



# SDK for iOS API Reference Guide

# Label Printer

---

**Rev.1.04**

**SLP-DX420 / DX423**  
**SLP-DX220 / DX223**  
**SLP-TX420 / TX423**  
**SLP-TX220 / TX223**  
**SLP-TX400 / TX403**  
**SLP-DL410 / DL413**

## ■ Table of Contents

<b>1. About This Manual .....</b>	<b>4</b>
1-1 Supported iOS Version .....	4
1-2 List of Supported Printer / Interface .....	4
1-3 List of Supported Methods .....	5
<b>2. Constants .....</b>	<b>9</b>
2-1 Result Codes .....	9
2-2 Alignment .....	10
2-3 Barcode HRI .....	11
2-4 MaxiCode Modes .....	12
2-5 1D Barcode Types .....	13
2-6 Barcode Origin Point .....	14
2-7 Error Correction Level .....	14
2-8 Data Compression Method .....	15
2-9 QRCode Model .....	15
2-10 Code 49 Starting Mode .....	16
2-11 Codablock Mode .....	17
2-12 Check Digit Option .....	17
2-13 RSS Barcode Type .....	18
2-14 Rotation Degrees .....	19
2-15 Device Fonts .....	20
2-16 Vector Fonts .....	21
2-17 Draw Block Options .....	21
2-18 Draw Circle Sizes .....	22
2-19 International Character Set .....	23
2-20 Code Pages .....	24
<b>3. LabelPrinterSDK Class Reference .....</b>	<b>26</b>
3-1 Overview .....	26
3-2 Methods .....	26
3-2-1 open .....	26
3-2-2 close .....	26
3-2-3 connectWithAddress .....	27
3-2-4 isConnected .....	27
3-2-5 disconnect .....	27
3-2-6 disconnectWithTimeout .....	28
3-2-7 getDeviceFontList .....	28
3-2-8 getVectorFontList .....	28
3-2-9 getDrawBlockOptions .....	29
3-2-10 getDrawCircleSizes .....	29
3-2-11 getBarcodeType1D .....	29
3-2-12 getBarcodeHRI .....	30
3-2-13 doPrint .....	30
3-2-14 drawTextDeviceFont .....	31
3-2-15 drawTextVectorFont .....	33
3-2-16 drawBarcode1D .....	35
3-2-17 drawBarcodeMaxiCode .....	36
3-2-18 drawBarcodePDF417 .....	37
3-2-19 drawBarcodeQRCode .....	39
3-2-20 drawBarcodeDataMatrix .....	40

3-2-21 drawBarcodeAztec .....	41
3-2-22 drawBarcodeCode49 .....	43
3-2-23 drawBarcodeCodaBlock .....	44
3-2-24 drawBarcodeMicroPDF.....	45
3-2-25 drawBarcodeIMB .....	46
3-2-26 drawBarcodeMSI .....	47
3-2-27 drawBarcodePlessey .....	48
3-2-28 drawBarcodeTLC39.....	49
3-2-29 drawBarcodeRSS.....	50
<b>4. Appendix .....</b>	<b>51</b>
4-1 Recommended Flow of Operation .....	51
4-1-1 General .....	51
4-1-2 Printing Contents .....	52
4-1-3 Printing Two or More Contents on One Side .....	53
4-2 Sample Codes .....	54
4-2-1 Connecting/Disconnecting .....	54
4-2-2 Printing Contents in the Buffer .....	54
4-2-3 Printing a Rectangular Box to the Image Buffer .....	54
4-2-4 Printing a Circle to the Image Buffer .....	55
4-2-5 Printing a Text String with Device Font to the Image Buffer .....	55
4-2-6 Printing a Text String with Vector Font to the Image Buffer.....	55
4-2-7 Printing 1D Barcode to the Image Buffer .....	56
4-2-8 Printing PDF417 Barcode to the Image Buffer .....	56
4-2-9 Printing QRCode Barcode to the Image Buffer .....	56
4-2-10 Printing DataMatrix Barcode to the Image Buffer .....	57
4-2-11 Printing MaxiCode Barcode to the Image Buffer .....	57
4-2-12 Printing Aztec Barcode to the Image Buffer.....	57
4-2-13 Printing Code49 Barcode to the Image Buffer .....	58
4-2-14 Printing Codablock Barcode to the Image Buffer .....	58
4-2-15 Printing MicroPDF417 Barcode to the Image Buffer .....	58
4-2-16 Printing IMB Barcode to the Image Buffer .....	59
4-2-17 Printing MSI Barcode to the Image Buffer .....	59
4-2-18 Printing Plessey Barcode to the Image Buffer .....	59
4-2-19 Printing TLC39 Barcode to the Image Buffer .....	60
4-2-20 Printing RSS Barcode to the Image Buffer .....	60
4-3 Sample Application .....	61
4-3-1 Connect / Disconnect.....	61
4-3-2 Print to Device Font.....	61
4-3-3 Print Vector Font.....	62
4-3-4 Print Image.....	62
4-3-5 Use to more features. ....	63
4-3-6 Print Rectangle.....	63
4-3-7 Print Circle shapes. ....	64

# 1. About This Manual

This SDK Manual contains the description of the library API that is required for the development of iOS application programs.

BIXOLON makes continuous improvements to functions and qualities of products, and specifications and contents of this manual are subject to change without prior notice for this reason.

## 1-1 Supported iOS Version

- iOS 6.0 and later.

## 1-2 List of Supported Printer / Interface

Method/Property	Ethernet	Wifi	Bluetooth
SLP-DX420	O	O	X
SLP-DX423	O	O	X
SLP-DX220	O	O	O
SLP-DX223	O	O	O
SLP-TX420	O	O	X
SLP-TX423	O	O	X
SLP-TX220	O	O	X
SLP-TX223	O	O	X
SLP-DL410	O	O	X
SLP-DL413	O	O	X
SLP-TX400	O	O	X
SLP-TX403	O	O	X

## 1-3 List of Supported Methods

Method		SLP-DX420 /DX423/DX220/DX223/TX420/TX423/ TX220/TX223/DL410/DL413/TX400/TX403
General	<i>open</i>	○
	<i>close</i>	○
Connection	<i>connectWithAddress:port:</i>	○
	<i>isConnected</i>	○
	<i>disconnect</i>	○
	<i>disconnectWithTimeout:</i>	○
Draw Text	<i>drawTextDeviceFont:</i> xPosition: yPosition: fontSelection: fontWidth: fontHeight: rightSideCharacterSpacing: fontRotation: reverse: bold: textAlignment:	○
	<i>drawTextVectorFont:</i> xPosition: yPosition: fontSelection: fontWidth: fontHeight: rightSideCharacterSpacing: fontRotation: reverse: bold: italic: textAlignment:	○

Method	SLP-DX420 /DX423/DX220/DX223/TX420/TX423/
--------	--

		TX220/TX223/DL410/DL413/TX400/TX403
Draw Barcode	<i>drawBarcode1D:</i> xPosition: yPosition: barcodeType: widthNarrow: widthWide: height: hri: quietZoneWidth: rotation:	O
	<i>drawBarcodeMaxiCode:</i> xPosition: yPosition: mode:	O
	<i>drawBarcodePDF417:</i> xPosition: yPosition: maximumRowCount: maximumColumnCount: errorCorrectionLevel: dataCompressionMethod: printBarcodeText: barcodeOriginPoint: moduleWidth: barHeight: rotation:	O
	<i>drawBarcodeQRCode:</i> xPosition: yPosition: barcodeSize: model: errorColectionLevel: rotation:	O
	<i>drawBarcodeDataMatrix:</i> xPosition: yPosition: barcodeSize: reverse: rotation:	O
	<i>drawBarcodeAztec:</i> xPosition: yPosition: barcodeSize: extendedChannel: errorCorrectionLevel: menuSymbol: numberOfSymbols: optionalID: rotation:	O
<b>Method</b>		<b>SLP-DX420</b>

		<b>/DX423/DX220/DX223/TX420/TX423/ TX220/TX223/DL410/DL413/TX400/TX403</b>
<b>Draw Barcode</b>	drawBarcodeCode49: xPosition: yPosition: widthNarrow: widthWide: height: hri: startingMode: rotation:	○
	drawBarcodeCodaBlock: xPosition: yPosition: widthNarrow: widthWide: height: securityLevel: numberOfCharactersPerrow: mode: numberOfRowToEncode:	○
	drawBarcodeMicroPDF: xPosition: yPosition: moduleWidth: barcodeHeight: mode: rotation:	○
	drawBarcodeIMB: xPosition: yPosition: printBarcodeText: rotation:	○
	drawBarcodeMSI: xPosition: yPosition: widthNarrow: widthWide: height: checkDigitSelection: printCheckDigitInHRI: hri: rotation:	○
	drawBarcodePlessey: xPosition: yPosition: widthNarrow: widthWide: height: printCheckDigit: hri: rotation:	○

<b>Method</b>		<b>SLP-DX420 /DX423/DX220/DX223/TX420/DX423/ TX220/TX223/DL410/DL413/TX400/TX403</b>
Draw Barcode	drawBarcodeTLC39: xPosition: yPosition: widthNarrow: widthWide: height: rowHeightOfMicroPDF417: narrowWidthOfMicroPDF417: rotation:	○
	drawBarcodeRSS: xPosition: yPosition: barcodeType: magnification: separatorHeight: barcodeHeight: segmentWidth: rotation:	○
drawBlock	drawBlock: startPosY: endPosX: endPosY: option: thickness:	○
drawCircle	drawCircle: startPosY: sizeSelection: multiplier:	○
drawImage	drawImage: startPosX: startPosY: width:	○



## 2. Constants

Constants used in the provided SDK are defined in the “LabelPrinterSDK\_Defines.h” file.

### 2-1 Result Codes

Results Codes are the values returned from various methods, which indicate error conditions of the corresponding method.

```
typedef enum{
    _SDK_RESULT_SUCCESS                = 0x0000,
    _SDK_RESULT_FAIL                    = 0xF000,
    _SDK_RESULT_FAIL_NOT_SUPPORT_FUNCTION,
    _SDK_RESULT_FAIL_NO_OPEN           ,
    _SDK_RESULT_FAIL_OPEN_ALREADY      ,
    _SDK_RESULT_FAIL_NO_CONNECT        ,
    _SDK_RESULT_FAIL_CONNECT_ALREADY   ,
    _SDK_RESULT_FAIL_WRITE_ERROR       ,
    _SDK_RESULT_FAIL_READ_ERROR        ,
    _SDK_RESULT_FAIL_INVALID_PARAMETER ,
}__SDK_RESULT_CODES_;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_RESULT_SUCCESS	0X0000	Success
_SDK_RESULT_FAIL	0XF000	Fail
_SDK_RESULT_FAIL_NOT_SUPPORT_FUNCTION	0XF001	Not supported
_SDK_RESULT_FAIL_NO_OPEN	0XF002	SDK is not open
_SDK_RESULT_FAIL_OPEN_ALREADY	0XF003	SDK is already open
_SDK_RESULT_FAIL_NO_CONNECT	0XF004	Printer is not connected
_SDK_RESULT_FAIL_CONNECT_ALREADY	0XF005	Printer is already connected
_SDK_RESULT_FAIL_WRITE_ERROR	0XF006	Failed to send data
_SDK_RESULT_FAIL_READ_ERROR	0XF007	Failed to receive data
_SDK_RESULT_FAIL_INVALID_PARAMETER	0XF008	Invalid parameter

## 2-2 Alignment

Alignment constants are used to specify the alignment property.

