

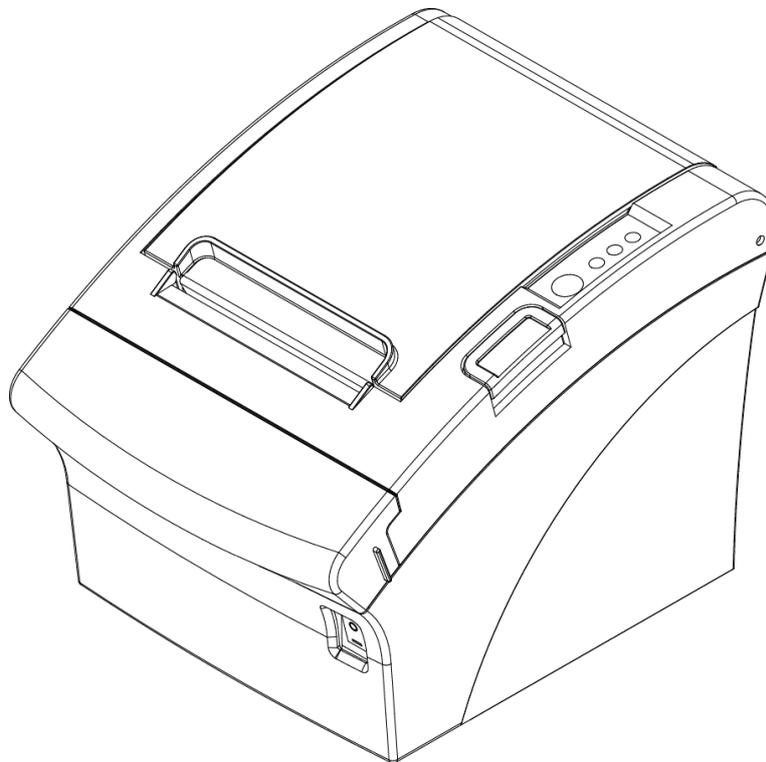
# METAPACE

## Software Manual **Metapace T-3II SDK for iOS**

---

Rev. 1.02

Metapace T-3II



<http://www.metapace.com>

## ■ Table of Contents

<b>1. About This Manual</b> .....	<b>5</b>
1-1 Supported Device .....	5
1-2 Supported Platform & Development Environment .....	5
1-3 List of Supported Printers / Interfaces.....	5
1-4 List of Supported Properties .....	5
1-5 List of Supported Methods.....	6
1-6 Setting SDK Project.....	7
1-6-1 Adding ExternalAccessory.framework .....	7
1-6-2 Adding Bluetooth Protocol .....	8
<b>2. Constants</b> .....	<b>9</b>
2-1 Defined Constants .....	9
2-1-1 Character Set .....	9
2-1-2 International Character Set.....	10
2-1-3 Text Encoding.....	10
2-1-4 Barcode/Image/Text Alignment .....	11
2-1-5 Text Size.....	11
2-1-6 Text Attribute .....	12
2-1-7 Barcode Text Position.....	12
2-1-8 Barcode Symbology .....	13
2-1-9 Image Width .....	14
2-1-10 Status Check Mask.....	14
2-1-11 Power .....	14
2-1-12 State .....	15
2-1-13 Connection Control.....	15
2-1-14 Drawer kick-out connector pin .....	15
2-1-15 Drawer open level.....	16
2-1-16 Model ID .....	16
2-1-17 Connection Class .....	16
2-1-18 Result Code.....	17
<b>3. BXBarcode Class Reference</b> .....	<b>18</b>
3-1 Overview.....	18
3-2 Properties .....	18
3-2-1 barNumber.....	18
3-2-2 name.....	18
3-2-3 support.....	18
<b>4. BXPrinter Class Reference</b> .....	<b>19</b>
4-1 Overview.....	19
4-2 Properties .....	19
4-2-1 name.....	19
4-2-2 address.....	19
4-2-3 port .....	19
4-2-4 modelStr .....	19
4-2-5 versionStr .....	20
4-2-6 connectionClass .....	20
<b>5. BXPrinterController Class Reference</b> .....	<b>21</b>
5-1 Overview.....	21

