarcodeMaxiCode

Saves data of the Maxicode barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeMaxiCode(string data, int xPos, int yPos, int mode)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int mode: Maxicode mode

LabelMaxicodeMode Enumeration	Value	Description
MAXICODE_MODE_0	0	MaxiCode Mode 0
MAXICODE_MODE_2	2	MaxiCode Mode 2
MAXICODE_MODE_3	3	MaxiCode Mode 3
MAXICODE_MODE_4	4	MaxiCode Mode 4

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-20 drawBarcodePDF417

Saves data of the PDF417 barcode to the printer buffer.

[Syntax]

Task<int> drawBarcodePDF417(string data, int xPos, int yPos, int maximumRowCount,
int maximumColumnCount, int errorCorrectionLevel,
int dataCompressionMethod, bool hri,
int barcodeOriginPoint, int moduleWidth, int barHeight,
int rotation)

[Parameters]

• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int maximumRowCount: Number of barcode Rows. (Range: 3 to 90)
• int maximumColumnCount: Number of barcode Columns. (Range: 1 to 30)
• int errorCorrectionLevel: Error correction level(Recognition rate when a part of a barcode is damaged)

LabelPDF417ECL Enumeration	Value	Description
LEVEL0	0	Error correction level 0
LEVEL1	1	Error correction level 1
LEVEL2	2	Error correction level 2
LEVEL3	3	Error correction level 3
LEVEL4	4	Error correction level 4
LEVEL5	5	Error correction level 5
LEVEL6	6	Error correction level 6
LEVEL7	7	Error correction level 7
LEVEL8	8	Error correction level 8

• int dataCompressionMethod: data compression method

LabelPDF41Compress Enumeration	Value	Description
COMPRESSION_TEXT	**0**	2 characters per codeword
COMPRESSION_NUMERIC	1	2.93 characters per codeword
COMPRESSION_BINARY	2	1.2 characters per codeword

• int hri : HRI (Human Readable Interface) printing position

LabelHRI Enumeration	Value	Description
TEXTNONE	0	Does not print
TEXTBELOW	2	Print below barcode

• int startPosition: starting point of a barcode (default value: 1)

LabelPDF41StartPosition Enumeration	Value	Description
ORIGIN_POINT_CENTER	0	Coordinate value based on middle of barcode
ORIGIN_POINT_UPPER	1	Coordinate value based on top left of barcode

• int moduleWidth: width of the barcode module. (Range: 2 to 9)
• int barHeight: height of the bar. (Range: 4 to 99)
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree

DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-21 drawBarcodeQRCode

Saves data of the QRCode barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeQRCode(string data, int xPos, int yPos, int size, int model,
int errorCorrectionLevel, int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int size: QRCode size (Range of 1 to 9)
• int model: QRCode Model

LabelQRCodeModel Enumeration	Value	Description
MODEL_1	1	Model 1
MODEL_2	2	Model 2

• int errorCorrectionLevel: Error correction level

LabelQRCodeECL Enumeration	Value	Description
ECCLEVEL_L	'L'	Error correction rate is configured to 7%
ECCLEVEL_M	'M'	Error correction rate is configured to 15%
ECCLEVEL_Q	'Q'	Error correction rate is configured to 25%
ECCLEVEL_H	'H'	Error correction rate is configured to 35%

• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-22 drawBarcodeDataMatrix

Saves data of the DataMatrix barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeDataMatrix(string data, int xPos, int yPos, int size, bool reverse,
int rotation);

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int size: DataMatrix size (Range of 1 to 4)
• int reverse : barcode reverse or normal
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-23 drawBarcodeAztec

Saves data of the Aztec barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeAztec(string data, int xPos, int yPos, int size,
bool extendedChannel, int errorCorrectionLevel,
bool menuSymbol, int numberOfSymbols, string optionalID,
int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int size: Aztec size (Range of 1 to 10)
• bool extendedChannel: Extended channel interpretation code
• int errorCorrectionLevel: Error correction level

Value	Description
0	Automatically configures the error correction level.
1~99	Directly enter the error correction level.
101~104	1~4 layer compact symbol
201~232	1~32 layer compact symbol
300	Simple Aztec ”Rune”

• bool menuSymbol: Configures the menu symbol
• int numberOfSymbols: Number of symbols for structured append(Range of 1 to 26)

• string optionalID: Optional ID field for structured append: ID field string(Maximum 24 letters)
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-24 drawBarcodeCode49

Saves data of the Code49 barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeCode49(string data, int xPos, int yPos, int widthNarrow, int widthWide, int height, int hri, int startingMode, int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int widthNarrow: Width of Narrow Bar
• int widthWide: Width of Wide Bar
• int height: Barcode height
• int hri: HRI (Human Readable Interface) printing position.