```
typedef enum{
    _SDK_ALIGNMENT_LEFT           = 0,
    _SDK_ALIGNMENT_RIGHT         = 1,
    _SDK_ALIGNMENT_CENTER        = 2,
    _SDK_ALIGNMENT_STRING_FROM_RIGHT_2_LEFT = 2,
}_SDK_ALIGNMENTS;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_ALIGNMENT_LEFT	0	Align to left
_SDK_ALIGNMENT_RIGHT	1	Align to right
_SDK_ALIGNMENT_CENTER	2	Align to center
_SDK_ALIGNMENT_STRING_FROM_RIGHT_2_LEFT	2	Print characters from right to left

### **[Discussion]**

\_SDK\_ALIGNMENT\_CENTER value is valid for Vector Font.

If this value is applied to the printing of Device Font, it has the same effect as

\_SDK\_ALIGNMENT\_STRING\_FROM\_RIGHT\_2\_LEFT, where actual integer values assigned to two constants are same.

## 2-3 Barcode HRI

Barcode HRI (Human Readable Interpretation) constant is used to specify the position and font of HRI when printing barcodes that support HRI.

```
typedef enum{
    _SDK_BARCODE_HRI_NONE                = 0,
    _SDK_BARCODE_HRI_BELOW_FONTSIZE1    = 1,
    _SDK_BARCODE_HRI_ABOVE_FONTSIZE1    = 2,
    _SDK_BARCODE_HRI_BELOW_FONTSIZE2    = 3,
    _SDK_BARCODE_HRI_ABOVE_FONTSIZE2    = 4,
    _SDK_BARCODE_HRI_BELOW_FONTSIZE3    = 5,
    _SDK_BARCODE_HRI_ABOVE_FONTSIZE3    = 6,
    _SDK_BARCODE_HRI_BELOW_FONTSIZE4    = 7,
    _SDK_BARCODE_HRI_ABOVE_FONTSIZE4    = 8,
}__SDK_BARCODE_HRI;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_BARCODE_HRI_NONE	0	HRI is not used.
_SDK_BARCODE_HRI_BELOW_FONTSIZE1	1	Position of HRI: Below barcode Font Size: 1
_SDK_BARCODE_HRI_ABOVE_FONTSIZE1	2	Position of HRI: Above barcode Font Size: 1
_SDK_BARCODE_HRI_BELOW_FONTSIZE2	3	Position of HRI: Below barcode Font Size: 2
_SDK_BARCODE_HRI_ABOVE_FONTSIZE2	4	Position of HRI: Above barcode Font Size: 2
_SDK_BARCODE_HRI_BELOW_FONTSIZE3	5	Position of HRI: Below barcode Font Size: 3
_SDK_BARCODE_HRI_ABOVE_FONTSIZE3	6	Position of HRI: Above barcode Font Size: 3
_SDK_BARCODE_HRI_BELOW_FONTSIZE4	7	Position of HRI: Below barcode Font Size: 4
_SDK_BARCODE_HRI_ABOVE_FONTSIZE4	8	Position of HRI: Above barcode Font Size: 4

### **[Discussion]**

When a parameter is set to one of these constants, it may be replaced by a valid one and the changed one will be applied accordingly to actual operation.

## **2-4 MaxiCode Modes**

MaxiCode Mode constants are used to set the barcode option when printing Mexi code barcode.

```
typedef enum{  
    _SDK_MAXICODE_MODE_0           = 0,  
    _SDK_MAXICODE_MODE_2           = 2,  
    _SDK_MAXICODE_MODE_3           = 3,  
    _SDK_MAXICODE_MODE_4           = 4,  
}_SDK_MAXICODE_MODE;
```

Description of each constant is as follows.

<b>Code</b>	<b>Value</b>	<b>Description</b>
_SDK_MAXICODE_MODE_0	0	MaxiCode Mode 0
_SDK_MAXICODE_MODE_2	2	MaxiCode Mode 2
_SDK_MAXICODE_MODE_3	3	MaxiCode Mode 3
_SDK_MAXICODE_MODE_4	4	MaxiCode Mode 4

## 2-5 1D Barcode Types

Barcode Types constants are used to set the barcode option when printing 1D barcode.

```
typedef enum{
    _SDK_BARCODE_TYPE_CODE39           = 0,
    _SDK_BARCODE_TYPE_CODE128         = 1,
    _SDK_BARCODE_TYPE_I2Of5           = 2,
    _SDK_BARCODE_TYPE_CODABAR         = 3,
    _SDK_BARCODE_TYPE_CODE93          = 4,
    _SDK_BARCODE_TYPE_UPC_A           = 5,
    _SDK_BARCODE_TYPE_UPC_E           = 6,
    _SDK_BARCODE_TYPE_EAN13           = 7,
    _SDK_BARCODE_TYPE_EAN8            = 8,
    _SDK_BARCODE_TYPE_EAN128          = 9,
    _SDK_BARCODE_TYPE_CODE11          = 10,
    _SDK_BARCODE_TYPE_PLANET          = 11,
    _SDK_BARCODE_TYPE_INDUSTRIAL_2Of5 = 12,
    _SDK_BARCODE_TYPE_STANDARD_2Of5  = 13,
    _SDK_BARCODE_TYPE_LOGMARS         = 14,
    _SDK_BARCODE_TYPE_UPC_EAN_EXTENSIONS = 15,
    _SDK_BARCODE_TYPE_POSTNET         = 16,
}_SDK_BARCODE_TYPE;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_BARCODE_TYPE_CODE39	0	Code39
_SDK_BARCODE_TYPE_CODE128	1	Code128
_SDK_BARCODE_TYPE_I2Of5	2	Interleaved 2of5
_SDK_BARCODE_TYPE_CODABAR	3	Codabar
_SDK_BARCODE_TYPE_CODE93	4	Code93
_SDK_BARCODE_TYPE_UPC_A	5	UPC-A
_SDK_BARCODE_TYPE_UPC_E	6	UPC-E
_SDK_BARCODE_TYPE_EAN13	7	EAN13
_SDK_BARCODE_TYPE_EAN8	8	EAN8
_SDK_BARCODE_TYPE_EAN128	9	UCC/EAN128
_SDK_BARCODE_TYPE_CODE11	10	Code11
_SDK_BARCODE_TYPE_PLANET	11	Planet
_SDK_BARCODE_TYPE_INDUSTRIAL_2Of5	12	Industrial 2of5
_SDK_BARCODE_TYPE_STANDARD_2Of5	13	Standard 2of5
_SDK_BARCODE_TYPE_LOGMARS	14	Logmars
_SDK_BARCODE_TYPE_UPC_EAN_EXTENSIONS	15	UPC/EAN Extensions
_SDK_BARCODE_TYPE_POSTNET	16	Postnet

## 2-6 Barcode Origin Point

Barcode Origin Point constants are used to set the reference origin position of barcode.

```
typedef enum{
    _SDK_BARCODE_ORIGIN_POINT_CENTER           = 0,
    _SDK_BARCODE_ORIGIN_POINT_UpperLeft      = 1,
}_SDK_BARCODE_ORIGIN_POINT;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_BARCODE_ORIGIN_POINT_CENTER	0	Set the reference point of the barcode to the center.
_SDK_BARCODE_ORIGIN_POINT_UpperLeft	1	Set the reference point of the barcode in the upper left corner.

## 2-7 Error Correction Level

Error Correction Level constants are used to set the level of error correction for possible corruption of barcode.

More corruption can be corrected with a higher level of correction but the length of the allowed data may be reduced with a higher level of correction.

```
typedef enum{
    _SDK_ERROR_CORRECTION_LEVEL0           = 0,
    _SDK_ERROR_CORRECTION_LEVEL1          = 1,
    _SDK_ERROR_CORRECTION_LEVEL2          = 2,
    _SDK_ERROR_CORRECTION_LEVEL3          = 3,
    _SDK_ERROR_CORRECTION_LEVEL4          = 4,
    _SDK_ERROR_CORRECTION_LEVEL5          = 5,
    _SDK_ERROR_CORRECTION_LEVEL6          = 6,
    _SDK_ERROR_CORRECTION_LEVEL7          = 7,
    _SDK_ERROR_CORRECTION_LEVEL8          = 8,
}_SDK_ERROR_CORRECTION_LEVEL
```

Description of each constant is as follows.

Code	Value	Description
_SDK_ERROR_CORRECTION_LEVEL0	0	Error Correction Level 0
_SDK_ERROR_CORRECTION_LEVEL1	1	Error Correction Level 1
_SDK_ERROR_CORRECTION_LEVEL2	2	Error Correction Level 2
_SDK_ERROR_CORRECTION_LEVEL3	3	Error Correction Level 3
_SDK_ERROR_CORRECTION_LEVEL4	4	Error Correction Level 4
_SDK_ERROR_CORRECTION_LEVEL5	5	Error Correction Level 5
_SDK_ERROR_CORRECTION_LEVEL6	6	Error Correction Level 6
_SDK_ERROR_CORRECTION_LEVEL7	7	Error Correction Level 7
_SDK_ERROR_CORRECTION_LEVEL8	8	Error Correction Level 8

## 2-8 Data Compression Method

Data Compression Method constants are used to specify the data compression property.

```
typedef enum{
    _SDK_DATA_COMPRESSION_METHOD_TEXT          = 0,
    _SDK_DATA_COMPRESSION_METHOD_NUMERIC      = 1,
    _SDK_DATA_COMPRESSION_METHOD_BINARY       = 2,
}__SDK_DATA_COMPRESSION_METHOD;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_DATA_COMPRESSION_METHOD_TEXT	0	2char / codeword
_SDK_DATA_COMPRESSION_METHOD_NUMERIC	1	2.93 char / codeword
_SDK_DATA_COMPRESSION_METHOD_BINARY	2	1.2bytes / codeword

## 2-9 QRCode Model

QRCode Model constants are used to set the options in printing QR barcode.

```
typedef enum{
    _SDK_QRCODE_MODEL_1                      = 1,
    _SDK_QRCODE_MODEL_2                      = 2,
}__SDK_QRCODE_MODE;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_QRCODE_MODEL_1	1	QR Model 1
_SDK_QRCODE_MODEL_2	2	QR Model 2

## 2-10 Code 49 Starting Mode

Code 49 Starting Mode constants are used to set the properties of Starting Mode in printing Code 49 barcode.