5-2 Properties .....	21
5-2-1 version .....	21
5-2-2 delegate .....	22
5-2-3 target .....	23
5-2-4 lookupDuration .....	24
5-2-5 lookupCount .....	25
5-2-6 alignment .....	26
5-2-7 attribute .....	27
5-2-8 textSize .....	28
5-2-9 characterSet .....	29
5-2-10 internationalCharacterSet .....	30
5-2-11 textEncoding .....	31
5-2-12 state .....	32
5-2-13 power .....	33
5-2-14 AutoConnection .....	34
5-2-15 drawerPin .....	34
5-2-16 drawerOpenLevel .....	34
5-3 Instance Methods .....	35
5-3-1 getInstance .....	35
5-3-2 open .....	36
5-3-3 close .....	36
5-3-4 lookup .....	37
5-3-5 selectTarget .....	39
5-3-6 connect .....	40
5-3-7 disconnect .....	41
5-3-8 enableLSB .....	42
5-3-9 printText .....	43
5-3-10 printBox .....	45
5-3-11 lineFeed .....	46
5-3-12 nextPrintPos .....	47
5-3-13 printBarcode .....	48
5-3-14 printBitmap .....	50
5-3-15 printBitmapWithImage .....	51
5-3-16 cutPaper .....	52
5-3-17 checkPrinter .....	53
5-3-18 directIO .....	54
5-3-19 nvImageList .....	55
5-3-20 downloadNVImage (Diffusion) .....	56
5-3-21 downloadNVImage (Normal) .....	58
5-3-22 printNVImage .....	59
5-3-23 removeNVImage .....	60
5-3-24 removeAllNVImages .....	61
5-3-25 openDrawer .....	61
5-3-26 isSupport_CashDrawer .....	62
5-3-27 isSupport_LSB .....	63
5-3-28 getBarcodeSupportTable .....	64
<b>6. BXPrinterControllerDelegate Protocol Reference .....</b>	<b>65</b>
6-1 Overview .....	65
6-2 Instance Methods .....	65
6-2-1 didStart .....	65
6-2-2 didStop .....	66

**Metapace T-3II SDK for iOS**

- 6-2-3 didFindPrinter .....66
- 6-2-4 didConnect .....67
- 6-2-5 didNotConnect.....68
- 6-2-6 willLookupPrinters .....68
- 6-2-7 didLookupPrinters.....69
- 6-2-8 didNotLookup .....69
- 6-2-9 didBeBrokenConnection .....70
- 6-2-10 didDisconnect .....71
- 6-2-11 didUpdateStatus .....72
  
- 7. Appendix .....73**
  - 7-1 Error Diffusion.....73

## 1. About This Manual

This SDK manual describes the contents of the library required to develop iOS application programs.

Metapace constantly makes improvements to the functions and quality of its products. The specifications and contents of this manual are subject to change without prior notice for this reason.

### 1-1 Supported Device

The devices in the following list are validated. iPod touch second generation or higher are compatible even though they are not listed here.

- iPhone 3GS / 4G / 5G / 5GS
- iPad / iPad2 / iPad mini

### 1-2 Supported Platform & Development Environment

- Platform
  - iOS 8.0 or higher
- Development Environment
  - XCode 10.2.1 higher

### 1-3 List of Supported Printers / Interfaces

Method/Property	Ethernet	Wi-Fi	Bluetooth
T-3II	○	○	○

### 1-4 List of Supported Properties

Property	POS Printer
Version	○
delegate	○
Target	○
lookupDuration	○
lookupCount	○
alignment	○
attribute	○
textSize	○
characterSet	○
internationalCharacterSet	○
State	○
Power	○
AutoConnection	○
drawerPin	○
drawerOpenLevel	○

