

# Woosim iOS SDK Programmer Reference

---

Version 3.0  
December 2021



## Contents

<b>1. OVERVIEW .....</b>	<b>3</b>
1.1. INTRODUCTION.....	3
1.2. PRINTER MODE .....	3
1.3. GET STARTED .....	3
1.4. BLUETOOTH COMMUNICATION.....	4
1.5. DEFINITIONS AND ABBREVIATIONS .....	5
<b>2. CLASS METHOD LIST .....</b>	<b>6</b>
2.1. WSCMD CLASS.....	6
2.2. WSIMAGE CLASS .....	7
2.3. WSPACKET CLASS .....	8
<b>3. WSCMD CLASS.....</b>	<b>9</b>
3.1. ENUM .....	9
3.2. METHOD.....	11
3.2.1. <i>Printer Operation</i> .....	11
3.2.2. <i>Text Attribute</i> .....	13
3.2.3. <i>Barcode</i> .....	18
3.2.4. <i>Page Mode</i> .....	25
3.2.5. <i>Card Operation</i> .....	27
<b>4. WSIMAGE CLASS .....</b>	<b>30</b>
4.1. METHOD.....	30
<b>5. WSPACKET CLASS .....</b>	<b>33</b>
5.1. ENUM .....	33
5.2. PROPERTY.....	33
5.3. METHOD.....	33
<b>6. SAMPLES .....</b>	<b>34</b>
6.1. BTPRINT .....	34
6.2. WIFIPRINT.....	34

# 1. Overview

---

Copyright © 2021 Woosim Systems Inc.

## 1.1. Introduction

This Woosim iOS Software Development Kit (SDK) document provides information about iOS application development using Woosim printers.

The static library `libwoosimmm.a` included in the SDK is developed based on Swift and supports device and simulator. This library is not compatible with same library version 2.x.

This SDK was developed under below environment.

Platform OS	macOS 12.0.1
Development Tool	Xcode 13.1
Language	Swift 5
iOS Deployment Target	iOS 12.0

Woosim printers are equipped with several types of MCU. The printer's MCU can be checked through the Self-Test function. If you turn on the printer while pressing the <FEED> button, the Self-Test function will work.

The SDK supports M16C, ARM, and RX MCU. All APIs work properly with RX MCU, but some APIs do not with M16C or ARM MCU.

Most library APIs create a printer command or a collection of printer commands. To use commands that are not provided in the library API, refer to the document "Woosim Command Manual".

## 1.2. Printer Mode

Woosim printers provide two modes, Standard mode and Page mode.

In the Standard mode, data sent to the printer is printed immediately.

In the Page mode, data sent to the printer is drawn in the designated area to be printed. The drawn content is printed at once when a specific printer command is received. The library APIs that are only valid in the Page mode start with "pageMode".

## 1.3. Get Started

This SDK includes several sample projects. You can refer to how to use the library through these examples.

To use the library in a project, the library must be registered in the project.

▼ Frameworks, Libraries, and Embedded Content

Name	Embed
 libwoosim300.a	
+ -	

And enter the **Library Search Paths** in the **Build Settings**. The figure below is an example taken from the sample project included in the SDK.

▼ Search Paths

Setting	 btprint
Always Search User Paths (Deprecated)	No ▾
Framework Search Paths	
Header Search Paths	
▶ Library Search Paths	<code>\$(PROJECT_DIR)/../../Library</code>
Rez Search Paths	

To use the library properly, you need not only the static library file, but also the contents of the subfolders. When copying the library to another location, please copy the entire “Library” folder in the SDK.

For Swift project, enter the **Import Paths** for Swift compiler in the **Build Settings**.

▼ Swift Compiler - Search Paths

Setting	 btprint
▶ Import Paths	<code>\$(PROJECT_DIR)/../../Library</code>

For Objective-C project, add empty Swift file and create bridging header. You may see the below popup message.



## 1.4. Bluetooth Communication

If your application connects to a Woosim printer through Bluetooth, you need to add a protocol string to the **info.plist** file or Info section of Target menu.

- 1) Open **info.plist** file in Xcode.
- 2) Add **Supported external accessory protocols** key in Information Property List.
- 3) Enter "**com.woosim.wspr240**" as protocol string value.

Key	Type	Value
▼ Information Property List	Dictionary	(15 items)
Localization native development re...	String	\$(DEVELOPMENT_LANGUAGE)
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	\$(PRODUCT_BUNDLE_PACKAGE_TYPE)
Bundle version string (short)	String	1.0
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
► Required device capabilities	Array	(1 item)
▼ Supported external accessory prot...	Array	(1 item)
Item 0	String	com.woosim.wspr240
► Supported interface orientations	Array	(1 item)
► Supported interface orientations (i...	Array	(4 items)

If you want to upload the app that communicates with a Woosim printer via Bluetooth to the Apple App Store, the app bundle ID should be registered on the MFi site, so please contact the Sales Department of Woosim Systems Inc.

## 1.5. Definitions and Abbreviations

API	Application Interface Unit
DBCS	Double Bytes Character Sets
ECC	Error Correction Code
HRI	Human Readable Interpretation
MCU	Main Control Unit
MSR	Magnetic Stripe Reader
SDK	Software Development Kit
TTF	True Type Font