```
typedef enum{
    _SDK_STARTINGMODE_REGULAR_ALPHANUMERIC           = 0,
    _SDK_STARTINGMODE_MULTIPLE_READ_ALPHANUMERIC     = 1,
    _SDK_STARTINGMODE_REGULAR_NUMERIC                = 2,
    _SDK_STARTINGMODE_GROUP_ALPHANUMERIC             = 3,
    _SDK_STARTINGMODE_REGULAR_ALPHANUMERIC_SHIFT1    = 4,
    _SDK_STARTINGMODE_REGULAR_ALPHANUMERIC_SHIFT2    = 5,
    _SDK_STARTINGMODE_AUTOMATIC                       = 7,
}_SDK_STARTING_MODE;
```

Description of each constant is as follows.

<b>Code</b>	<b>Value</b>	<b>Description</b>
_SDK_STARTINGMODE_REGULAR_ALPHANUMERIC	0	Regular Alphanumeric Mode
_SDK_STARTINGMODE_MULTIPLE_READ_ALPHANUMERIC	1	Multiple Read Alphanumeric
_SDK_STARTINGMODE_REGULAR_NUMERIC	2	Regular Numeric Mode
_SDK_STARTINGMODE_GROUP_ALPHANUMERIC	3	Group Alphanumeric Mode
_SDK_STARTINGMODE_REGULAR_ALPHANUMERIC_SHIFT1	4	Regular Alphanumeric Shift 1
_SDK_STARTINGMODE_REGULAR_ALPHANUMERIC_SHIFT2	5	Regular Alphanumeric Shift 2
_SDK_STARTINGMODE_AUTOMATIC	7	Automatic Mode



## 2-11 Codablock Mode

Codablock Mode constants are used to set the options when printing Codablock barcode.

```
typedef enum{
    _SDK_CODABLACK_MODE_A           = 'A',
    _SDK_CODABLACK_MODE_E           = 'E',
    _SDK_CODABLACK_MODE_F           = 'F',
}_SDK_CODABLACK_MODE;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_CODABLACK_MODE_A	'A'	Code 39 character set is used.
_SDK_CODABLACK_MODE_E	'E'	Code 128 character set is used.
_SDK_CODABLACK_MODE_F	'F'	Code 128 character set is used. Function 1 is added automatically.

## 2-12 Check Digit Option

These constants are used to set the Check Digit property when printing MSI barcode.

```
typedef enum{
    _SDK_CHECKDIGIT_NONE             = 0,
    _SDK_CHECKDIGIT_1MOD10           = 1,
    _SDK_CHECKDIGIT_2MOD10           = 2,
    _SDK_CHECKDIGIT_1MOD11_AND_1MOD_10 = 3,
}_SDK_CHECKDIGIT;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_CHECKDIGIT_NONE	0	No Check Digit
_SDK_CHECKDIGIT_1MOD10	1	Check Digit 1 Mod 10
_SDK_CHECKDIGIT_2MOD10	2	Check Digit 2 Mod 10
_SDK_CHECKDIGIT_1MOD11_AND_1MOD_10	3	Check Digit 1 Mod 10

## 2-13 RSS Barcode Type

RSS Barcode Type constants are used to set the barcode type when printing RSS barcode.

```
typedef enum{
    _SDK_RSS_BARCODE_TYPE_RSS14                = 0,
    _SDK_RSS_BARCODE_TYPE_RSS14_TRUNCATED      = 1,
    _SDK_RSS_BARCODE_TYPE_RSS14_STACKED        = 2,
    _SDK_RSS_BARCODE_TYPE_RSS14_STACKED_OMNIDIRECTIONAL = 3,
    _SDK_RSS_BARCODE_TYPE_RSS_LIMITED          = 4,
    _SDK_RSS_BARCODE_TYPE_RSS_EXPANDED         = 5,
    _SDK_RSS_BARCODE_TYPE_UPC_A                 = 6,
    _SDK_RSS_BARCODE_TYPE_UPC_E                 = 7,
    _SDK_RSS_BARCODE_TYPE_EAN13                 = 8,
    _SDK_RSS_BARCODE_TYPE_EAN8                  = 9,
    _SDK_RSS_BARCODE_TYPE_UCC_EAN128_CC_A_B    = 10,
    _SDK_RSS_BARCODE_TYPE_UCC_EAN128_CC_C      = 11,
}__SDK_RSS_BARCODE_TYPE;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_RSS_BARCODE_TYPE_RSS14	0	RSS14
_SDK_RSS_BARCODE_TYPE_RSS14_TRUNCATED	1	RSS14 truncated
_SDK_RSS_BARCODE_TYPE_RSS14_STACKED	2	RSS14 stacked
_SDK_RSS_BARCODE_TYPE_RSS14_STACKED_OMNIDIRECTIONAL	3	RSS14 Stacked omnidirectional
_SDK_RSS_BARCODE_TYPE_RSS_LIMITED	4	RSS limited
_SDK_RSS_BARCODE_TYPE_RSS_EXPANDED	5	RSS Expanded
_SDK_RSS_BARCODE_TYPE_UPC_A	6	RSS UPC A
_SDK_RSS_BARCODE_TYPE_UPC_E	7	RSS UPC E
_SDK_RSS_BARCODE_TYPE_EAN13	8	EAN13
_SDK_RSS_BARCODE_TYPE_EAN8	9	EAN 8
_SDK_RSS_BARCODE_TYPE_UCC_EAN128_CC_A_B	10	EAN128 CC-A/B
_SDK_RSS_BARCODE_TYPE_UCC_EAN128_CC_C	11	EAN128 CC-C

## **2-14 Rotation Degrees**

Rotation Degrees constants are used to set the rotation property of the printing.

```
typedef enum{  
    _SDK_ROTATION_DEGREES_0           = 0,  
    _SDK_ROTATION_DEGREES_90         = 1,  
    _SDK_ROTATION_DEGREES_180        = 2,  
    _SDK_ROTATION_DEGREES_270        = 3,  
}_SDK_ROTATION_DEGREES;
```

Description of each constant is as follows.

<b>Code</b>	<b>Value</b>	<b>Description</b>
_SDK_ROTATION_DEGREES_0	0	No rotation
_SDK_ROTATION_DEGREES_90	1	90 degrees of rotation
_SDK_ROTATION_DEGREES_180	2	180 degrees of rotation
_SDK_ROTATION_DEGREES_270	3	270 degrees of rotation.

## 2-15 Device Fonts

Device Fonts constants are used to set the property of Device Font.

```
typedef enum {
    _SDK_DEVICE_FONT_6PT           ='0',
    _SDK_DEVICE_FONT_8PT           ='1',
    _SDK_DEVICE_FONT_10PT          ='2',
    _SDK_DEVICE_FONT_12PT          ='3',
    _SDK_DEVICE_FONT_15PT          ='4',
    _SDK_DEVICE_FONT_20PT          ='5',
    _SDK_DEVICE_FONT_30PT          ='6',
    _SDK_DEVICE_FONT_14PT          ='7',
    _SDK_DEVICE_FONT_18PT          ='8',
    _SDK_DEVICE_FONT_24PT          ='9',
    _SDK_DEVICE_FONT_KOREAN1       ='a',
    _SDK_DEVICE_FONT_KOREAN2       ='b',
    _SDK_DEVICE_FONT_KOREAN3       ='c',
    _SDK_DEVICE_FONT_KOREAN4       ='d',
    _SDK_DEVICE_FONT_KOREAN5       ='e',
    _SDK_DEVICE_FONT_KOREAN6       ='f',
    _SDK_DEVICE_FONT_GB2312        ='m',
    _SDK_DEVICE_FONT_BIG5           ='n',
    _SDK_DEVICE_FONT_SHIFT_JIS     ='j',
}_SDK_DEVICE_FONT;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_DEVICE_FONT_6PT	'0'	9 X 15 (dots)
_SDK_DEVICE_FONT_8PT	'1'	12 X 20 (dots)
_SDK_DEVICE_FONT_10PT	'2'	16 X 25 (dots)
_SDK_DEVICE_FONT_12PT	'3'	19 X 30 (dots)
_SDK_DEVICE_FONT_15PT	'4'	24 X 38 (dots)
_SDK_DEVICE_FONT_20PT	'5'	32 X 40 (dots)
_SDK_DEVICE_FONT_30PT	'6'	48 X 76 (dots)
_SDK_DEVICE_FONT_14PT	'7'	22 X 34 (dots)
_SDK_DEVICE_FONT_18PT	'8'	28 X 44 (dots)
_SDK_DEVICE_FONT_24PT	'9'	37 X 58 (dots)
_SDK_DEVICE_FONT_KOREAN1	'a'	16 X 16 (dots) (ascii 9 X 15)
_SDK_DEVICE_FONT_KOREAN2	'b'	24 X 24 (dots) (ascii 12 X 24)
_SDK_DEVICE_FONT_KOREAN3	'c'	20 X 20 (dots) (ascii 12 X 20)
_SDK_DEVICE_FONT_KOREAN4	'd'	26 X 26 (dots) (ascii 16 X 30)
_SDK_DEVICE_FONT_KOREAN5	'e'	20 X 26 (dots) (ascii 16 X 30)
_SDK_DEVICE_FONT_KOREAN6	'f'	38 X 38 (dots) (ascii 22 X 34)
_SDK_DEVICE_FONT_GB2312	'm'	24 X 24 (dots) (ascii 12 X 24)
_SDK_DEVICE_FONT_BIG5	'n'	24 X 24 (dots) (ascii 12 X 24)
_SDK_DEVICE_FONT_SHIFT_JIS	'j'	24 X 24 (dots) (ascii 12 X 24)

## 2-16 Vector Fonts

Vector Fonts constants are used to set the property of Vector Fonts.

```
typedef enum {
    _SDK_VECTOR_FONT_ASCII           ='U',
    _SDK_VECTOR_FONT_KS5601         ='K',
    _SDK_VECTOR_FONT_BIG5            ='B',
    _SDK_VECTOR_FONT_GB2312         ='G',
    _SDK_VECTOR_FONT_SHIFT_JIS      ='J',
    _SDK_VECTOR_FONT_OCR_A           ='a',
    _SDK_VECTOR_FONT_OCR_B           ='b',
}_SDK_VECTOR_FONT;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_VECTOR_FONT_ASCII	'U'	ASCII (1Byte code)
_SDK_VECTOR_FONT_KS5601	'K'	KS5601 (2Byte code)
_SDK_VECTOR_FONT_BIG5	'B'	BIG5 (2Byte code)
_SDK_VECTOR_FONT_GB2312	'G'	GB2312 (2Byte code)
_SDK_VECTOR_FONT_SHIFT_JIS	'J'	Shift-JIS (2Byte code)
_SDK_VECTOR_FONT_OCR_A	'a'	OCR-A (1Byte code)
_SDK_VECTOR_FONT_OCR_B	'b'	OCR-B (1Byte code)

## 2-17 Draw Block Options

Draw Block Options constants are used to set the Draw Block Option.

```
typedef enum {
    _SDK_DRAW_BLOCK_OPTION_LINE_OVERWRITING  ='O',
    _SDK_DRAW_BLOCK_OPTION_LINE_EXCLUSIVE_OR ='E',
    _SDK_DRAW_BLOCK_OPTION_LINE_DELETE       ='D',
    _SDK_DRAW_BLOCK_OPTION_SLOPE             ='S',
    _SDK_DRAW_BLOCK_OPTION_BOX               ='B',
}_SDK_DRAW_BLOCK_OPTION;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_DRAW_BLOCK_OPTION_LINE_OVERWRITING	'O'	Line Overwriting
_SDK_DRAW_BLOCK_OPTION_LINE_EXCLUSIVE_OR	'E'	Line Exclusive OR
_SDK_DRAW_BLOCK_OPTION_LINE_DELETE	'D'	Line Delete
_SDK_DRAW_BLOCK_OPTION_SLOPE	'S'	Slope(a oblique line)
_SDK_DRAW_BLOCK_OPTION_BOX	'B'	Box

## 2-18 Draw Circle Sizes

Draw Circle Sizes constants are used to set the property related to the size when using the Draw Circle Method.