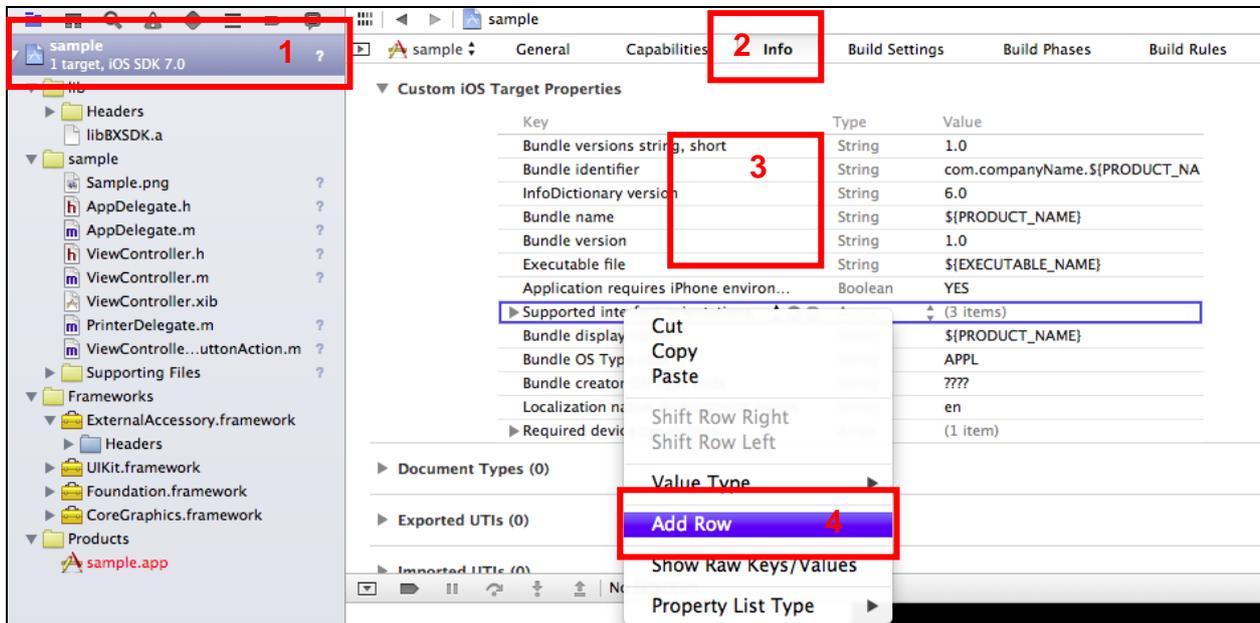
**1-5 List of Supported Methods**

	<b>Method</b>	<b>POS Printer</b>
General	<i>getInstance</i>	○
	<i>open</i>	○
	<i>close</i>	○
Search	<i>lookup</i>	○
	<i>willLookupPrinters</i>	○
	<i>didFindPrinter</i>	○
	<i>didLookupPrinters</i>	○
	<i>didNotLookup</i>	○
Connection	<i>selectTarget</i>	○
	<i>connect</i>	○
	<i>disconnect</i>	○
	<i>didConnect</i>	○
	<i>didDisconnect</i>	○
	<i>didNotConnect</i>	○
	<i>didBeBrokenConnection</i>	○
Status Check	<i>enableLSB</i>	○
	<i>checkPrinter</i>	○
	<i>isSupport_CashDrawer</i>	○
	<i>isSupport_LSB</i>	○
	<i>getBarcodeSupportTable</i>	○
Printings	<i>printText</i>	○
	<i>printBox</i>	○
	<i>lineFeed</i>	○
	<i>nextPrintPos</i>	○
	<i>printBarcode</i>	○
	<i>printBitmap</i>	○
	<i>printBitmapWithImage</i>	○
Direct IO	<i>directIO</i>	○
Cash Drawer	<i>openDrawer</i>	○

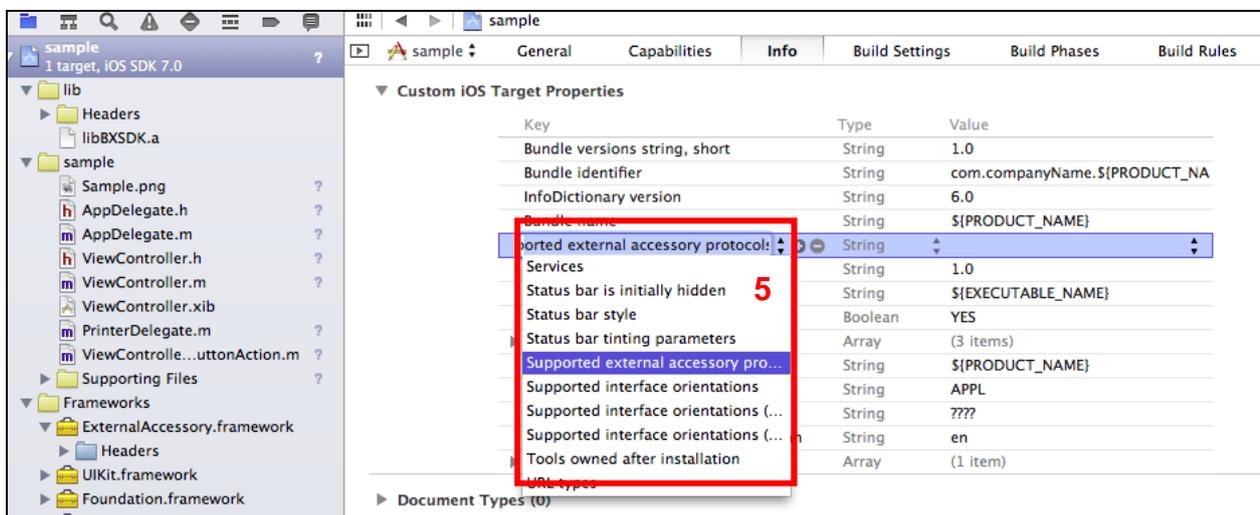
## 1-6 Setting SDK Project

[Note] Registration with the Apple Developer Program is required to develop iOS applications. Refer to the Apple Developer's Website (<http://developer.apple.com/devcenter/ios>) for details.

### 1-6-1 Adding ExternalAccessory.framework



- 1) Select project file
- 2) Select [Info] tab
- 3) Ctrl-click in the Area 3
- 4) Select [Add Row] from the pop up menu