LabelHRI Enumeration	Value	Description
TEXTNONE	0	Does not print
TEXTABOVE	1	Print above barcode
TEXTBELOW	2	Print below barcode

• int startingMode: Starting Mode of a barcode

LabelCode49Mode Enumeration	Value	Description
REGULAR_ALPHANUMERIC	0	Regular Alphanumeric Mode
MULTIPLE_READ_ALPHANUMERIC	1	Multiple Read Alphanumeric
REGULAR_NUMERIC	2	Regular Numeric Mode
GROUP_ALPHANUMERIC	3	Group Alphanumeric Mode
REGULAR_ALPHANUMRIC_SHIFT1	4	Regular Alphanumeric Shift 1
REGULAR_ALPHANUMRIC_SHIFT2	5	Regular Alphanumeric Shift 2
AUTOMATIC	7	Automatic Mode

• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-25 drawBarcodeCodaBlock

Saves data of the CodaBlock barcode to the printer buffer.

[Syntax]

Task<int> drawBarcodeCodaBlock(string data, int xPos, int yPos, int widthNarrow,
int widthWide, int height, bool securityLevel,
int numberOfCharactersPerRow, char mode,
int numberOfRowToEncode)

[Parameters]

• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int widthNarrow: Width of Narrow Bar
• int widthWide: Width of Wide Bar
• int height: Barcode height
• bool securityLevel: Barcode security level
• int numberOfCharactersPerRow: Number of characters per row(Range of 2 to 62)
• char mode: Codablock mode

LabelCodaBlockMode Enumeration	Value	Description
MODE_A	'A'	Creates a code that uses the Code 39 character set
MODE_E	'E'	Creates a code that uses the Code 128 character set
MODE_F	'F'	Creates a code that uses the Code 128 character set and that automatically has func1 added to

• int numberOfRowToEncode: The following values can be used for each mode.

[Returns]

If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-26 drawBarcodeMicroPDF

Saves data of the MicroPDF barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeMicroPDF(string data, int xPos, int yPos, int moduleWidth,
int moduleHeight, int mode, int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int moduleWidth: module width(Range of 2 to 8)
• int moduleHeight: barcode height(Range of 1 to 99)
• int mode: barcode mode(Range of 0 to 33)
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-27 drawBarcodeIMB

Saves data of the IMB barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodeIMB(string data, int xPos, int yPos, bool hri, int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• bool hri: Determines whether or not to print HRI (Human Readable Interface)
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-28 drawBarcodeMSI

Saves data of the MSI barcode to the printer buffer.

[Syntax]

```
Task<int> drawBarcodeMSI(string data, int xPos, int yPos, int narrowWidth, int widthWide,
                         int height, int checkDigitSelection, bool printCheckDigit, int hri,
                         int rotation);
```

[Parameters]

- string data: data of the barcode
- int xPos: x axis coordinates of barcode
- int yPos: y axis coordinates of barcode
- int widthNarrow: Width of Narrow Bar
- int widthWide: Width of Wide Bar
- int height: Barcode height
- int checkDigitSelection: Check digit mode

LabelMSICheckDigit Enumeration	Value	Description
CHECKDIGIT_NONE	0	Check digit not configured
CHECKDIGIT_1MOD10	1	1 Mod 10
CHECKDIGIT_2MOD10	2	2 Mode 10
CHECKDIGIT_1MOD11_AND_1MOD_10	3	1 Mod 11 and Mod 10

- bool printCheckDigit: Determines whether or not to print check digit
- int hri: HRI (Human Readable Interface) printing position.

LabelHRI Enumeration	Value	Description
TEXTNONE	0	Does not print
TEXTABOVE	1	Print above barcode
TEXTBELOW	2	Print below barcode

- int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]

If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-29 drawBarcodePlessey

Saves data of the Plessey barcode to the printer buffer.

[Syntax]
Task<int> drawBarcodePlessey(string data, int xPos, int yPos, int narrowWidth, int widthWide, int height, bool printCheckDigit, int hri, int rotation)

[Parameters]
• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int widthNarrow: Width of Narrow Bar
• int widthWide: Width of Wide Bar
• int height: Barcode height
• bool printCheckDigit: Determines whether or not to print check digit
• int hri: HRI (Human Readable Interface) printing position.

Code	Value	Description
TEXTNONE	0	Does not print
TEXTABOVE	1	Print above barcode
TEXTBELOW	2	Print below barcode

• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-30 drawBarcodeTLC39

Saves data of the TLC39 barcode to the printer buffer.