```
typedef enum {
    _SDK_DRAW_CIRCLE_SIZE_40X40           = 1,    // 5(mm) diameter
    _SDK_DRAW_CIRCLE_SIZE_56X56           = 2,
    _SDK_DRAW_CIRCLE_SIZE_72X72           = 3,
    _SDK_DRAW_CIRCLE_SIZE_88X88           = 4,
    _SDK_DRAW_CIRCLE_SIZE_104X104         = 5,
    _SDK_DRAW_CIRCLE_SIZE_168X168         = 6,
}_SDK_DRAW_CIRCLE_SIZE;
```

Description of each constant is as follows.

Code	Value	Diameter (mm)	Width x Height(Dots)
_SDK_DRAW_CIRCLE_SIZE_40X40	1	5	40 · 40
_SDK_DRAW_CIRCLE_SIZE_56X56	2	7	56 · 56
_SDK_DRAW_CIRCLE_SIZE_72X72	3	9	72 · 72
_SDK_DRAW_CIRCLE_SIZE_88X88	4	11	88 · 88
_SDK_DRAW_CIRCLE_SIZE_104X104	5	13	104 · 104
_SDK_DRAW_CIRCLE_SIZE_168X168	6	21	168 · 168

## 2-19 International Character Set

These constants are used to set the International Character Set.

```
typedef enum {
    _SDK_CONFIG_ICS_USA           = 0,
    _SDK_CONFIG_ICS_FRANCE       = 1,
    _SDK_CONFIG_ICS_GERMANY      = 2,
    _SDK_CONFIG_ICS_UK           = 3,
    _SDK_CONFIG_ICS_DENMARK_I    = 4,
    _SDK_CONFIG_ICS_SWEDEN       = 5,
    _SDK_CONFIG_ICS_ITALY        = 6,
    _SDK_CONFIG_ICS_SPAIN_I      = 7,
    _SDK_CONFIG_ICS_NORWAY       = 8,
    _SDK_CONFIG_ICS_DENMARK_II   = 9,
    _SDK_CONFIG_ICS_JAPAN        = 10,
    _SDK_CONFIG_ICS_SPAIN_II     = 11,
    _SDK_CONFIG_ICS_LATIN_AMERICA = 12,
    _SDK_CONFIG_ICS_KOREA        = 13,
    _SDK_CONFIG_ICS_SLOVENIA_CROATIA = 14,
    _SDK_CONFIG_ICS_CHINA        = 15,
}_SDK_CONFIG_ICS;
```

Description of each constant is as follows.

Code	Value	Description
_SDK_CONFIG_ICS_USA	0	U.S.A
_SDK_CONFIG_ICS_FRANCE	1	France
_SDK_CONFIG_ICS_GERMANY	2	Germany
_SDK_CONFIG_ICS_UK	3	U.K
_SDK_CONFIG_ICS_DENMARK_I	4	Denmark I
_SDK_CONFIG_ICS_SWEDEN	5	Sweden
_SDK_CONFIG_ICS_ITALY	6	Italy
_SDK_CONFIG_ICS_SPAIN_I	7	Spain I
_SDK_CONFIG_ICS_NORWAY	8	Norway
_SDK_CONFIG_ICS_DENMARK_II	9	Denmark II
_SDK_CONFIG_ICS_JAPAN	10	Japan
_SDK_CONFIG_ICS_SPAIN_II	11	Spain II
_SDK_CONFIG_ICS_LATIN_AMERICA	12	Latin America
_SDK_CONFIG_ICS_KOREA	13	Korea
_SDK_CONFIG_ICS_SLOVENIA_CROATIA	14	Slovenia/Croatia
_SDK_CONFIG_ICS_CHINA	15	China

## **2-20 Code Pages**

These constants are used to set the Code Page.

```
typedef enum {  
    _SDK_CONFIG_CODEPAGE_CP437           = 0,  
    _SDK_CONFIG_CODEPAGE_CP850           = 1,  
    _SDK_CONFIG_CODEPAGE_CP852           = 2,  
    _SDK_CONFIG_CODEPAGE_CP860           = 3,  
    _SDK_CONFIG_CODEPAGE_CP863           = 4,  
    _SDK_CONFIG_CODEPAGE_CP865           = 5,  
    _SDK_CONFIG_CODEPAGE_CP81252         = 6,  
    _SDK_CONFIG_CODEPAGE_CP865_WCP1252   = 7,  
    _SDK_CONFIG_CODEPAGE_CP857           = 8,  
    _SDK_CONFIG_CODEPAGE_CP737           = 9,  
    _SDK_CONFIG_CODEPAGE_WPC1250         = 10,  
    _SDK_CONFIG_CODEPAGE_WPC1253         = 11,  
    _SDK_CONFIG_CODEPAGE_WPC1254         = 12,  
    _SDK_CONFIG_CODEPAGE_CP855           = 13,  
    _SDK_CONFIG_CODEPAGE_CP862           = 14,  
    _SDK_CONFIG_CODEPAGE_CP866           = 15,  
    _SDK_CONFIG_CODEPAGE_WCP1251         = 16,  
    _SDK_CONFIG_CODEPAGE_WCP1255         = 17,  
    _SDK_CONFIG_CODEPAGE_CP928           = 18,  
    _SDK_CONFIG_CODEPAGE_CP864           = 19,  
    _SDK_CONFIG_CODEPAGE_CP775           = 20,  
    _SDK_CONFIG_CODEPAGE_WCP1257         = 21,  
    _SDK_CONFIG_CODEPAGE_CP858           = 22,  
}_SDK_CONFIG_CODEPAGE;
```



Description of each constant is as follows.

<b>Code</b>	<b>Value</b>		
<code>_SDK_CONFIG_CODEPAGE_CP437</code>	0	CP437	U.S.A
<code>_SDK_CONFIG_CODEPAGE_CP850</code>	1	CP850	Latin1
<code>_SDK_CONFIG_CODEPAGE_CP852</code>	2	CP 852	Latin2
<code>_SDK_CONFIG_CODEPAGE_CP860</code>	3	CP 860	Portuguese
<code>_SDK_CONFIG_CODEPAGE_CP863</code>	4	CP 863	Canadian French
<code>_SDK_CONFIG_CODEPAGE_CP865</code>	5	CP 865	Nordic
<code>_SDK_CONFIG_CODEPAGE_CP1252</code>	6	WCP 1252	Latin I
<code>_SDK_CONFIG_CODEPAGE_CP865_WCP1252</code>	7	CP 865 + WCP 1252	European Combined
<code>_SDK_CONFIG_CODEPAGE_CP857</code>	8	CP 857	Turkish
<code>_SDK_CONFIG_CODEPAGE_CP737</code>	9	CP 737	Greek
<code>_SDK_CONFIG_CODEPAGE_WPC1250</code>	10	WCP 1250	Latin 2
<code>_SDK_CONFIG_CODEPAGE_WPC1253</code>	11	WCP 1253	Greek
<code>_SDK_CONFIG_CODEPAGE_WPC1254</code>	12	WCP 1254	Turkish
<code>_SDK_CONFIG_CODEPAGE_CP855</code>	13	CP 855	Cyrillic
<code>_SDK_CONFIG_CODEPAGE_CP862</code>	14	CP 862	Hebrew
<code>_SDK_CONFIG_CODEPAGE_CP866</code>	15	CP 866	Cyrillic
<code>_SDK_CONFIG_CODEPAGE_WCP1251</code>	16	WCP 1251	Cyrillic
<code>_SDK_CONFIG_CODEPAGE_WCP1255</code>	17	WCP 1255	Hebrew
<code>_SDK_CONFIG_CODEPAGE_CP928</code>	18	CP 928	Greek
<code>_SDK_CONFIG_CODEPAGE_CP864</code>	19	CP 864	Arabic
<code>_SDK_CONFIG_CODEPAGE_CP775</code>	20	CP 775	Baltic
<code>_SDK_CONFIG_CODEPAGE_WCP1257</code>	21	WCP1257	Baltic
<code>_SDK_CONFIG_CODEPAGE_CP858</code>	22	CP858	Latin 1 + Euro

## 3. LabelPrinterSDK Class Reference

<b>Inherits from</b>	NSObject
<b>Framework</b>	libLabelPrinterSDK.a
<b>Declared</b>	LabelPrinterSDK.h

### 3-1 Overview

libLabelPrinterSDK Class is the main object that controls the printer operation.

### 3-2 Methods

#### 3-2-1 open

---

This method initializes the settings to use the LabelPrinterSDK class.

**[Syntax]**

```
-(long) open;
```

**[Return Value]**

\_SDK\_RESULT\_SUCCESS will be returned upon successful initialization.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[Discussion]**

This method should be called before calling other methods in the LabelPrinterSDK.

**[See Also]**

[2-1 Result Codes](#)

#### 3-2-2 close

---

This method terminates the use of the LabelPrinterSDK class.

**[Syntax]**

```
-(long) close;
```

**[Return Value]**

\_SDK\_RESULT\_SUCCESS will be returned upon successful operation.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[Discussion]**

All resources inside the LabelPrinterSDK will be released when this method is called.  
This method should be called to end the use of the printer.

**[See Also]**

[2-1 Result Codes](#)

### **3-2-3 connectWithAddress**

---

This method tries to connect to the printer.

**[Syntax]**

-(long) connectWithAddress:(NSString\*)address port:(NSString\*) port;

**[Parameters]**

*address*

WiFi Printer: Enter the IP Address.

Bluetooth Printer: Enter the Mac Address.

*port :*

Enter the communication port of the printer.

Communication port of the BIXOLON printers is 9100.

Bluetooth printers are not supported.

**[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[See Also]**

[2-1 Result Codes](#)

### **3-2-4 isConnected**

---

This method checks the status of printer connection.

**[Syntax]**

-(BOOL) isConnected;

**[Return Value]**

YES will be returned if printer is connected.

### **3-2-5 disconnect**

---

This method disconnects the printer.

**[Syntax]**

-(long) disconnect;

**[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[See Also]**

[2-1 Result Codes](#)

### **3-2-6 disconnectWithTimeout**

---

This method disconnects the printer within the specified Timeout period.

**[Syntax]**

-(long) disconnectWithTimeout:(NSInteger)timeout;

**[Parameters]**

*timeout*

Set the Timeout period. (Unit: Second)

**[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[Discussion]**

If data remains in the output buffer, buffer will be cleared and printer will be disconnected within the Timeout period. If this parameter is set to -1, this method does not return until all buffers are cleared.

**[See Also]**

[2-1 Result Codes](#)

### **3-2-7 getDeviceFontList**

---

This method gets information and string values declared in relation to DeviceFont in NSMutableArray data type.

**[Syntax]**

-(NSMutableArray\*) getDeviceFontList;

**[Return Value]**

“nil” will be returned if the operation fails.

**[See Also]**

[2-12 Device Fonts](#)

### **3-2-8 getVectorFontList**

---

This method gets information and string values declared in relation to VectorFont in NSMutableArray data type.

**[Syntax]**

-(NSMutableArray\*) getVectorFontList

**[Return Value]**

“nil” will be returned if the operation fails.

**[See Also]**

[2-16 Vector Fonts](#)

### **3-2-9 getDrawBlockOptions**

---

This method gets information and string values declared in relation to Draw Block Options in NSMutableArray data type.

**[Syntax]**

-(NSMutableArray\*) getDrawBlockOptions;

**[Return Value]**

“nil” will be returned if the operation fails.

**[See Also]**

[2-17 Draw Block Options](#)

### **3-2-10 getDrawCircleSizes**

---

This method gets information and string values declared in relation to Draw Circle Sizes in NSMutableArray data type.

**[Syntax]**

-(NSMutableArray\*) getDrawCircleSizes;

**[Return Value]**

“nil” will be returned if the operation fails.

**[See Also]**

[2-18 Draw Circle Sizes](#)

### **3-2-11 getBarcodeType1D**

---

This method gets information and string values declared in relation to 1D Barcode Type in NSMutableArray data type.

**[Syntax]**

-(NSMutableArray\*) getBarcodeType1D;

**[Return Value]**

“nil” will be returned if the operation fails.

**[See Also]**

[2-5 1D Barcode Types](#)

### **3-2-12 getBarcodeHRI**

---

This method gets information and string values declared in relation to Barcode HRI in NSMutableArray data type.

**[Syntax]**