---

## 2. Class Method List

---

### 2.1. WSCmd Class

```
static func alignContent(_: AlignType) -> Data
static func alignText(_: AlignType) -> Data
static func cutPaper(_: CutType) -> Data
static func exitMsrMode() -> Data
static func exitNonSecureScrMode() -> Data
static func exitScrMode() -> Data
static func extendFont(width: UInt8, height: UInt8) -> Data
static func feedToMark() -> Data
static func initPrinter() -> Data
static func moveAbsPosition(_: UInt16) -> Data
static func moveRelPosition(_: UInt16) -> Data
static func openCashDrawer() -> Data
static func pageModeDeleteData() -> Data
static func pageModeMoveAbsVertical(_: UInt16) -> Data
static func pageModeMoveRelVertical(_: UInt16) -> Data
static func pageModePrintData() -> Data
static func pageModePrintStdMode() -> Data
static func pageModeSetArea(x: UInt16, y: UInt16, width: UInt16, height: UInt16) -
> Data
static func pageModeSetDirection(_: PageDirection) -> Data
static func pageModeSetPosition(x: UInt16, y: UInt16) -> Data
static func pageModeToStdMode() -> Data
static func printData() -> Data
static func printFeed(dot: UInt8) -> Data
static func printFeed(line: UInt8) -> Data
static func requestBatteryStatus() -> Data
static func requestDeviceName() -> Data
static func requestDeviceVersion() -> Data
static func requestStatus() -> Data
static func resetLineSpace() -> Data
static func saveFeedLengthFromMark(_: UInt16) -> Data
static func selectTTF(_: String) -> Data
static func setBarcodeHeight(_: UInt8) -> Data
static func setBarcodeHri(_: Bool) -> Data
static func setBarcodeWidth(_: UInt8) -> Data
```

```

static func setBold(_: Bool) -> Data
static func setCharSpace(_: UInt8) -> Data
static func setCodeTable(_: Charset, mcu: McuType) -> Data
static func setFontSize(_: FontSize) -> Data
static func setLeftMargin(_: UInt16) -> Data
static func setLineSpace(_: UInt8) -> Data
static func setMsrMode1stTrack() -> Data
static func setMsrMode2ndTrack() -> Data
static func setMsrMode3rdTrack() -> Data
static func setMsrModeDoubleTrack() -> Data
static func setMsrModeTripleTrack() -> Data
static func setNonSecureScrMode() -> Data
static func setPageMode() -> Data
static func setPrintingWidth(_: UInt16) -> Data
static func setReverse(_: Bool) -> Data
static func setScrMode() -> Data
static func setUnderline(_: Bool) -> Data
static func setUpsideDown(_: Bool) -> Data
static func writeBarcode(_: String, type: BarcodeType) -> Data
static func writeBarcode(data: [UInt8], type: BarcodeType) -> Data
static func writeDataMatrix ( _: Data, height: UInt8, width: UInt8, size: UInt8) ->
Data
static func writeGS1Databar(_: String, type: GS1DatabarType, segment: UInt8) ->
Data
static func writeMaxicode(_: Data, mode: UInt8) -> Data
static func writeMicroPDF417(_: Data, column: UInt8, row: UInt8, ratio: UInt8) ->
Data
static func writePDF417(_: Data, column: UInt8, secureLevel: UInt8, ratio: UInt8)
-> Data
static func writeQRcode(_: Data, version: UInt8, ecLevel: UInt8, size: UInt8) ->
Data
static func writeTruncatedPDF417(_: Data, column: UInt8, secureLevel: UInt8,
ratio: UInt8) -> Data
static func writeTTF(_: String, width: UInt8, height: UInt8) -> Data

```

## 2.2. WSIImage Class

```

static func adjustImage(_: UIImage, brightness: Float, contrast: Float) ->
UIImage?

```

```
static func drawEllipse(x: UInt16, y: UInt16, radiusW: UInt16, radiusH: UInt16,
thick: UInt8) -> Data
static func drawImage(_: UIImage, x: UInt16, y: UInt16, dithering: Bool, compress:
Bool) -> Data
static func drawLine(x1: UInt16, y1: UInt16, x2: UInt16, y2: UInt16, thick: UInt8)
-> Data
static func drawRect(x: UInt16, y: UInt16, width: UInt16, height: UInt16, thick:
UInt8) -> Data
static func printImage(_: UIImage, dithering: Bool, compress: Bool) -> Data
static func printStoredImage(index: UInt8) -> Data
```

## 2.3. WSPacket Class

```
init(_: Data)
```



## 3. WSCmd Class

The WSCmd class provides most of the commands to operate and configure Woosim printers except for operations related to graphic and image printing.

### 3.1. enum

`public enum CutType: UInt8`

The cutter operation in a printer equipped with a cutter.

- `case full = 0`                      Cut the whole paper.
- `case partial = 1`                  Cut 90% of the paper.

`public enum FontSize: UInt8`

The size of the font to be used among the bitmap fonts included in the printer. The available font size is different depending on the selected character set.

- `case Large = 0`                      12 x 24 font. Some character sets use different size.
- `case Medium = 1`                   9 x 24 font. Some character sets use different size.
- `case Small = 2`                     8 x 16 font.

`public enum AlignType: UInt8`

Alignment of content to be printed.

- `case Left = 0`                       Left alignment.
- `case Center = 1`                   Center alignment.
- `case Right = 2`                    Right alignment.

`public enum BarcodeType: UInt8`

The type of barcode system.

- `case UPCA = 0x41`                    UPC-A barcode system
- `case UPCE = 0x42`                    UPC-E barcode system
- `case EAN13 = 0x43`                   EAN13 barcode system
- `case EAN8 = 0x44`                    EAN8 barcode system
- `case Code39 = 0x45`                  CODE39 barcode system
- `case ITF = 0x46`                     ITF barcode system
- `case Codabar = 0x47`                CODABAR barcode system
- `case Code93 = 0x48`                  CODE93 barcode system
- `case Code128 = 0x49`                CODE128 barcode system

`public enum GS1DatabarType: UInt8`

The type of GS1 Databar barcode system.