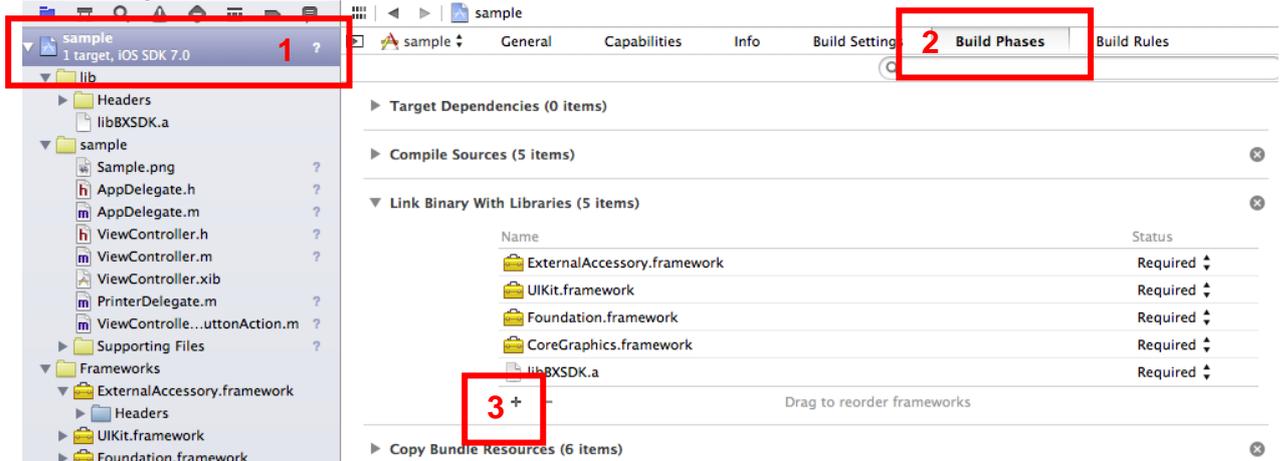


- 5) Enter "Supported external accessory protocols".
- 6) Enter "com.bixolon.protocol" in the Items field.

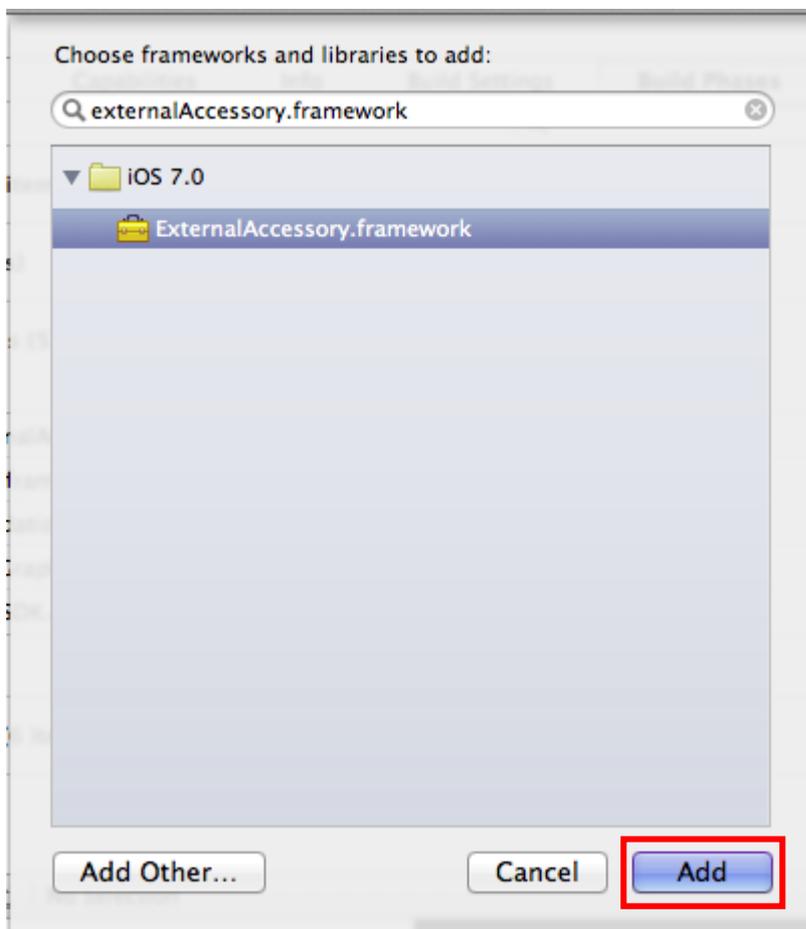
Supported external accessory protocols			Array	(1 item)
Item 0	+	-	String	com.bixolon.protocol
Bundle version			String	1.0
Executable file			String	\${EXECUTABLE_NAME}
Application requires iPhone environ...			Boolean	YES

## 1-6-2 Adding Bluetooth Protocol

1) Select Project, Build Phases, and + in the order shown below.



2) Search “ExternalAccessory.framework” and click the [Add] to add it.



## 2. Constants

Constants used by the provided SDK are defined in the “BXCode.h” file.

### 2-1 Defined Constants

#### 2-1-1 Character Set

This is an attribute to define code page, and the default value is set to BXL\_CS\_437.

The following options are available for the code page.