```
-(NSMutableArray*) getBarcodeHRI;
```

**[Return Value]**

“nil” will be returned if the operation fails.

**[See Also]**

[2-3 1D Barcode HRI](#)

### **3-2-13 doPrint**

---

This method prints the contents stored in the output buffer of the printer.

**[Syntax]**

```
-(long) doPrint:(NSInteger)numberOfCopies;
```

**[Parameters]**

*numberOfCopies*

Number of copies to print.

**[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[See Also]**

[2-1 Result Codes](#)

### 3-2-14 drawTextDeviceFont

---

This method requests the printing of text string to the image buffer using the Device Fonts.

#### [Syntax]

```

-(long) drawTextDeviceFont:(NSString*)text
        xPosition:(NSInteger)xPosition
        yPosition:(NSInteger)yPosition
        fontSelection:(char)fontSelection
        fontWidth:(NSInteger)fontWidth
        fontHeight:(NSInteger)fontHeight
        rightSideCharacterSpacing:(NSInteger)rightSideCharacterSpacing
        fontRotation:(NSInteger)fontRotation
        reverse:(BOOL)reverse
        bold:(BOOL)bold
        textAlignment:(NSInteger)textAlignment;

```

#### [Parameters]

*text*

Text string to print.

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*fontSelection*

Selection of fonts to print (Reference: [2-15 Device Fonts](#))

*fontWidth*

Horizontal magnification of the selected font (1~4).

*fontHeight*

Vertical magnification of the selected font (1~4).

*rightSideCharacterSpacing*

Right side margin of the characters (ex: 5, +3, -10...).

*fontRotation*

Rotation setting of the selected font (Reference: [2-14 Rotation Degrees](#))

*reverse*

Option to use the **Reverse Image Font**.

**Reverse Image Font** will be used for printing if this parameter is set to YES.

*bold*

Option to use the **Bold Font**.

**Bold Font** will be used for printing if this parameter is set to YES.

*textAlignment*

Alignment method (Reference: [2-2 Alignment](#))

### **[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### **[Discussion]**

- When `textAlignment` is set to `_SDK_ALIGNMENT_CENTER`, it has the same effect as the `_SDK_ALIGNMENT_STRING_FROM_RIGHT_2_LEFT` option.
- Nothing will be printed when this API is called.  
Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### **[See Also]**

[2-1 Result Codes](#)

[2-2 Alignment](#)

[2-14 Rotation Degrees](#)

[2-15 Device Fonts](#)

[3-2-13 doPrint](#)



### 3-2-15 drawTextVectorFont

---

This method requests the printing of text string to the image buffer using the Vector Fonts.

#### [Syntax]

```

-(long) drawTextVectorFont:(NSString*)text
        xPosition:(NSInteger)xPosition
        yPosition:(NSInteger)yPosition
        fontSelection:(char)fontSelection
        fontWidth:(NSInteger)fontWidth
        fontHeight:(NSInteger)fontHeight
        rightSideCharacterSpacing:(NSInteger)rightSideCharacterSpacing
        fontRotation:(NSInteger)fontRotation
        reverse:(BOOL)reverse
        bold:(BOOL)bold
        italic:(BOOL)italic
        textWriteDirectionRightToLeft:(BOOL)textWriteDirectionRightToLeft
        textAlignment:(NSInteger)textAlignment;

```

#### [Parameters]

*text*

Text string to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*fontSelection*

Selection of fonts to print (Reference: [2-15 Device Fonts](#))

*fontWidth*

Horizontal magnification of the selected font (1~4).

*fontHeight*

Vertical magnification of the selected font (1~4).

*rightSideCharacterSpacing*

Right side margin of the characters (ex: 5, +3, -10...).

*fontRotation*

Rotation setting of the selected font (Reference: [2-14 Rotation Degrees](#))

*reverse*

Option to use the **Reverse Image Font**.

**Reverse Image Font** will be used for printing if this parameter is set to YES.

*bold*

Option to use the **Bold Font**.

**Bold Font** will be used for printing if this parameter is set to YES.

*italic*

Option to use the *Italic Font*.

*Italic Font* will be used for printing if this parameter is set to YES.

*textWriteDirectionRightToLeft*

Option to print the character string in the direction from right to left

*textAlignment*

Alignment method (Reference: [2-2 Alignment](#))

### **[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### **[Discussion]**

- When `textAlignment` is set to `_SDK_ALIGNMENT_STRING_FROM_RIGHT_2_LEFT`, it has the same effect as the `_SDK_ALIGNMENT_CENTER` option.
- Nothing will be printed when this API is called.  
Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### **[See Also]**

[2-1 Result Codes](#)

[2-2 Alignment](#)

[2-14 Rotation Degrees](#)

[2-15 Device Fonts](#)

[3-2-13 doPrint](#)

## 3-2-16 drawBarcode1D

---

This method requests the printing of 1D barcode to the image buffer.

### [Syntax]

```

-(long) drawBarcode1D:(NSString*)data
      xPosition:(NSInteger)xPosition
      yPosition:(NSInteger)yPosition
      barcodeType:(NSInteger)barcodeType
      widthNarrow:(NSInteger)widthNarrow
      widthWide:(NSInteger)widthWide
      height:(NSInteger)height
      hri:(NSInteger)hri
      quietZoneWidth:(NSInteger)quietZoneWidth
      rotation:(NSInteger)rotation;

```

### [Parameters]

*text*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*barcodeType*

Barcode type. (Reference: [2-5 1D Barcode Types](#))

*widthNarrow*

Width of narrow bar

*widthWide*

Width of wide bar

*height*

Height of barcode

*hri*

Print position of barcode data value (Human Readable Interpretation)  
(Reference: [2-3 Barcode HRI](#))

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-5 1D Barcode Types](#)

[2-14 Rotation Degrees](#)

[2-3 Barcode HRI](#)

[3-2-13 doPrint](#)

## **3-2-17 drawBarcodeMaxiCode**

---

This method requests the printing of MaxiCode barcode to the image buffer.

### **[Syntax]**

```
-(long) drawBarcodeMaxiCode:(NSString*)data  
        xPosition:(NSInteger)xPosition  
        yPosition:(NSInteger)yPosition  
        mode:(NSInteger)mode;
```

### **[Parameters]**

*text*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*mode*

Mode of MaxiCode (Reference: [2-4 MaxiCode Modes](#))

### **[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### **[Discussion]**

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### **[See Also]**

[2-1 Result Codes](#)

[2-4 MaxiCode Modes](#)

[3-2-13 doPrint](#)

## 3-2-18 drawBarcodePDF417

---

This method requests the printing of PDF417 barcode to the image buffer.

### [Syntax]

```

-(long) drawBarcodePDF417:(NSString*)data
        xPosition:(NSInteger)xPostion
        yPosition:(NSInteger)yPostion
        maximumRowCount:(NSInteger)maximumRowCount // 3~90
        maximumColumnCount:(NSInteger)maximumColumnCount // 1~30
        errorCorrectionLevel:(NSInteger)errorCorrectionLevel
        dataCompressionMethod:(NSInteger)dataCompressionMethod
        printBarcodeText:(BOOL)printBarcodeText
        barcodeOriginPoint:(NSInteger)barcodeOriginPoint
        moduleWidth:(NSInteger)moduleWidth // 2~9
        barHeight:(NSInteger)barHeight // 4~99
        rotation:(NSInteger)rotation;

```

### [Parameters]

*text*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*maximumRowCount*

maximum Row Count (3~90).

*errorCorrectionLevel*

Error correction level (Reference: [2-7 Error Correction Level](#))

*dataCompressionMethod*

Data Compression Method (Reference: [2-8 Data Compression Method](#))

*printBarcodeText*

HRI printing option.

*barcodeOriginPoint*

Position to be used as the reference point of barcode

(Reference: [2-6 Barcode Origin Point](#))

*moduleWidth*

Module Width (2~9)

*BarHeight*

Height of barcode (4~99)

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

**[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[Discussion]**

Nothing will be printed when this API is called.  
Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

**[See Also]**

[2-1 Result Codes](#)

[2-6 Barcode Origin Point](#)

[2-7 Error Correction Level](#)

[2-8 Data Compression Method](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)

## 3-2-19 drawBarcodeQRCode

---

This method requests the printing of QRCode barcode to the image buffer.

### [Syntax]

```
-(long) drawBarcodeQRCode:(NSString*)data
                    xPosition:(NSInteger)xPosition
                    yPosition:(NSInteger)yPosition
                    barcodeSize:(NSInteger)barcodeSize // 1~4
                    model:(NSInteger)model
                    errorColectionLevel:(NSInteger)errorColectionLevel
                    rotation:(NSInteger)rotation;
```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*barcodeSize*

Size of barcode (1~4)

*model*

Model of QRCode (Reference: [2-9 QRCode Model](#))

*errorCorrectionLevel*

Error Correction Level (Reference: [2-7 Error Correction Level](#))

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-4 MaxiCode Modes](#)

[2-9 QRCode Model](#)

[2-14 Rotation Degrees](#))

[3-2-13 doPrint](#)

## 3-2-20 drawBarcodeDataMatrix

---

This method requests the printing of Data Matrix barcode to the image buffer.

### [Syntax]

```
-(long) drawBarcodeDataMatrix:(NSString*)data
                        xPosition:(NSInteger)xPosition
                        yPosition:(NSInteger)yPosition
                        barcodeSize:(NSInteger)barcodeSize // 1~4
                        reverse:(BOOL)reverse
                        rotation:(NSInteger)rotation;
```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*barcodeSize*

Size of barcode (1~4)

*reverse*

Reverse mode

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)



### **3-2-21 drawBarcodeAztec**

---

This method requests the printing of Aztec barcode to the image buffer.

**[Syntax]**