- `case Omnidirectional = 0`
- `case Truncated = 1`
- `case Stacked = 2`

```
case StackedOmnidirectional = 3
case Limited = 4
case Expanded = 5
case ExpandedStacked = 6
```

```
public enum PageDirection: UInt8
```

The printing direction in page mode.

```
case LeftToRight = 0      Print from left to right starting from the top left
case BottomToTop = 1      Print from bottom to top starting from the bottom left
case RightToLeft = 2      Print from right to left starting from the bottom right
case TopToBottom = 3      Print from top to bottom starting from the top right
```

```
public enum McuType: UInt8
```

The type of MCU.

```
case M16C                M16C MCU
case ARM                  ARM MCU
case RX                   RX MCU
```

```
public enum Charset: UInt8
```

Character set code table.

```
case CP437 = 0           USA, Standard Europe
case Katakana = 1        Katakana
case CP850 = 2           Multilingual (Latin 1)
case CP860 = 3           Portuguese
case CP863 = 4           Canadian French
case CP865 = 5           Nordic
case CP852 = 6           Slavic (Latin 2)
case CP857 = 7           Turkish
case CP737 = 8           Greek
case CP866 = 9           Russian (Cyrillic)
case CP852 = 10          Hebrew
case CP775 = 11          Baltic
case Polish = 12          Polish
case ISO8859-15 = 13     Latin 9
case Win1252 = 14        Latin1
case CP858 = 15          Multilingual Latin I + Euro
case CP855 = 16          Russian (Cyrillic)
case Win1251 = 17        Russian (Cyrillic)
case Win1250 = 18        Central Europe
case Win1253 = 19        Greek
case Win1254 = 20        Turkish
```

<code>case Win1255 = 21</code>	Hebrew
<code>case Win1258 = 22</code>	Vietnamese
<code>case Win1257 = 23</code>	Baltic
<code>case Azerbaijani = 24</code>	Azerbaijani
<code>case CP874 = 30</code>	Thai
<code>case CP8720 = 40</code>	Arabic
<code>case Win1256 = 41</code>	Arabic
<code>case Farsi = 42</code>	Farsi
<code>case ArabicPFB = 43</code>	Arabic presentation forms B
<code>case Hindi = 50</code>	Hindi Devanagari
<code>case DBCS = 0xFF</code>	Usually one of the Korean(EUC_KR), Japanese(Shift-JIS) or Chinese(Big5 or GB18030)

## 3.2. Method

### 3.2.1. Printer Operation

`public static func initPrinter() -> Data`

Initialize printer. Clear the data in the printer buffer and resets the printer configuration.

Returns

Data for printer control

`public static func printData() -> Data`

Print the data in the print buffer and feed one line based on the current line spacing. It sets the print position to the beginning of the line.

Returns

Data for printer control

`public static func printFeed(dot: UInt8) -> Data`

Print the data in the print buffer and feed specified dots.

Parameters

dot                      The feed length in dots (0~255)

Returns

Data for printer control

`public static func printFeed(line: UInt8) -> Data`

Print the data in the print buffer and feed specified lines.

Parameters

line                      The feed length in lines (0~255)

Returns

Data for printer control

```
public static func setPageMode() -> Data
```

Change mode from standard mode to page mode.

Returns

Data for printer control

```
public static func cutPaper(_ mode: CutType) -> Data
```

Select cut mode and cut paper. It works for auto-cutter installed printers.

Parameters

mode                      The cutting mode

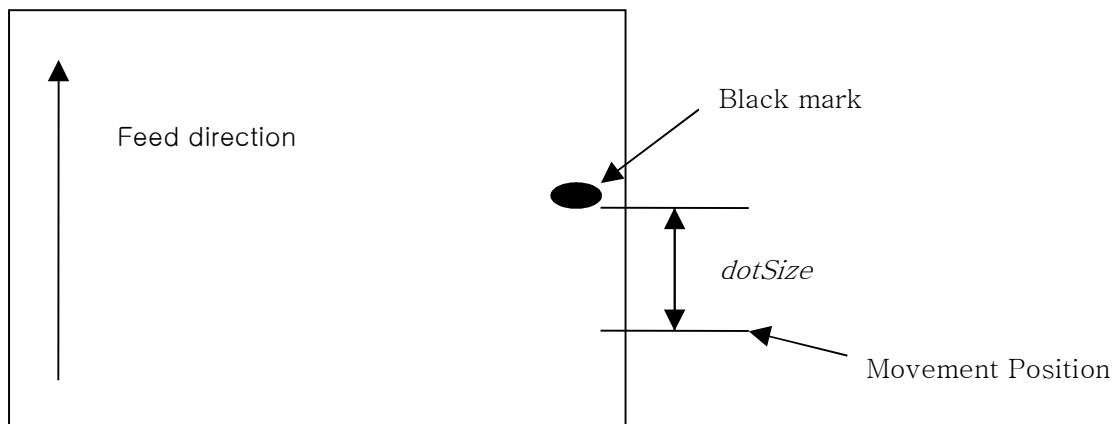
Returns

Data for printer control

```
public static func saveFeedLengthFromMark(_ dotSize: UInt16) -> Data
```

Save the movement position from the black mark to the printer flash memory area. It is strongly recommended that this method will be used in separated setting utility program because it does flash memory writing action.

After the movement position is set to the printer, paper will feed the amount of assigned distance from the black mark whenever call **feedToMark** method.



Parameters

dotSize                      The distance from the black mark in dots

Returns

Data for printer control

```
public static func feedToMark() -> Data
```

Feed the paper to the movement position after mark sensing position.

Returns

Data for printer control

```
public static func openCashDrawer() -> Data
```

Generate pulse to open cash drawer.

Returns

Data for printer control

`public static func requestStatus() -> Data`

Require status of the connected printer. The one-byte response data from device includes condition information by paper sensor, cover sensor, mark sensor and so on. See the <ESC v> command explanation in the "Woosim Command Manual" for details because the value varies with each printer model.

Returns

Data for printer control

`public static func requestBatteryStatus() -> Data`

Require battery status of the connected printer. The one-byte response data from device includes battery capacity level in high 4 bits and printer status in low 4 bits. The most significant bit of response byte is always 1.

Battery Voltage	High 4 bits
7.4 V	1 0 0 0
7.5 V	1 0 0 1
7.8 V	1 0 1 0
8.2 V	1 0 1 1

Returns

Data for printer control

`public static func requestDeviceName() -> Data`

Require device model name and MCU type of the connected printer. The response data from device is firmware dependent. For example, **R240(RX)**.

Returns

Data for printer control

`public static func requestDeviceVersion() -> Data`

Require device version and build date of the connected printer. The response data from device is firmware dependent. For example, **[Ver 2.0 2018/10/31]**.

Returns

Data for printer control

### 3.2.2. Text Attribute

`public static func setCodeTable(_ charset: Charset, mcu: McuType) -> Data`

Set character set code table.