Code	Value	Description
BXL_CS_PC437	0	Code page PC437
BXL_CS_Katakana	1	Katakana
BXL_CS_PC850	2	Code page PC850
BXL_CS_PC860	3	Code page PC860
BXL_CS_PC863	4	Code page PC863
BXL_CS_PC865	5	Code page PC860
BXL_CS_WPC1252	16	Code page WPC1252
BXL_CS_PC866	17	Code page PC860
BXL_CS_PC852	18	Code page PC852
BXL_CS_PC858	19	Code page PC858
BXL_CS_PC864	22	Code page PC864
BXL_CS_THAI42	23	Code page THAI42
BXL_CS_WPC1253	24	Code page WPC1253
BXL_CS_WPC1254	25	Code page WPC1254
BXL_CS_WPC1257	26	Code page WPC1257
BXL_CS_FARSI	27	Code page FARSI
BXL_CS_WPC1251	28	Code page WPC1251
BXL_CS_PC737	29	Code page PC737
BXL_CS_PC775	30	Code page PC775
BXL_CS_THAI14	31	Code page THAI14
BXL_CS_PC862	33	Code page PC862
BXL_CS_PC855	36	Code page PC855
BXL_CS_PC857	37	Code page PC857
BXL_CS_PC928	38	Code page PC928
BXL_CS_THAI16	39	Code page THAI16
BXL_CS_WPC1256	40	Code page WPC1256

## 2-1-2 International Character Set

This is an attribute to define international character set and the default value is set to BXL\_ICS\_USA.

The following options are available for the International Character Set.

Code	Value	Description
BXL_ICS_USA	0	U.S.A.
BXL_ICS_FRANCE	1	France
BXL_ICS_GERMANY	2	Germany
BXL_ICS_UK	3	U.K.
BXL_ICS_DENMARK1	4	Denmark /
BXL_ICS_SWEDEN	5	Sweden
BXL_ICS_ITALY	6	Italy
BXL_ICS_SPAIN	7	Spain
BXL_ICS_NORWAY	9	Norway
BXL_ICS_DENMARK2	10	Denmark //

## 2-1-3 Text Encoding

This is an attribute to define Text Encoding type and the default value is set to BXL\_TEXTENCODING\_SINGLEBYTEFONT.

The following options are available for Text Encoding type.

Code	Value	Description
BXL_TEXTENCODING_SINGLEBYTEFONT	0x00000001	Default. Use Single byte font.
BXL_TEXTENCODING_KSC5601	0x80000422	Korean encoding
BXL_TEXTENCODING_SHIFT_JIS	0x00000008	Japanese encoding
BXL_TEXTENCODING_BIG5	0x80000a03	Chinese encoding-BIG5
BXL_TEXTENCODING_GB2312	0x80000421	Chinese encoding-GB2312
BXL_TEXTENCODING_GB18030	0x80000632	Chinese encoding-GB18030

Refer to NSString.h, NSStringEncoding to see more lists of encoding type.

## 2-1-4 Barcode/Image/Text Alignment

This is an attribute to define alignment of barcode/image/text, and the default value is set to BXL\_ALIGNMENT\_LEFT.

The following options are available for this value.

Code	Value	Description
BXL_ALIGNMENT_LEFT	0	Align to Left
BXL_ALIGNMENT_CENTER	1	Align to Center
BXL_ALIGNMENT_RIGHT	2	Align to Right

## 2-1-5 Text Size

This is an attribute to set the size of text, and the height and width can be set simultaneously using bitwise OR combination.

The following options are available.

### <Width Attribute>

Code	Value	Description
BXL_TS_0WIDTH	0	Width magnification factor is set to X1.
BXL_TS_1WIDTH	16	Width magnification factor is set to X2.
BXL_TS_2WIDTH	32	Width magnification factor is set to X3.
BXL_TS_3WIDTH	48	Width magnification factor is set to X4.
BXL_TS_4WIDTH	64	Width magnification factor is set to X5.
BXL_TS_5WIDTH	80	Width magnification factor is set to X6.
BXL_TS_6WIDTH	96	Width magnification factor is set to X7.
BXL_TS_7WIDTH	112	Width magnification factor is set to X8.

### <Height Attribute >

Code	Value	Description
BXL_TS_0HEIGHT	0	Height magnification factor is set to X1.
BXL_TS_1HEIGHT	16	Height magnification factor is set to X2.
BXL_TS_2HEIGHT	32	Height magnification factor is set to X3.
BXL_TS_3HEIGHT	48	Height magnification factor is set to X4.
BXL_TS_4HEIGHT	64	Height magnification factor is set to X5.
BXL_TS_5HEIGHT	80	Height magnification factor is set to X6.
BXL_TS_6HEIGHT	96	Height magnification factor is set to X7.
BXL_TS_7HEIGHT	112	Height magnification factor is set to X8.

## 2-1-6 Text Attribute

This attribute is to set the text attribute and each attribute can be combined by bitwise OR operation.

The following options are available for this attribute.

Code	Value	Description
BXL_FT_DEFAULT	0	Default attribute NOT BOLD, FONTA, NOT UNDERLINE, NOT REVERSE
BXL_FT_FONTB	1	Use FONTB.
BXL_FT_FONTC	16	Use FONTC.
BXL_FT_BOLD	2	Use Bold attribute.
BXL_FT_UNDERLINE	4	Use Underline attribute.
BXL_FT_REVERSE	8	Use Reverse attribute.

## 2-1-7 Barcode Text Position

It sets the position to print barcode data.

The following options are available.