[Syntax]

Task<int> drawBarcodeTLC39(string data, int xPos, int yPos, int narrowWidth, int widthWide,
int height, int rowHeightOfMicroPDF417,
int narrowWidthOfMicroPDF417, int rotation)

[Parameters]

- string data: data of the barcode
- int xPos: x axis coordinates of barcode
- int yPos: y axis coordinates of barcode
- int narrowWidth: Width of Narrow Bar
- int widthWide: Width of Wide Bar
- int height: Barcode height
- int rowHeightOfMicroPDF417: Height of Micro PDF417's row(Range of 1 to 255)
- int narrowWidthOfMicroPDF417: Width of Micro PDF417's Narrow bar(Range of 1 to 10)
- int rotation : Printing direction of barcode to be printed

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]

If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-31 drawBarcodeRSS

Saves data of the RSS barcode to the printer buffer.

[Syntax]

Task<int> drawBarcodeRSS(string data, int xPos, int yPos, int barcodeType,
int magnification, .int separatorHeight, int barcodeHeight,
int segmentWidth, int rotation)

[Parameters]

• string data: data of the barcode
• int xPos: x axis coordinates of barcode
• int yPos: y axis coordinates of barcode
• int barcodeType: Configures the barcode type.

LabelRSSType Enumeration	Value	Description
RSS14	0	RSS14
RSS14_TRUNCATED	1	RSS14 truncated
RSS14_STACKED	2	RSS14 stacked
RSS14_STACKED_OMNIDIRECTIONAL	3	RSS14 Stacked omnidirectional
RSS_LIMITED	4	RSS limited
RSS_EXPANDED	5	RSS Expanded
UPC_A	6	RSS UPC A
UPC_E	7	RSS UPC E
EAN13	8	EAN13
EAN8	9	EAN 8
UCC_EAN128_CC_A_B	10	EAN128 CC-A/B
UCC_EAN128_CC_C	11	EAN128 CC-C

• int magnification: Can use a number between 1 and 10
• int separatorHeight: Configures the height of an RSS barcode identifier(Range of 0 to 22)
• int barcodeHeight: Configures the barcode height
• int segmentWidth: Configures segment width(even number between 0 and 22)
• int rotation: printing direction of barcode

LabelRotation Enumeration	Value	Description
DEGREES_0	0	No rotated
DEGREES_90	1	Rotated 90 degree (clockwise)
DEGREES_180	2	Rotated 180 degree
DEGREES_270	3	Rotated 270 degree (clockwise)

[Returns]

If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-32 drawBlock

Saves data of a quadrangle or line to the printer buffer.

[Syntax]
Task<int> drawBlock(int startPosX, int startPosY, int endPosX, int endPosY, string option,
int thickness)

[Parameters]
• int startPosX: x axis coordinates of top left
• int startPosY: y axis coordinates of top left
• int endPosX: x axis coordinates of bottom right
• int endPosY: y axis coordinates of bottom right
• char option: drawing option

LabelBlock Enumeration	Value	Description
LINE_OVERWRITING	'O'	Repeat drawing on places where lines overlap
LINE_EXCLUSIVE_OR	'E'	Do not draw on places where lines overlap
LINE_DELETE	'D'	Delete line
SLOPE	'S'	Diagonal line
BOX	'B'	Quadrangle edge

• int thickness: Selects line thickness. This only applies if the 'option' value is 'S' or 'B'.

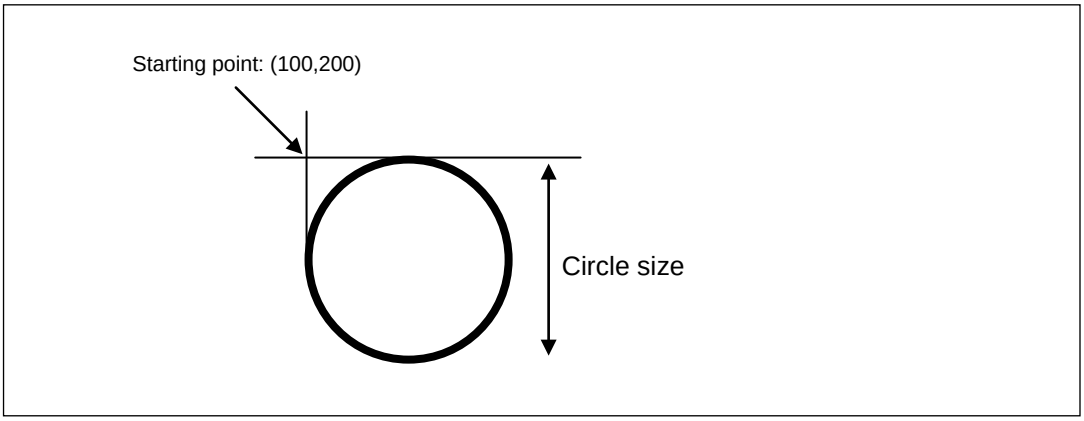
[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-33 drawCircle

• Saves data of a circle to the printer buffer.