Supported code tables are dependent on MCU of target printer. In case of M16C or ARM MCU, it only supports CP437, Katakana, CP850, CP860, ISO8859-15, Polish, and DBCS. DBCS means

double byte character set which is usually one of Korean, Chinese or Japanese. See the <ESC t> command explanation in the "Woosim Command Manual" for details.

Parameters

charset                      The character set code table

mcu                          The MCU type of printer

Returns

Data for printer control

```
public static func alignText(_ mode: AlignType) -> Data
```

Align bitmap font text data in one line. It is affect in the area between the current position and the end of printing area.

Parameters

mode                          The alignment type

Returns

Data for printer control

```
public static func alignContent (_ mode: AlignType) -> Data
```

Align data in one line including TTF and barcode as well as bitmap font text data. It is affect in the area between the current position and the end of printing area.

It can be worked with firmware released after Oct. 2015.

Parameters

mode                          The alignment type

Returns

Data for printer control

```
public static func setFontSize(_ size: FontSize) -> Data
```

Set font size. When the font size is changed, the text style is initialized.

Supported font sizes are dependent on MCU of target printer and character set code table. See the <ESC t> command explanation in the "Woosim Command Manual" for details.

Parameters

size                          The font size

Returns

Data for printer control

**Table for Supported Font Size by Charset**

Charset	FontSize
CP437 Katakana CP850 CP860 CP863 CP865 CP852 CP857 CP737 CP866 CP862 CP775 Polish ISO8859-15 Win1252 CP858 CP855 Win1251 Win1250 Win1253 Win1254 Win1255	Large : 12 x 24 Medium : 9 x 24 Small : 8 x 16

Win1258 Win1257 Azerbaijani	
CP874	Large : 12 x 24 Medium : N.A Small : N.A
CP720 Win1256 Farsi ArabicPFB Hindi	Large : 16 x 24 Medium : N.A Small : N.A
DBCS (EUC_KR)	Large : 24 x 24 Medium : 16 x 24 Small : N.A
DBCS (GB18030) DBCS (Big5) DBCS (Shift-JIS)	Large : 24 x 24 Medium : N.A Small : N.A

```
public static func extendFont(width: UInt8, height: UInt8) -> Data
```

Extend font size.

Parameters

width                      The horizontal extension (1 ~ 8)

height                     The vertical extension (1 ~ 8)

Returns

Data for printer control

```
public static func setBold(_ enable: Bool) -> Data
```

Turn on or off the bold style printing mode.

Parameters

enable                     If true, bold attribute is applied to the following text.

Returns

Data for printer control

```
public static func setUnderline(_ enable: Bool) -> Data
```

Turn on or off the underline style printing mode.

Parameters

enable                     If true, underline attribute is applied to the following text.

Returns

Data for printer control

```
public static func setReverse(_ enable: Bool) -> Data
```

Turn on or off the reverse style printing mode.

Parameters

enable                      If true, reverse attribute is applied to the following text.

Returns

Data for printer control

```
public static func setCharSpace(_ dotSize: UInt8) -> Data
```

Set right-side character spacing.

It can be set independently in standard mode and page mode.

Parameters

dotSize                      The character spacing in dots (0~255)

Returns

Data for printer control

```
public static func setLineSpace(_ dotSize: UInt8) -> Data
```

Set line spacing.

It can be set independently in standard mode and page mode.

Parameters

dotSize                      The line spacing in dots (0~255)

Returns

Data for printer control

```
public static func resetLineSpace() -> Data
```

Set line spacing to default value.

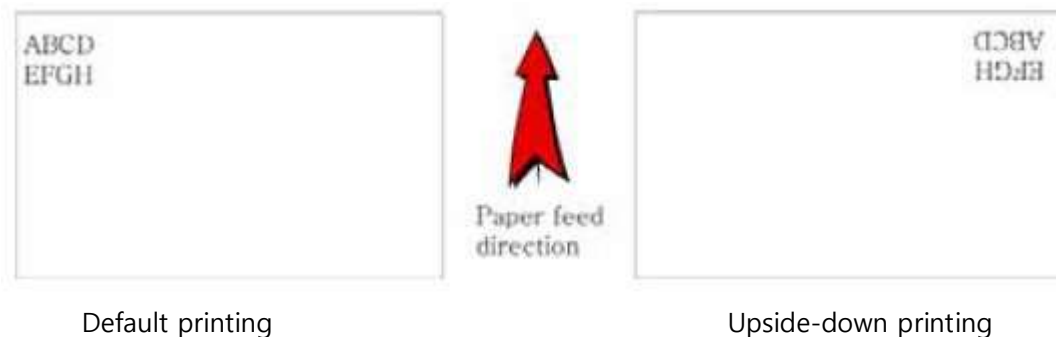
It can be set independently in standard mode and page mode.

Returns

Data for printer control

```
public static func setUpsideDown(_ enable: Bool) -> Data
```

Turn on or off upside down printing. It is enabled only when processed at the beginning of a line in standard mode.



Default printing

Upside-down printing

Parameters

enable                      If true, upside-down printing is applied to the following text.

Returns

Data for printer control



```
public static func moveAbsPosition(_ dotSize: UInt16) -> Data
```

Move printing position from the beginning of the line. If the position is out of printable area, the method is ignored.

Parameters

dotSize                      The distance from the beginning of the line in dots

Returns

Data for printer control

```
public static func moveRelPosition(_ dotSize: UInt16) -> Data
```

Move printing position from the current position. If the position is out of printable area, the method is ignored.

Parameters

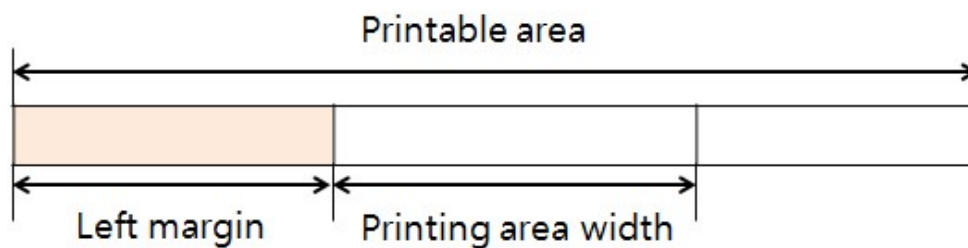
dotSize                      The distance from the current position in dots

Returns

Data for printer control

```
public static func setLeftMargin(_ dotSize: UInt16) -> Data
```

Set left margin. If the margin value exceeds the printable area, the method is ignored.