```

-(long) drawBarcodeAztec:(NSString*)data
        xPosition:(NSInteger)xPosition
        yPosition:(NSInteger)yPosition
        barcodeSize:(NSInteger)barcodeSize // 1~10
        extendedChannel:(BOOL)extendedChannel
        errorCorrectionLevel:(NSInteger)errorCorrectionLevel
        menuSymbol:(BOOL)menuSymbol
        numberOfSymbols:(NSInteger)numberOfSymbols
        optionalID:(NSString*)optionalID
        rotation:(NSInteger)rotation;
    
```

**[Parameters]**

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*barcodeSize*

Size of barcode (1~10)

*extendedChannel*

Reverse mode

*errorCorrectionLevel*

Error Correction Level

Value	Error control and symbol size/type
0	Default error correction level
1 ~ 99	Error correction percentage
101 ~ 104	1 ~ 4 layer compact symbol
201 ~ 232	1 ~ 32 layer full range symbol
300	Simple Aztec "Rune"

*menuSymbol*

Option to use menu symbol

*numberOfSymbols*

Number of symbols for structured append: (1 ~ 26)

*optionalID*

Optional ID field for structured append: ID field string (Maximum 24 character)

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

**[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

**[Discussion]**

Nothing will be printed when this API is called.  
Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

**[See Also]**

[2-1 Result Codes](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)

## 3-2-22 drawBarcodeCode49

---

This method requests the printing of Code49 barcode to the image buffer.

### [Syntax]

```
-(long)drawBarcodeCode49:(NSString*)data
    xPosition:(NSInteger)xPosition
    yPosition:(NSInteger)yPosition
    widthNarrow:(NSInteger)widthNarrow
    widthWide:(NSInteger)widthWide
    height:(NSInteger)height
    hri:(NSInteger)hri
    startingMode:(NSInteger)startingMode
    rotation:(NSInteger)rotation;
```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*widthNarrow*

Width of narrow bar

*widthWide*

Width of wide bar

*hri*

Print position of barcode data value (Human Readable Interpretation)  
(Reference: [2-3 Barcode HRI](#))

*height*

Height of barcode

*startingMode*

Starting mode (Reference: [2-10 Code 49 Starting Mode](#))

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-3 Barcode HRI](#)

[2-10 Code 49 Starting Mode](#)

[2-14 Rotation Degrees](#)

[2-3 Barcode HRI](#)

[3-2-13 doPrint](#)

## **3-2-23 drawBarcodeCodaBlock**

This method requests the printing of CodaBlock barcode to the image buffer.

### **[Syntax]**

```

-(long) drawBarcodeCodaBlock:(NSString*)data
    xPosition:(NSInteger)xPosition
    yPosition:(NSInteger)yPosition
    widthNarrow:(NSInteger)widthNarrow
    widthWide:(NSInteger)widthWide
    height:(NSInteger)height
    securityLevel:(BOOL)securityLevel
    numberOfCharactersPerrow:(NSInteger)numberOfCharactersPerrow
    mode:(char)mode
    numberOfRowsToEncode:(NSInteger)numberOfRowsToEncode;
    
```

### **[Parameters]**

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*widthNarrow*

Width of narrow bar

*widthWide*

Width of wide bar

*height*

Height of barcode

*securityLevel*

securityLevel setting

*numberOfCharactersPerrow*

Number of characters per row (data columns): 2~62

*mode*

Barcode printing mode (Reference: [2-11 Codablock Mode](#))

*numberOfRowsToEncode*

Number of rows to encode

Mode	Value
_SDK_CODABLACK_MODE_A	1 ~ 18
_SDK_CODABLACK_MODE_E	2 ~ 4
_SDK_CODABLACK_MODE_F	2 ~ 4

### **[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### **[Discussion]**

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### **[See Also]**

[2-1 Result Codes](#)

[2-11 Codablock Mode](#)

[3-2-13 doPrint](#)

## 3-2-24 drawBarcodeMicroPDF

---

This method requests the printing of Micro PDF417 barcode to the image buffer.

### [Syntax]

```
-(long) drawBarcodeMicroPDF:(NSString*)data
      xPosition:(NSInteger)xPosition
      yPosition:(NSInteger)yPosition
      moduleWidth:(NSInteger)moduleWidth // 2~8
      barcodeHeight:(NSInteger)barcodeHeight // 1~99
      mode:(NSInteger)mode // 0~33
      rotation:(NSInteger)rotation;
```

### [Parameters]

*text*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*moduleWidth*

module Width (2~8)

*barcodeHeight*

Height of barcode (1~99)

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)

## **3-2-25 drawBarcodeIMB**

---

This method requests the printing of drawBarcodeIMB barcode to the image buffer.

### **[Syntax]**

```
-(long) drawBarcodeIMB:(NSString*)data  
      xPosition:(NSInteger)xPosition  
      yPosition:(NSInteger)yPosition  
      printBarcodeText:(BOOL)printBarcodeText  
      rotation:(NSInteger)rotation;
```

### **[Parameters]**

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*printBarcodeText*

HRI printing option

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### **[Return Value]**

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### **[Discussion]**

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### **[See Also]**

[2-1 Result Codes](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)

## 3-2-26 drawBarcodeMSI

---

This method requests the printing of drawBarcodeMSI barcode to the image buffer.

### [Syntax]

```

-(long) drawBarcodeMSI:(NSString*)data
    xPosition:(NSInteger)xPosition
    yPosition:(NSInteger)yPosition
    widthNarrow:(NSInteger)widthNarrow
    widthWide:(NSInteger)widthWide
    height:(NSInteger)height
    checkDigitSelection:(NSInteger)checkDigitSelection
    printCheckDigitInHRI:(BOOL)printCheckDigitInHRI
    hri:(NSInteger)hri
    rotation:(NSInteger)rotation;

```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*widthNarrow*

Width of narrow bar

*widthWide*

Width of wide bar

*height*

Height of barcode.

*checkDigitSelection*

checkDigit option (Reference: [2-12 Check Digit Option](#))

*printCheckDigitInHRI*

Option to include check digit in HRI

*hri*

Print position of barcode data value (Human Readable Interpretation)  
(Reference: [2-3 Barcode HRI](#))

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-3 Barcode HRI](#)

[2-12 Check Digit Option](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)

## 3-2-27 drawBarcodePlessey

---

This method requests the printing of drawBarcodePlessey barcode to the image buffer.

### [Syntax]

```

-(long) drawBarcodePlessey:(NSString*)data
    xPosition:(NSInteger)xPosition
    yPosition:(NSInteger)yPosition
    widthNarrow:(NSInteger)widthNarrow
    widthWide:(NSInteger)widthWide
    height:(NSInteger)height
    printCheckDigit:(BOOL)printCheckDigit
    hri:(NSInteger)hri
    rotation:(NSInteger)rotation;

```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*widthNarrow*

Width of narrow bar

*widthWide*

Width of wide bar

*height*

Height of barcode.

*printCheckDigitInHRI*

Option to include check digit in HRI

*hri*

Print position of barcode data value (Human Readable Interpretation)  
(Reference: [2-3 Barcode HRI](#))

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.  
Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-3 Barcode HRI](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)



## 3-2-28 drawBarcodeTLC39

---

This method requests the printing of drawBarcodeTLC39 barcode to the image buffer.

### [Syntax]

```
-(long) drawBarcodeTLC39:(NSString*)data
      xPosition:(NSInteger)xPosition
      yPosition:(NSInteger)yPosition
      widthNarrow:(NSInteger)widthNarrow
      widthWide:(NSInteger)widthWide
      height:(NSInteger)height
      rowHeightOfMicroPDF417:(NSInteger)rowHeightOfMicroPDF417
      narrowWidthOfMicroPDF417:(NSInteger)narrowWidthOfMicroPDF417
      rotation:(NSInteger)rotation;
```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*widthNarrow*

Width of narrow bar

*widthWide*

Width of wide bar

*height*

Height of barcode.

*rowHeightOfMicroPDF417*

Row height of microPDF417 barcode

*narrowWidthOfMicroPDF417*

narrowWideHeight of microPDF417 barcode

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-14 Rotation Degrees](#)

[3-2-13 doPrint](#)

## 3-2-29 drawBarcodeRSS

---

This method requests the printing of drawBarcodeRSS barcode to the image buffer.

### [Syntax]