Code	Value	Description
BXL_BC_TEXT_NONE	0	Barcode data is not printed.
BXL_BC_TEXT_ABOVE	1	Barcode data is printed above the barcode.
BXL_BC_TEXT_BELOW	2	Barcode data is printed below the barcode.

## 2-1-8 Barcode Symbology

This attribute defines the barcode type.

The following options are available.

Code	Value	Number of Data	Range of Data
BXL_BCS_UPCA	101	11 <= n <= 12	48 <= data <= 57
BXL_BCS_UPCE	102	11 <= n <= 12	48 <= data <= 57
BXL_BCS_EAN13	103	12 <= n <= 13	48 <= data <= 47
BXL_BCS_JAN13	104	7 <= n <= 8	48 <= data <= 57 64 <= data <= 90 data = 32,36,37,43,45,46,47
BXL_BCS_EAN8	105	7 <= n <= 8	48 <= data <= 57
BXL_BCS_JAN8	106	7 <= n <= 8	48 <= data <= 57
BXL_BCS_Code39	107	1 <= n <= 255	48 <= data <= 57 65 <= data <= 68 data = 32,36,37,43,45,46,47
BXL_BCS_ITF	108	1 <= n <= 255 (even number)	48 <= data <= 57
BXL_BCS_Codabar	109	1 <= n <= 255	48 <= data <= 57 65 <= data <= 68 data = 36,43,45,46,47,58
BXL_BCS_Code93	110	1 <= n <= 255	0 <= data <= 127
BXL_BCS_Code128	111	2 <= n <= 255	0 <= data <= 127
BXL_BCS_PDF417	200	2 <= n <= 928	0 <= data <= 255
BXL_BCS_QRCODE	202~203	2 <= n <= 928	0 <= data <= 255
BXL_BCS_DATAMATRIX	204	2 <= n <= 928	0 <= data <= 255
BXL_BCS_MAXICODE	205~6	2 <= n <= 928	0 <= data <= 255

## 2-1-9 Image Width

This attribute sets the width of image. The range of this value can be from 0 to maximum width of printer.

Values set as in the following table changes the image size accordingly as described in the table.

The following options are available for this attribute.

Code	Value	Description
BXL_WIDTH_FULL	-1	Image width is set to the maximum width and converted to the maximum width of given paper.
BXL_WIDTH_NONE	-2	Image size is not changed.

## 2-1-10 Status Check Mask

This attribute is to set the range of printer status for status checking. It is used as a parameter of checkPrinter method.

The following options are available.

Code	Value	Description
BXL_MASK_COVERSTATUS	1	Check the status of printer cover.
BXL_MASK_PAPERSTATUS	2	Check the status of printer paper.
BXL_MASK_POWERSTATUS	4	Check the status of printer power.
BXL_MASK_MODELNAME	8	Check the model name of printer.
BXL_MASK_VERSION	16	Check the firmware version of printer.
BXL_MASK_CASHDRAWER	32	Check the status of Cash Drawer if it is connected to printer.
BXL_MASK_ALL	0XFF	Check all of above status.

## 2-1-11 Power

This attribute indicates the battery status. It is a read-only parameter and the value is updated whenever the status of printer battery changes.

The battery status is reported as follows.

Code	Value	Description
BXL_PWR_HIGH	0	The battery is 95% charged.
BXL_PWR_MIDDLE	1	The battery is 85% charged.
BXL_PWR_LOW	2	The battery is 50% charged.
BXL_PWR_SMALL	3	The battery is 25% charged.
BXL_PWR_NOT	4	The battery is less than 25% charged.

## 2-1-12 State

This attribute is to get the printer status. It is read-only and printer status is returned in this attribute when calling CheckPrinter function. Each setting can be set multiple times and individual status can be extracted through bit operation.

Printer status values are as follows.

Code	Value	Description
BXL_STS_NORMAL	0	Normal state
BXL_STS_PAPEREMPTY	1	No paper
BXL_STS_COVEROPEN	2	Printer cover is open.
BXL_STS_POWEROVER	4	Battery is low.
BXL_STS_MSR_READY	8	Printer is not ready. It is in MSR read mode.
BXL_STS_PRINTING	16	Printer is printing / exchanging data.
BXL_STS_ERROR	32	There is an error in communication with printer.
BXL_STS_NOT_OPEN	64	The "open" method on BXPrinterControl has not been called.
BXL_STS_ERROR_OCCUR	128	There is an error in the printer.
BXL_STS_NOT_CONNECTED	-1	Printer is currently disconnected.

## 2-1-13 Connection Control

This is for setting the method of printer connection.

The available options are as follows.

Code	Value	Description
BXL_CONNECTIONMODE_AUTO	0	Automatic connection mode
BXL_CONNECTIONMODE_NOAUTO	100	Manual connection mode

## 2-1-14 Drawer kick-out connector pin

This is for setting a pin number to be used for connecting cash drawer.

The available options are as follows.

Code	Value	Description
BXL_CASHDRAWER_PIN_2	0	Cash drawer connection pin: 2
BXL_CASHDRAWER_PIN_5	1	Cash drawer connection pin: 5

## 2-1-15 Drawer open level

This is for setting cash drawer type.