Parameters

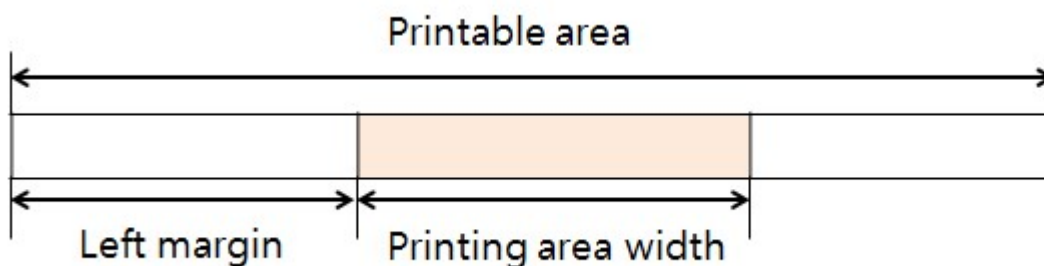
dotSize                      The left margin value from the beginning of the line in dots

Returns

Data for printer control

```
public static func setPrintingWidth(_ dotSize: UInt16) -> Data
```

Set printing area width after left margin. If the sum of left margin and printing area width exceeds the printable area, the method is ignored.



#### Parameters

dotSize                      The printable width in dots

#### Returns

Data for printer control

```
public static func selectTTF(_ filename: String) -> Data
```

Select TrueType font file name stored in the printer. The assigned TTF file should be saved in the printer in advance. The length of the name for TTF file should be less than 30 letters in English.

#### Parameters

filename                      The TTF file name

#### Returns

Data for printer control

```
public static func writeTTF(_ text: String, width: UInt8, height: UInt8) -> Data
```

Set TTF size and get Unicode value from the given string.

#### Parameters

text                          The string to be printed

width                          The TTF character width (4~255)

height                          The TTF character height (4~255)

#### Returns

Data for printer control and printing TTF string

### 3.2.3. Barcode

```
public static func setBarcodeWidth(_ width: UInt8) -> Data
```

Set the width of a barcode. If the parameter value is out of range, this method is ignored. It affects 1D barcode and PDF417.

#### Parameters

width                          The barcode width in dots (1 ~ 8)

#### Returns

Data for printer control

```
public static func setBarcodeHeight(_ height: UInt8) -> Data
```

Set the height of a barcode. If the parameter value is out of range, this method is ignored. It affects 1D barcode.

#### Parameters

width                          The barcode height in dots (1 ~ 255)

#### Returns

Data for printer control

```
public static func setBarcodeHri(_ enable: Bool) -> Data
```

Turn on or off the HRI characters print mode. It affects 1D barcode and PDF417.

Parameters

enable                      If true, HRI characters are printed with barcode.

Returns

Data for printer control

```
public static func writeBarcode(_ string: String, type: BarcodeType) -> Data
```

Write 1D barcode.

Data length and value is dependent on barcode type. See the <GS k> command explanation in the "Woosim Command Manual" for details.

Barcode System	Number of Characters	Data Source
UPC-A	$11 \leq n \leq 12$	$48 \leq d \leq 57$
UPC-E	$11 \leq n \leq 12$	$48 \leq d \leq 57$
EAN13	$12 \leq n \leq 13$	$48 \leq d \leq 57$
EAN8	$7 \leq n \leq 8$	$48 \leq d \leq 57$
CODE39	$1 \leq n \leq 255$	$48 \leq d \leq 57, 65 \leq d \leq 90,$ $d = 32, 36, 37, 43, 45, 46, 47$
ITF	$1 \leq n \leq 255$ (even number)	$48 \leq d \leq 57$
CODABAR	$1 \leq n \leq 255$	$48 \leq d \leq 57, 65 \leq d \leq 68,$ $d = 36, 43, 45, 46, 47, 58$
CODE93	$1 \leq n \leq 255$	$0 \leq d \leq 127$
CODE128	$2 \leq n \leq 255$	$0 \leq d \leq 127$ $d = C1H, C2H, C3H, C4H$

Parameters

string                      The barcode source data.

type                        The barcode type

Returns

Data for printer control and barcode data

```
public static func writeBarcode(data: [UInt8], type: BarcodeType) -> Data
```

Write 1D barcode.

Data length and value is dependent on barcode type. See the <GS k> command explanation in the "Woosim Command Manual" for details.

Parameters

data                        The barcode source data.

type                        The barcode type

Returns

Data for printer control and barcode data

```
public static func writeGS1Databar(_ string: String, type: GS1DatabarType,
segment: UInt8) -> Data
```

Write GS1 Databar.

It can be worked with RX MCU printer firmware released after Oct. 2012.

Type	Data	Segment
Omnidirectional	This field should be digits less than 14.	
Truncated		
Stacked		
Stacked Omnidirectional		
Limited		
Expanded	Use '[' and ']' instead of '(' and ')' Ex) "(01)90012345678908(3103)012233" → "[01]90012345678908[3103]012233"	2 ≤ s ≤ 20 (even number)
Expanded Stacked		

Parameters

string	The GS1 Databar source data. See the <GS 1> command explanation in the "Woosim Command Manual" for details.
type	The GS1 Databar type
segment	The segment per row that should be even number in range of 2~20. It is only valid for Expanded Stacked type.

Returns

Data for printer control and GS1 Databar data

```
public static func writePDF417(_ data: Data, column: UInt8, secureLevel: UInt8,
ratio: UInt8) -> Data
```

Write PDF417 barcode.

Parameters

data	The barcode source data.
column	The column number (1 ~ 30)
secureLevel	The security level to restore when barcode image is damaged (0 ~ 8)
ratio	The horizontal and vertical ratio (2 ~ 5)

Returns

Data for printer control and barcode data

```
public static func writeDataMatrix(_ data: Data, height: UInt8, width: UInt8,
size: UInt8) -> Data
```

Write Data Matrix barcode.