[Syntax]
Task<int> drawCircle(int startPosX, int startPosY, int sizeSelection, int multiplier)

[Parameters]
• int startPosX: x axis coordinates of starting point* of circle region
Add starting point (intersection point of tangent of left and top) description
• int startPosY: y axis coordinates of starting point* of circle region
* Starting point: Intersection point of tangent of left and top



• int sizeSelection: size of circle

LabelCircle Enumeraton	Value	W x H Size (dots)
SIZE_40X40	1	40 × 40
SIZE_56X56	2	56 × 56
SIZE_72X72	3	72 × 72
SIZE_88X88	4	88 × 88
SIZE_104X104	5	104 × 104
SIZE_168X168	6	168 × 168

• int multiplier: Expands a circle by the scaling unit (Range of 1 to 4)

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-34 drawImage

Saves data of an image to the printer buffer.

[Syntax]
Task<int> drawImage(object bitmapSrc, int startPosX, int startPosY, int width, int brightness,
bool isDithering, bool isCompress)

[Parameters]
• object bitmapSrc: A platform dependent image object.

Note	iOS		UIKit.UIImage
	Android		Android.Graphics.Bitmap
	Windows		Windows.UI.Xaml.Media.Imaging.WriteableBitmap

• int startPosX: x axis coordinates of top left of image to be printed
• int startPosY: y axis coordinates of top left of image to be printed
• int width: Width of printed image
• int brightness: The brightness level (Range: 0 ≤ brightness ≤ 100)

Platform	Brightness Range
Android, UWP	0 ≤ brightness ≤ 100
iOS	-100 ≤ brightness ≤ 100

• bool isDithering: Whether or not to apply dithering
• bool isCompress: Whether or not to compress data

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

7-2-35 drawImageFile

Saves data of the image file specified by the path to the printer buffer.

[Syntax]
Task<int> drawImageFile(string filePath, int startPosX, int startPosY, int width, int brightness, bool isDithering, bool isCompress)

[Parameters]
• string filePath: The path of the image file

Note	In Android, the image file should be in the SD card.

• int startPosX: x axis coordinates of top left of image to be printed
• int startPosY: y axis coordinates of top left of image to be printed
• int width: Width of printed image
• int brightness: The brightness level

Platform	Brightness Range
Android, UWP	0 ≤ brightness ≤ 100
iOS	-100 ≤ brightness ≤ 100

• bool isDithering: Whether or not to apply dithering
• bool isCompress: Whether or not to compress data

[Returns]
If the method succeeds, the return value is zero(0). Other cases, returns non-zero.

8. Appendix

8-1 Error Code Table

• The error code table below summarizes the values returned when calling the API provided in this manual.

ResultCode Enumeration	Value	Description
SUCCESS	0	Suceess of the method Operation
ALREADY_OPEN	1	Already Open
FAIL	1000	Failure of the method Operation
FAIL_INVALID_INTERFACE	1001	Not Supported Communication Interface Type
FAIL_NO_OPEN	1002	Tried to access the device which is not opened
FAIL_NOT_SUPPORT	1003	Not supported API by the device
FAIL_INVALID_PARAMETER	1004	Invalid parameter for the method
FAIL_NO_RESPONSE	1005	No response from the device
FAIL_NOT_CONNECT	1006	Failed to connect the device
FAIL_NO_FILE	1008	No file in the the path specified
FAIL_NOT_SUPPORT_CODEPAGE	1012	Not supported code page
FAIL_NOT_SUPPORT_ICS	1013	Not supported code international code page
PAGEMODE_ALREADY_IN	1014	Already entered in page mode.
TRANSACTION_ALREADY_IN	1015	Already entered in transactoin mode.
FAIL_NOT_SUPPORT_ESCSEQ	1017	Not supported escape sequence

FAIL_NO_DEVICE_FOUND	5000	No device found
FAIL_NOT_PERMISSION_ALLOWD	5001	Failed to get permission required
FAIL_NOT_SUPPORTED_IMAGE _TYPE	5002	Not supported file extention
FAIL_IMAGE_DECODE	5003	Failed to decode image in BASE64

Copyright

SATO maintains ongoing efforts to enhance and upgrade the functions and quality of all our products.
In the following, product specifications and/or user manual content may be changed without prior notice.

Caution

Some semiconductor devices are easily damaged by static electricity. You should turn the printer “OFF”, before you connect or remove the cables on the rear side, in order to guard the printer against the static electricity. If the printer is damaged by the static electricity, you should turn the printer “OFF”.

Revision History

Rev.	Date	Page	Description
1.00	19.10.17	-	New
1.01	20.04.17	-	Added PV4
		5	Modify Supported Platforms for UWP
		17	Modify max speed