The following options are available.

Code	Value	Description
BXL_CASHDRAWER_OPENLEVEL_LOW	0	Low when cash drawer is open
BXL_CASHDRAWER_OPENLEVEL_HIGH	1	High when cash drawer is open

## 2-1-16 Model ID

This is for setting the printer type.

The following options are available.

Code	Value	Description
BXL_MODEL_ID_T3II	0x23500003	T-3II

## 2-1-17 Connection Class

This is for defining the connection type of the printer.

When 'didFindPrinter' function is called, the connectionClass of BXPrinter class is updated.

Code	Value	Description
BXL_CONNECTIONCLASS_WIFI	0x0000	Wi-Fi Connection
BXL_CONNECTIONCLASS_ETHERNET	0x0001	Ethernet Connection
BXL_CONNECTIONCLASS_BLUETOOTH	0x0002	Bluetooth Connection

## Metapace T-3II SDK for iOS

### 2-1-18 Result Code

This is for defining the result returned after performing specific functions by various methods.

<b>Code DEFINE</b>	<b>Value</b>	<b>Description</b>
BXL_SUCCESS	0	Success
BXL_NOT_CONNECTED	-1	Printer is not connected.
BXL_NOT_OPENED	101	SDK is not open.
BXL_STATUS_ERROR	103	There is an error during status check.
BXL_CONNECT_ERROR	105	Connection error
BXL_NOT_SUPPORT	107	Not supported
BXL_BAD_ARGUMENT	108	Wrong function argument
BXL_BUFFER_ERROR	109	Error in MSR buffer
BXL_NOT_CONNECTED	110	Printer is not connected
BXL_RGBA_ERROR	111	Error in converting image file to RGBA data
BXL_MEMORY_ERROR	112	Memory allocation error
BXL_TOO_LARGE_IMAGE	113	Image file to download NV area is too big
BXL_NOT_SUPPORT_DEVICE	114	Not supported by the printer.
BXL_READ_ERROR	301	Error in data reception
BXL_WRITE_ERROR	300	Error in data transmission
BXL_BITMAPLOAD_ERROR	400	Error in reading image file
BXL_BC_DATA_ERROR	500	Error in bar code data
BXL_BC_NOT_SUPPORT	501	Unsupported barcode type
BXLMSR_NOTREADY	602	MSR is not ready.
BXLMSR_FAILEDMODE	601	Automatic read mode is not set.
BXLMSR_DATAEMPTY	603	There is no data read from MSR.

## 3. BXBarcode Class Reference

### 3-1 Overview

BXBarcode Class is an object that contains the information of supported barcode types of target printer.

### 3-2 Properties

#### 3-2-1 barNumber

Barcode Define Number

@property int barNumber

#### **[Discussion]**

It is saved automatically with the information obtained from the connected printer.

#### **[See Also]**

[2-1-8 Barcode Symbology](#)

#### 3-2-2 name

Barcode name

@property NSString \* barName

#### **[Discussion]**

It is saved automatically with the information obtained from the connected printer.

#### 3-2-3 support

Barcode support

@property BOOL support

#### **[Discussion]**

It is saved automatically with the information obtained from the connected printer.

## 4. BXPrinter Class Reference

### 4-1 Overview

BXPrinter Class is an object that contains the information of target printer (name / network address/port).

### 4-2 Properties

#### 4-2-1 name

Printer name

@property(readonly) NSString \* name

#### **[Discussion]**

Name is saved automatically with the information obtained from the connected printer.

#### 4-2-2 address

Printer network address

@property(readwrite) NSString \* address

#### **[Discussion]**

Network address of the printer should be assigned before making connection.

#### 4-2-3 port

Network port of printer

@property(readwrite) unsigned short port

#### **[Discussion]**

Network port of printer should be assigned before connection.

#### 4-2-4 modelStr

Printer model name

It is a model name provided by firmware - \_T-3II in case of T-3II printer.

@property(readwrite) NSString \* modelStr

#### **[Discussion]**

This value is updated by the checkPrinter method of BXPrinterController.

#### **[See Also]**

[5-3-17 checkPrinter](#)

### 4-2-5 versionStr

Firmware version of printer

It is a version name provided by firmware and is provided in the format like V01.00 STOB 040711.

@property(readwrite) NSString \* versionStr

#### **[Discussion]**

This value is updated by the checkPrinter method of BXPrinterController.

#### **[See Also]**

[5-3-17 checkPrinter](#)

### 4-2-6 connectionClass

Printer connection type

Information about network connection is saved in this object.

@property(readwrite) unsigned short \* connectionClass

#### **[See Also]**

[2-1-15 Connection Class](#)

## 5. BXPrinterController Class Reference

### 5-1 Overview

BXPrinterController Class is a main object for printer control.

### 5-2 Properties

#### 5-2-1 version

SDK version

@property(readonly) NSString \* version

#### [Discussion]

It is a character string with a format like “1.0.0” and it is a read-only variable.