When height or width parameter is 0, the printer selects the barcode size automatically. See the <ESC Z> command explanation in the "Woosim Command Manual" for details.

## Parameters

data	The barcode source data.
height	The height of the symbol (0 : auto size)
width	The width of the symbol (0 : auto size)
size	The module size (1 ~ 8)

## Returns

Data for printer control and barcode data

Table for Data Matrix size

Symbol-size		Data capacity(bytes)			ECC(%)	Remark
Height	Width	Numeric	Alpha-numeric	Byte(8bit)		
8	18	10	6	5	58.3	rectangular
8	32	20	12	10	52.4	rectangular
10	10	6	3	3	62.5	
12	12	10	6	5	58.3	
12	26	32	21	16	46.7	rectangular
12	36	44	30	22	45.0	rectangular
14	14	16	9	8	55.6	
16	16	24	15	12	50.0	
16	36	34	45	32	42.9	rectangular
16	48	98	72	49	36.4	rectangular
18	18	36	24	18	43.8	
20	20	44	30	22	45.0	
22	22	60	42	30	40.0	
24	24	72	51	36	40.0	
26	26	88	63	44	38.9	
32	32	124	90	62	36.7	
36	36	172	126	86	32.8	
40	40	228	168	114	29.6	
44	44	288	213	144	28.0	
48	48	348	258	174	28.1	
52	52	408	303	204	29.2	
64	64	560	417	280	28.6	
72	72	736	549	368	28.1	
80	80	912	681	456	29.6	
88	88	1152	861	576	28.0	
96	96	1392	1041	696	28.1	
104	104	1632	1221	816	29.2	
120	120	2100	1572	1050	28.0	

132	132	2608	1953	1304	27.6	
144	144	3116	2334	1558	28.5	

Used only square type for auto sized symbol.

```
public static func writeQRcode(_ data: Data, version: UInt8, ecLevel: String,
size: UInt8) -> Data
```

Write QR Code.

When the version parameter is 0, the printer selects the barcode size automatically. See the <ESC Z> command explanation in the "Woosim Command Manual" for details.

Parameters

data	The barcode source data.
version	The version of the symbol (0 ~ 40, 0 : auto size)
ecLevel	The EC level (L : 7%, M : 15%, Q : 25%, H : 30%)
size	The module size (1 ~ 8)

Returns

Data for printer control and barcode data

**Table for QR Code size(version)**

Version	Capacity (Code words) by EC level			
	L ( 7% )	M ( 15% )	Q ( 25% )	H ( 30% )
1	19	16	13	9
2	34	28	22	16
3	55	44	34	26
4	80	64	48	36
5	108	86	62	46
6	136	108	76	60
7	156	124	88	66
8	194	154	110	86
9	232	182	132	100
10	274	216	154	122
11	324	254	180	140
12	370	290	206	158
13	428	334	244	180
14	461	365	261	197
15	523	415	295	223
16	589	453	325	253
17	647	507	367	283
18	721	563	397	313
19	795	627	445	341
20	861	669	485	385

21	932	714	512	406
22	1006	782	568	442
23	1094	860	614	464
24	1174	914	664	514
25	1276	1000	718	538
26	1370	1062	754	596
27	1468	1128	808	628
28	1531	1193	871	661
29	1631	1267	911	701
30	1735	1373	985	745
31	1843	1455	1033	793
32	1955	1541	1115	845
33	2071	1631	1171	901
34	2191	1725	1231	961
35	2306	1812	1286	986
36	2434	1914	1354	1054
37	2566	1992	1426	1096
38	2702	2102	1502	1142
39	2812	2216	1582	1222
40	2956	2334	1666	1276

```
public static func writeMicroPDF417(_ data: Data, column: UInt8, row: UInt8,
ratio: UInt8) -> Data
```

Write Micro PDF417 barcode.

Parameters

data	The barcode source data.
column	The column number of the barcode (1 ~ 4)
row	The row number of the barcode (4 ~ 44, 0 : auto size)
ratio	The horizontal and vertical ratio (2 ~ 5)

Returns

Data for printer control and barcode data

**Table for Micro PDF417 Data Size**

Number of Columns	Number of Rows	Max Data Bytes	Max Alpha Characters	Max Digits
1	11	3	6	8
1	14	7	12	17
1	17	10	18	26
1	20	13	22	32

1	24	18	30	44
1	28	22	38	55
2	8	8	14	20
2	11	14	24	35
2	14	21	36	52
2	17	27	46	67
2	40	33	56	82
2	46	38	64	93
2	52	43	72	105
3	6	6	10	14
3	8	10	18	26
3	10	15	26	38
3	12	20	34	49
3	15	27	46	67
3	20	39	66	96
3	26	54	90	132
3	32	68	114	167
3	38	82	138	202
3	44	97	162	237
4	4	8	14	20
4	6	13	22	32
4	8	20	34	49
4	10	27	46	67
4	12	34	58	85
4	15	45	76	111
4	20	63	106	155
4	26	85	142	208
4	32	106	178	261
4	38	128	214	313
4	44	150	250	366

```
public static func writeTruncatedPDF417(_ data: Data, column: UInt8, secureLevel:
UInt8, ratio: UInt8) -> Data
```

Write Truncated PDF417 barcode.

Parameters

data	The barcode source data.
column	The column number (1 ~ 4)
secureLevel	The security level to restore when barcode image is damaged (0 ~ 8)



ratio                      The horizontal and vertical ratio (2 ~ 5)

Returns

Data for printer control and barcode data

```
public static func writeMaxicode(_ data: Data, mode: UInt8) -> Data
```

Write Maxicode.

It can be worked with RX MCU printer firmware released after Aug. 2012.

When *mode* is 2 or 3, first 15 bytes of the data is primary data. The primary data structure is as follow:

- Post/Zip code (9 bytes)

- mode 2                      5-digit zip code + 4-digit code extension.