```

-(long) drawBarcodeRSS:(NSString*)data
    xPosition:(NSInteger)xPosition
    yPosition:(NSInteger)yPosition
    barcodeType:(NSInteger)barcodeType
    magnification:(NSInteger)magnification // 1~10
    separatorHeight:(NSInteger)separatorHeight // 1~2
    barcodeHeight:(NSInteger)barcodeHeight
    segmentWidth:(NSInteger)segmentWidth // 0~22
    rotation:(NSInteger)rotation;

```

### [Parameters]

*data*

Barcode value to print

*xPosition*

X coordinate of the position to print

*yPosition*

Y coordinate of the position to print

*barcodeType*

RSS barcode type (Reference: [2-13 RSS Barcode Type](#)).

*magnification*

*magnification*. (1~10)

*separatorHeight*

Height of separator (1~2)

*barcodeHeight*

Height of barcode.

*segmentWidth*

segmentWidth(0~22)

*rotation*

Rotation setting of barcode (Reference: [2-14 Rotation Degrees](#))

### [Return Value]

Upon successful operation, `_SDK_RESULT_SUCCESS` will be returned.

Refer to the [2-1 Result Codes](#) for more details on other result codes.

### [Discussion]

Nothing will be printed when this API is called.

Contents requested by this API will be printed when [3-2-13 doPrint](#) API is called.

### [See Also]

[2-1 Result Codes](#)

[2-13 RSS Barcode Type](#)

[2-14 Rotation Degrees](#)

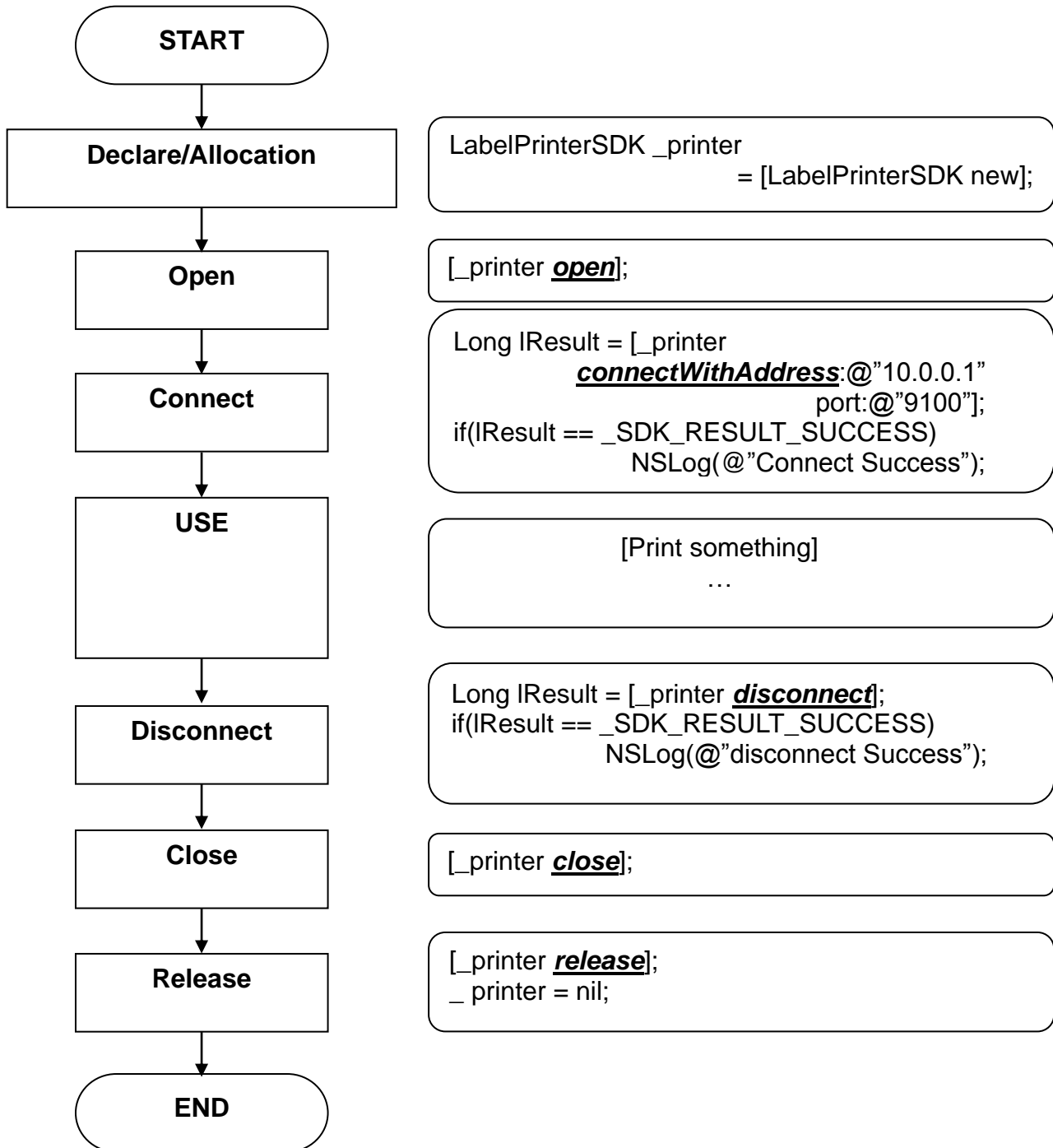
[3-2-13 doPrint](#)

## 4. Appendix

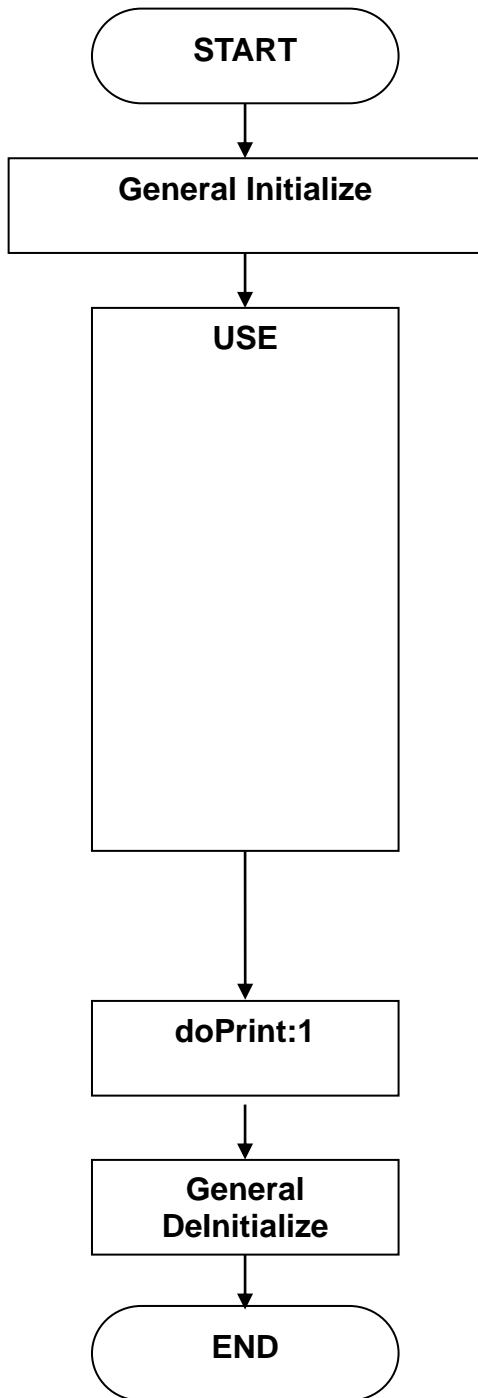
### 4-1 Recommended Flow of Operation

#### 4-1-1 General

---



## 4-1-2 Printing Contents



General ***Initialize*** ... (Declare ~ Connect)  
Reference: [4-1-1 General](#)

```

// [USE] Functions to print barcode, text,
// image, and various shapes can be used.
Long IResult =
[_printer drawImage:_imageSample.image
  stratPosX:100
  startPosY:100
  width:300];

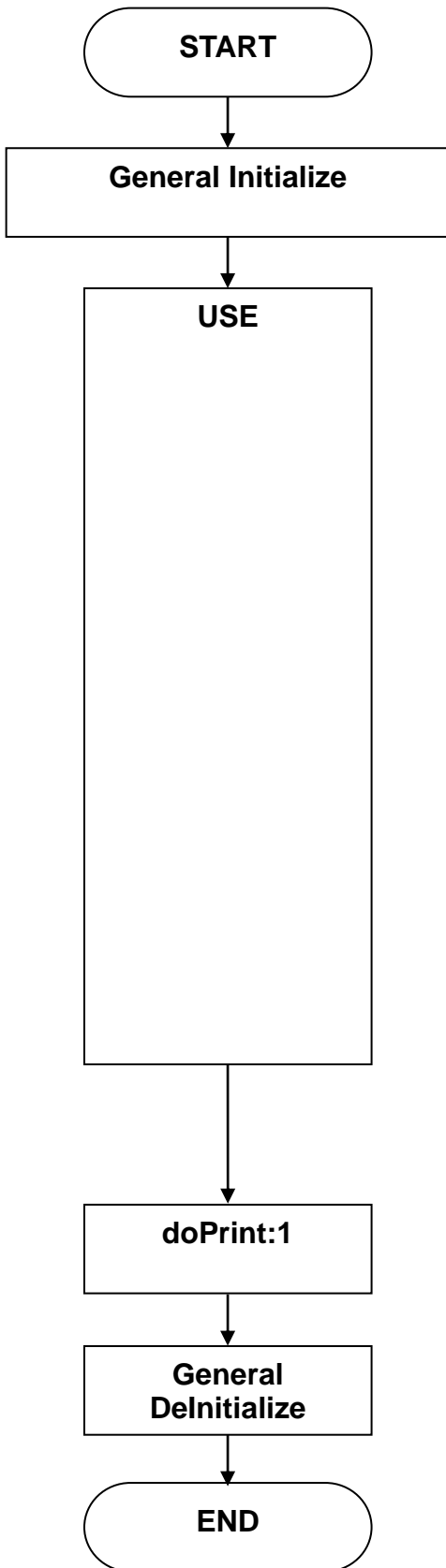
if(IResult = _SDK_RESULT_SUCCESS)
  NSLog(@"drawImage Success");
  
```

```

Long IResult = [_printer doPrint:1];
if(IResult = _SDK_RESULT_SUCCESS)
  NSLog(@"drawImage Success");
  
```

General ***Delnitialize*** ... (Declare ~ Connect)  
Reference : [4-1-1 General](#)

### 4-1-3 Printing Two or More Contents on One Side



General ***Initialize*** ... (Declare ~ Connect)  
Reference : [4-1-1 General](#)

```

// [Contents 1]
Long IResult =
[_printer drawImage: _imageSample.image
 stratPosX:100
 startPosY:100
 width:300];
if(IResult = _SDK_RESULT_SUCCESS)
  NSLog(@"drawImage Success");
// [Contents 2]
IResult =
[_printer drawTextDeviceFont:@"0" 12pt,
 70, 220" xPosition:70 yPosition:220
 fontSelection:_SDK_DEVICE_FONT_12PT
 fontWidth:1 fontHeight:1
 rightSideCharacterSpacing:0
 fontRotation:_SDK_ROTATION_DEGREES
 _0 reverse:NO bold:NO
 textAlignment:_SDK_ALIGNMENT_LEFT];
if(IResult = _SDK_RESULT_SUCCESS)
  NSLog(@"drawText Success");
  
```

```

Long IResult = [_printer doPrint:1];
if(IResult = _SDK_RESULT_SUCCESS)
  NSLog(@"drawImage Success");
  
```

General ***Delinitialize*** ... (Declare ~ Connect)  
Reference : [4-1-1 General](#)

## 4-2 Sample Codes

### 4-2-1 Connecting/Disconnecting

```
// Declaration/Allocation and open SDK
```

```
LabelPrinterSDK _printer = [LabelPrinterSDK new];  
[_printer open];
```

```
// Connect to the printer that is connected with IP Address "10.0.0.1" and Port  
9100
```

```
long IResult = [_printer connectWithAddress:@"10.0.0.1" port:@"9100"];  
if(IResult == _SDK_RESULT_SUCCESS)  
    NSLog(@"Connect Success");
```

```
// After finishing the use of SDK
```

```
IResult = [_printer disconnect]; // Disconnect  
[_printer release];
```

### 4-2-2 Printing Contents in the Buffer

```
long IResult = [_printer doPrint:1]; // Any number bigger than 1 will  
repeat the same printing for the specified number of times.
```

```
if(IResult == _SDK_RESULT_SUCCESS)  
    NSLog(@"doPrint Success");
```

### 4-2-3 Printing a Rectangular Box to the Image Buffer

```
IResult = [_printer drawBlock:100  
startPosY:100  
endPosX:300 endPosY:300  
option:_SDK_DRAW_BLOCK_OPTION_SLOPE  
thickness:1];
```

```
NSLog(@" drawBlock Success");
```

#### **4-2-4 Printing a Circle to the Image Buffer**

```

IResult = [_printer drawCircle:200
startPosY:200
sizeSelection:_SDK_DRAW_CIRCLE_SIZE_40X40
multiplier:1];

if(IResult == _SDK_RESULT_SUCCESS)
NSLog(@"drawTextDeviceFont Success");

```

#### **4-2-5 Printing a Text String with Device Font to the Image Buffer**

```

IResult = [_printer drawTextDeviceFont:@"device Font Test"
xPosition:70 yPosition:220
fontSelection:_SDK_DEVICE_FONT_12PT
fontWidth:1 fontHeight:1
rightSideCharacterSpacing:0
fontRotation:_SDK_ROTATION_DEGREES_0
reverse:NO bold:NO textAlignment:_SDK_ALIGNMENT_LEFT];
if(IResult == _SDK_RESULT_SUCCESS)
NSLog(@"drawTextDeviceFont Success");

```

#### **4-2-6 Printing a Text String with Vector Font to the Image Buffer**