- If code extension does not exist, "0000" must be specified.

- mode 3                      6 alpha-numeric bytes + 3 bytes filler (ex. Spaces)

- Country code (3 digit) from ISO 3166

- Class of service (3 digit)

Parameters

data                      The barcode source data.

mode                      The mode of the Maxicode (2 ~ 6)

Returns

Data for printer control and barcode data

### 3.2.4. Page Mode

```
public static func pageModeToStdMode() -> Data
```

Change mode from page mode to standard mode.

Returns

Data for printer control

```
public static func pageModePrintData() -> Data
```

Print data in page mode.

Returns

Data for printer control

```
public static func pageModePrintStdMode() -> Data
```

Print data in page mode, and change mode from page mode to standard mode.

Returns

Data for printer control

```
public static func pageModeDeleteData() -> Data
```

Delete all printable data in page mode.

Returns

Data for printer control

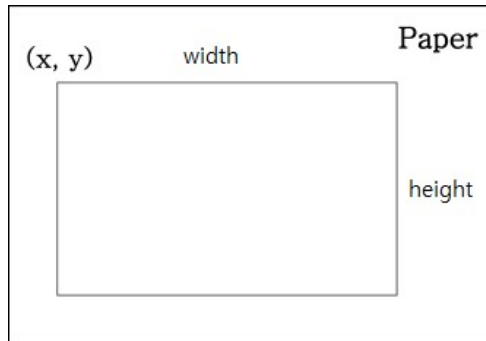
```
public static func pageModeSetArea(x: UInt16, y: UInt16, width: UInt16, height: UInt16) -> Data
```

Set write area in page mode.

If the horizontal or vertical starting position is set outside the printable area, this method is ignored.

If ( $x + width > \text{printable area}$ ), then the *width* value will be set to ( $\text{printable area} - x$ ).

If ( $y + height > \text{printable area}$ ), then the *height* value will be set to ( $\text{printable area} - y$ ).



#### Parameters

- x** The horizontal starting position of printing area in dot unit.  
The device default value is 0.
- y** The vertical starting position of printing area in dot unit  
The device default value is 0.
- width** The printing area width in dot unit. It should be bigger than 0.  
The maximum horizontal printable area is according to the printing width  
(1 inch: 192, 2 inch: 384, 3 inch: 576, 4 inch: 832)  
The device default value is same as maximum printable area value.
- height** The printing area height in dot unit. It should be bigger than 0.  
The maximum vertical printable area is 2400.  
The device default value is same as maximum printable area value.

#### Returns

Data for printer control

```
public static func pageModeSetDirection(_: PageDirection) -> Data
```

Set the printing direction and start position in page mode. It works in the printable area set by **pageModeSetArea** method.

direction	starting position	printing direction	
LeftToRight	upper left (A)	left → right	
BottomToTop	lower left (B)	bottom → up	
RightToLeft	lower right (C)	right → left	
TopToBottom	upper right (D)	top → bottom	

## Parameters

direction                      The symbolic value to specify printing direction and start position

## Returns

Data for printer control

```
public static func pageModeSetPosition(x: UInt16, y: UInt16) -> Data
```

Set position in page mode.

## Parameters

x                                The horizontal starting position of printing area in dot unit.

y                                The vertical starting position of printing area in dot unit

## Returns

Data for printer control

```
public static func pageModeMoveAbsVertical(_ dotSize: UInt16) -> Data
```

Move vertical printing position from the start position in page mode. If moved position exceeds specified printing area, this method is ignored. This method operates depending on the printing direction set by **pageModeSetDirection** method.

## Parameters

dotSize                        The vertical distance from the start position in dots

## Returns

Data for printer control

```
public static func pageModeMoveRelVertical(_ dotSize: UInt16) -> Data
```

Move vertical printing position from the current position in page mode. If moved position exceeds specified printing area, this method is ignored. This method operates depending on the printing direction set by **pageModeSetDirection** method.

## Parameters

dotSize                        The vertical distance from the current position in dots

## Returns

Data for printer control

### 3.2.5. Card Operation

```
public static func setMsrMode1stTrack() -> Data
```

Enter MSR 1st track reading mode. The 1st track means track 2 for 23 track MSR or track 1 for other types of MSR.

The device waits for MS card reading in MSR reading mode. After successful card swiping, device sends the card data to host and exits MSR reading mode.

## Returns

Data for printer control

```
public static func setMsrMode2ndTrack() -> Data
```

Enter MSR 2nd track reading mode. The 2nd track means track 3 for 23 track MSR or track 2 for other types of MSR.

The device waits for MS card reading in MSR reading mode. After successful card swiping, device sends the card data to host and exits MSR reading mode.

Returns

Data for printer control

```
public static func setMsrMode3rdTrack() -> Data
```

Enter MSR 3rd track reading mode. The 3rd track means track 3 for 123 track MSR. It is working for 123 track MSR only.

The device waits for MS card reading in MSR reading mode. After successful card swiping, device sends the card data to host and exits MSR reading mode.

Returns

Data for printer control

```
public static func setMsrModeDoubleTrack() -> Data
```

Enter MSR double track reading mode. The double track means track 2 and 3 for 23 track MSR or track 1 and 2 for other types of MSR.

The device waits for MS card reading in MSR reading mode. After successful card swiping, device sends the card data to host and exits MSR reading mode.

Returns

Data for printer control

```
public static func setMsrModeTripleTrack() -> Data
```

Enter MSR triple track reading mode. The triple track means all tracks for 123 track MSR. It is working for 123 track MSR only.

The device waits for MS card reading in MSR reading mode. After successful card swiping, device sends the card data to host and exits MSR reading mode.

Returns

Data for printer control

```
public static func exitMsrMode() -> Data
```

Exit MSR reading mode.

Returns

Data for printer control

```
public static func setScrMode() -> Data
```

Enter SCR mode. After enter SCR mode, application should control directly the SCR module.

Returns

Data for printer control

```
public static func exitScrMode() -> Data
```

Exit SCR mode.

Returns

Data for printer control

```
public static func setNonSecureScrMode() -> Data
```

Enter non-secure SCR reading mode to get card data.