```

IResult = [_printer drawTextVectorFont:@"draw Font Test - Vector"
xPosition:50 yPosition:100
fontSelection:_SDK_VECTOR_FONT_ASCII
fontWidth:10 fontHeight:10
rightSideCharacterSpacing:0
fontRotation:_SDK_ROTATION_DEGREES_0
reverse:NO bold:YES italic:YES
textWriteDirectionRightToLeft:NO textAlignment:_SDK_ALIGNMENT_LEFT];

if(IResult == _SDK_RESULT_SUCCESS)
NSLog(@"drawTextVectorFont Success");

```

## 4-2-7 Printing 1D Barcode to the Image Buffer

---

```

IResult = [_printer drawBarcode1D:@">A1234567890"
xPosition:50 yPosition:100
barcodeType:_SDK_BARCODE_TYPE_CODE128
widthNarrow:2 widthWide:6 height:100
hri:_SDK_BARCODE_HRI_BELOW_FONTSIZE1
quietZoneWidth:0
rotation:_SDK_ROTATION_DEGREES_0];

if(IResult == _SDK_RESULT_SUCCESS)
    NSLog(@"drawBarcode1D Success");

```

## 4-2-8 Printing PDF417 Barcode to the Image Buffer

---

```

IResult =[_printer drawBarcodePDF417:@"PDF417 Test"
xPosition:50 yPosition:300
maximumRowCount:30 maximumColumnCount:5
errorCorrectionLevel:_SDK_ERROR_CORRECTION_LEVEL0
dataCompressionMethod:_SDK_DATA_COMPRESSION_METHOD_TEXT
printBarcodeText:YES
barcodeOriginPoint:_SDK_BARCODE_ORIGIN_POINT_UpperLeft
moduleWidth:3 barHeight:10
rotation:_SDK_ROTATION_DEGREES_0];

if(IResult == _SDK_RESULT_SUCCESS)
    NSLog(@"drawBarcodePDF417 Success");

```

## 4-2-9 Printing QRCode Barcode to the Image Buffer

---

```

IResult =[_printer drawBarcodeQRCode:@"QR Code Test"
xPosition:450 yPosition:300
barcodeSize:3
model:_SDK_QRCODE_MODEL_2
errorColectionLevel:_SDK_ERROR_CORRECTION_LEVEL1
rotation:_SDK_ROTATION_DEGREES_0];

if(IResult == _SDK_RESULT_SUCCESS)
    NSLog(@"drawBarcodeQRCode Success");

```



## **4-2-10 Printing DataMatrix Barcode to the Image Buffer**

```
IResult =[_printer drawBarcodeDataMatrix:@"dataMatrix Test"  
xPosition:600 yPosition:300  
barcodeSize:3 reverse:NO  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeDataMatrix Success");
```

## **4-2-11 Printing MaxiCode Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodeMaxiCode:@"Maxi Code Test!!!"  
xPosition:50 yPosition:400  
mode:_SDK_MAXICODE_MODE_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@"drawBarcodeMaxiCode Success");
```

## **4-2-12 Printing Aztec Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodeAztec:@"Aztec Barcode Test"  
xPosition:450 yPosition:400  
barcodeSize:3  
extendedChannel:NO  
errorCorrectionLevel:0  
menuSymbol:NO numberOfSymbols:1  
optionalID:@""  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@"drawBarcodeAztec Success");
```

## 4-2-13 Printing Code49 Barcode to the Image Buffer

```
IResult = [_printer drawBarcodeCode49:@"Code 49 Test"  
xPosition:600 yPosition:400  
widthNarrow:1 widthWide:3  
height:100  
hri:_SDK_BARCODE_HRI_ABOVE_FONTSIZE1  
startingMode:_SDK_STARTINGMODE_AUTOMATIC  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeCode49 Success");
```

## 4-2-14 Printing Codablock Barcode to the Image Buffer

```
IResult = [_printer drawBarcodeIMB:@"01234567890123456789"  
xPosition:50 yPosition:1000  
printBarcodeText:YES  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeCodaBlock Success");
```

## 4-2-15 Printing MicroPDF417 Barcode to the Image Buffer

```
IResult = [_printer drawBarcodeMicroPDF:@"MICRO PDF 417 TEST"  
xPosition:50 yPosition:900  
moduleWidth:4  
barcodeHeight:4  
mode:8  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@"drawBarcodeMicroPDF Success");
```

## **4-2-16 Printing IMB Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodeIMB:@"01234567890123456789"  
xPosition:50 yPosition:1000  
printBarcodeText:YES  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeIMB Success");
```

## **4-2-17 Printing MSI Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodeMSI:@"123456"  
xPosition:50 yPosition:1100  
widthNarrow:2 widthWide:7  
height:100  
checkDigitSelection:_SDK_CHECKDIGIT_1MOD10  
printCheckDigitInHRI:YES  
hri:_SDK_BARCODE_HRI_BELOW_FONTSIZE1  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeMSI Success");
```

## **4-2-18 Printing Plessey Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodePlessey:@"12345"  
xPosition:100 yPosition:100  
widthNarrow:2 widthWide:7  
height:100 printCheckDigit:YES  
hri:_SDK_BARCODE_HRI_BELOW_FONTSIZE1  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodePlessey Success");
```

## **4-2-19 Printing TLC39 Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodeTLC39:@"123456,ABCD12345678901234"  
xPosition:100 yPosition:300  
widthNarrow:2 widthWide:7  
height:50 rowHeightOfMicroPDF417:3  
narrowWidthOfMicroPDF417:2  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeTLC39 Success");
```

## **4-2-20 Printing RSS Barcode to the Image Buffer**

```
IResult = [_printer drawBarcodeRSS:@"12345678901|this is composite info"  
xPosition:100 yPosition:600  
barcodeType:_SDK_RSS_BARCODE_TYPE_RSS14  
magnification:2  
separatorHeight:1 barcodeHeight:20 segmentWidth:10  
rotation:_SDK_ROTATION_DEGREES_0];  
  
if(IResult == _SDK_RESULT_SUCCESS)  
NSLog(@" drawBarcodeRSS Success");
```

## 4-3 Sample Application

### 4-3-1 Connect / Disconnect

1. Select to Interface
2. Input the MacAddress for target printer.
3. Connect to target printer.
4. Print to testPage.
5. Disconnect to targetPrinter.

※ Do not disconnect. If you want to use other Features.(print text, print Image, etc.)

### 4-3-2 Print to Device Font

1. Enter text to print.
2. X,Y coordinate of the position to print.
3. Horizontal/Vertical magnification of the selected font (1~4).
4. Requests the printing of text string to the output buffer using the Device Fonts
5. print the contents stored in the output buffer of the printer.

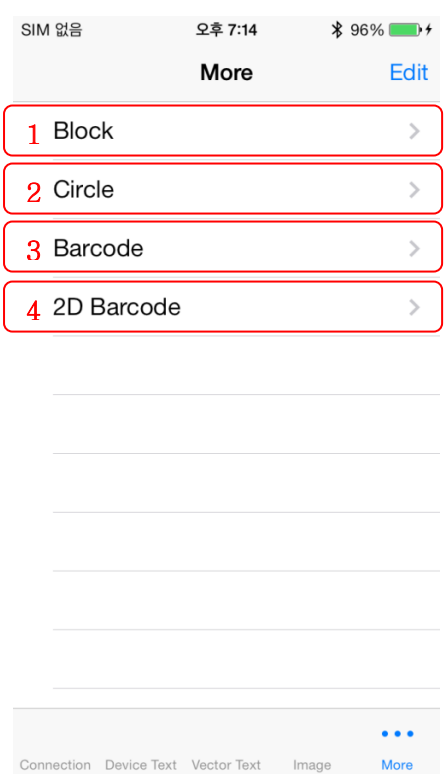
### 4-3-3 Print Vector Font

1. Enter text to print.
2. X,Y coordinate of the position to print.(1~4)
3. Horizontal/Vertical magnification of the selected font (1~4).
4. Requests the printing of text string to the output buffer using the Device Fonts
5. prints the contents stored in the output buffer of the printer.

### 4-3-4 Print Image

1. X,Y coordinate of the position to print.
2. Enter Width of Image.
3. Store into output buffer.
4. print the contents stored in the output buffer of the printer.

### 4-3-5 Use to more features.



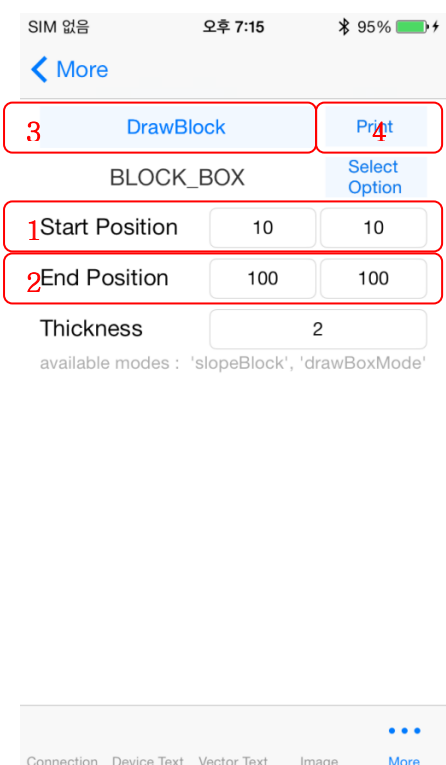
1. Block shapes can be used.

2. Circle shapes can be used.

3. 1D Barcodes can be used..

4. 2D Barcodes can be used.

### 4-3-6 Print Rectangle.



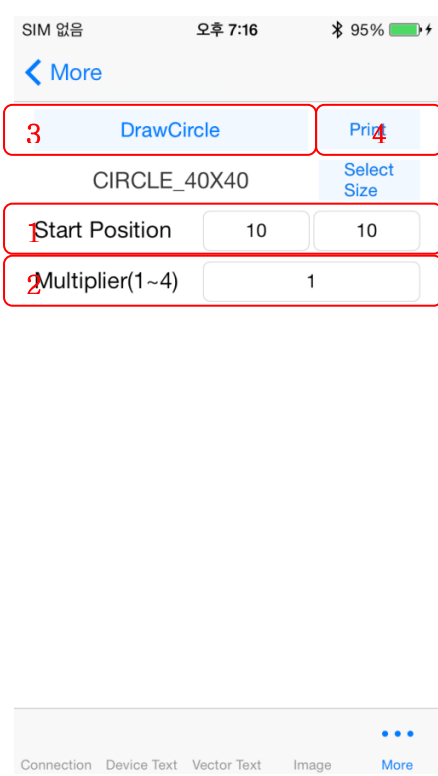
1. X,Y coordinate of the position to print.(Left, Top)

2. X,Y coordinate of the position to print.(Right, Bottom)

3. Store into output buffer.

4. Print the contents stored in the output buffer of the printer

### **4-3-7 Print Circle shapes.**



1. X,Y coordinate of the position to print.  
2. Enter size of circle.(1~4)  
3. Store into output buffer  
4. Print the contents stored in the output buffer of the printer.