The device waits for smart card reading in SCR reading mode. After successful card reading, device sends the card data to host and exits SCR reading mode.

Returns

Data for printer control

```
public static func exitNonSecureScrMode() -> Data
```

Exit non-secure SCR reading mode.

Returns

Data for printer control

## 4. WSImage Class

The WSImage class provides commands to draw graphics and images in page mode and to print images in standard mode

### 4.1. Method

```
public static func drawLine(x1: UInt16, y1: UInt16, x2: UInt16, y2: UInt16, thick: UInt8) -> Data
```

Draw a line on specified position in page mode.

Parameters

x1	The x-coordinate of start position
y1	The y-coordinate of start position
x2	The x-coordinate of end position
y2	The y-coordinate of end position
thick	The thickness of line ( > 0)

Returns

Data for printer control to draw line

```
public static func drawRect(x: UInt16, y: UInt16, width: UInt16, height: UInt16, thick: UInt8) -> Data
```

Draw a rectangle on specified position in page mode.



Parameters

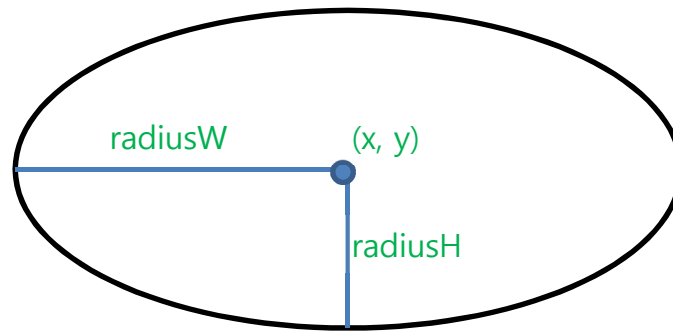
x	The x-coordinate of upper-left corner of box
y	The y-coordinate of upper-left corner of box
width	The box width in dots
height	The box height in dots
thick	The thickness of line ( > 0)

Returns

Data for printer control to draw rectangle

```
public static func drawEllipse(x: UInt16, y: UInt16, radiusW: UInt16, radiusH: UInt16, thick: UInt8) -> Data
```

Draw an ellipse on specified position in page mode.



#### Parameters

x	The x-coordinate of central point
y	The y-coordinate of central point
radiusW	The radius of horizontal axis
radiusH	The radius of vertical axis
thick	The thickness of line ( > 0)

#### Returns

Data for printer control to draw ellipse

```
public static func drawImage(_ image: UIImage, x: UInt16, y: UInt16, dithering: Bool, compress: Bool) -> Data
```

Draw an image on specified position in page mode.

#### Parameters

image	The image object to be drawn
x	The x-coordinate of upper-left corner of box
y	The y-coordinate of upper-left corner of box
dithering	If true, the image will be drawn with dithering technique
compress	If true, the image data will be delivered in compressed form

#### Returns

Data for printer control to draw image

```
public static func printImage(_ image: UIImage, dithering: Bool, compress: Bool) -> Data
```

Print an image in standard mode.

#### Parameters

image	The image object to be drawn
dithering	If true, the image will be drawn with dithering technique
compress	If true, the image data will be delivered in compressed form

#### Returns

Data for printer control to print image

```
public static func printStoredImage(index: UInt8) -> Data
```

Print an image that was downloaded in the printer.

The maximum count of images that can be stored in device is dependent on MCU type. See the <ESC f> command explanation in the "Woosim Command Manual" for details.

Parameters

index	The index of image stored in the printer
-------	--

Returns

Data for printer control to print image

```
public static func adjustImage(_ image: UIImage, brightness: Float, contrast: Float) -> UIImage?
```

Adjust image brightness and contrast.

Parameters

image	The original image object
brightness	The modified brightness value (-1.0 ~ 1.0, default 0.0)
contrast	The modified contrast value (0 ~ 10, default 1.0)

Returns

UIImage with specified brightness and contrast values applied or nil if image modification fails



## 5. WSPacket Class

The WSPacket class is used to interpret the data received from the printer.

When a WSPacket class instance is created with the received data, the result of analyzing the data is saved in the **packetType** property and **packetData** property.

### 5.1. enum

```
public enum PacketType: UInt
```

The type of data received from the printer.

case BINARY	Meaningless or impossible to interpret
case STRING	The received data or analysis result is a string
case MSR	The MSR track data

### 5.2. property

```
var packetType: PacketType { get }
```

The type of data received from the printer.

```
var packetData: Any { get }
```

The result of interpret of data received from the printer.

### 5.3. Method

```
init(_ rcvData: Data)
```

Create a WSPacket class instance and interpret received data. The value and type stored in each property according to the analysis result are as follows.

packetType	packetData Type	packetData Contents
BINARY	String	Convert received data to hex byte string
STRING	String	Convert received data to string or save the result of analyzing the data as a string
MSR	[Data, Data, Data]	The MSR data of each track PacketData[0] = MSR track 1 data PacketData[1] = MSR track 2 data PacketData[2] = MSR track 3 data

Parameters

rcvData	The data received from the printer
---------	------------------------------------

## 6. Samples

---

Woosim iOS SDK includes several sample applications to provide useful help.

Each sample has the same function and is made with Swift and Objective-C respectively.

### 6.1. BTPrint

It is a traditional sample application including following functions:

- Text printing
- Image printing
- Various 1D and 2D barcode printing
- Enter MSR mode and show the magnetic card data by card swiping
- Get device information and status

To connect to a Woosim printer, touch the right bar button of the title bar labeled "Printer". If there is no device connected via Bluetooth, you can select a menu item to move to Settings to connect one.

### 6.2. WiFiPrint

It has similar functions as the BTPrint, but Wi-Fi connection is used instead of Bluetooth.

The printer and host device must be connected to the same wireless router. If you want to change configuration of the Woosim wireless printer, Woosim WLAN Manager Utility is needed. You can download it from Woosim homepage.

To setup Wi-Fi connection, user should know IP address and port number of the target printer. Most of Woosim printers use 9100 as port number and some old ones use 1470. You can check this information through Self-Test function of the printer.