#### [See Also]

[5-3-17 checkPrinter](#)

#### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController == nil)  
    NSLog (@”getInstance Fail.”);  
  
NSLog(@”SDK Version : %@”, pController.version);
```

## 5-2-2 delegate

It is an object to delegate BXPriinterControlDelegate method

```
@property(readwrite) id<BXPriinterControlDelegate> delegate
```

### [Example]

```
@interface delegateTestClass : NSObject<BXPriinterControllerDelegate>
{
}
@end
```

Declaration

```
@implementation TestClass
// process
-(void) printerInitialize
{
    BXPriinterControler * pController = [BXPriinterController getInstance];

    If(pController == nil)
        NSLog (@"getInstance Fail.");

    pController.delegate = self;
}
//process
@end
```

Implementation

### 5-2-3 target

It is a target control printer object.

```
@property(readwrite) BXPrinter * target
```

#### [Discussion]

This object for target control printer should be assigned before controlling printer.

#### [See Also].

[5-3-5 selectTarget](#)

#### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

[pController Open];

// printer object searched by using lookup function can also be used.
BXPrinter* printer = [BXPrinter new];
printer.connectionClass=BXL_CONNECTIONCLASS_BT;
printer.macAddress = @"XX:XX:XX:XX:XX:XX";

pController.target = printer;
If(BXL_SUCCESS == [pController selectTarget])
{
    NSLog(@"Select Target Success");
}
// ...
[printer release];
//...
```

## 5-2-4 lookupDuration

Printer search period (unit: second)

@property(readwrite) CGFloat lookupDuration

### [Discussion]

Fractional number like 0.5 can also be used.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
pController.lookupDuration = 2.0;
```

```
pController.lookupCount = 5;
```

```
[pController lookup]
```

### 5-2-5 lookupCount

Number of repetitions in sending signal for printer search

@property(readwrite) unsigned lookupCount

#### [Discussion]

Default value is set to 1. When it is higher than 1, printer search signal is repeated by the specified number in 0.2 second units.

#### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
pController.lookupDuration = 2.0;
```

```
pController.lookupCount = 5;
```

```
[pController lookup]
```

## 5-2-6 alignment

Horizontal alignment setting

```
@property(readwrite) int          alignment
```

### [Discussion]

Default value is Left Align, and this setting affects all print settings including text and barcode.

### [See Also]

[2-1-4 Barcode/Image/Text Alignment](#)

[5-3-9 printText](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

pController.alignment = BXL_ALIGNMENT_LEFT;
If(BXL_SUCCESS == [pController printText:@"This is alignment Test - left"])
    NSLog(@"printText Success");

pController.alignment = BXL_ALIGNMENT_CENTER;
If(BXL_SUCCESS == [pController printText:@"This is alignment Test - center"])
    NSLog(@"printText Success");

pController.alignment = BXL_ALIGNMENT_RIGHT;
If(BXL_SUCCESS == [pController printText:@"This is alignment Test - right"])
    NSLog(@"printText Success");
```

## 5-2-7 attribute

Text output attribute

@property(readwrite) int attribute

### [See Also]

[2-1-6 Text Attribute](#)

[5-3-9 printText](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
pController.attribute = BXL_ALIGNMENT_LEFT;
```

```
If(BXL_SUCCESS == [pController printText:@"This is alignment Test - left"])  
    NSLog(@"printText Success");
```

```
pController.alignment = BXL_ALIGNMENT_CENTER;
```

```
If(BXL_SUCCESS == [pController printText:@"This is alignment Test - center"])  
    NSLog(@"printText Success");
```

```
pController.alignment = BXL_ALIGNMENT_RIGHT;
```

```
If(BXL_SUCCESS == [pController printText:@"This is alignment Test - right"])  
    NSLog(@"printText Success");
```

## 5-2-8 textSize

Text size to be printed

@property(readwrite) int                      textSize

### [See Also]

[2-1-5 Text Size](#)

[5-3-9 printText](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");



pController.textSize = BXL_TS_0WIDTH| BXL_TS_1HEIGHT;  
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 0x0-default"])  
    NSLog(@"printText Success");



pController.textSize = BXL_TS_1WIDTH| BXL_TS_1HEIGHT;  
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 1x1"])  
    NSLog(@"printText Success");



pController.textSize = BXL_TS_2WIDTH| BXL_TS_2HEIGHT;  
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 2x2"])  
    NSLog(@"printText Success");



pController.textSize = BXL_TS_2WIDTH| BXL_TS_4HEIGHT;  
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 2x4"])  
    NSLog(@"printText Success");



pController.textSize = BXL_TS_7WIDTH| BXL_TS_7HEIGHT;  
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 7x7"])  
    NSLog(@"printText Success");


```

## 5-2-9 characterSet

Attribute to define the code page of printer

@property(readwrite) char characterSet

### [Discussion]

Default value is set to BXL\_CS\_437.

### [See Also]

[2-1-1 Character Set](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");



pController.characterSet = BXL_CS_437;
If(BXL_SUCCESS == [pController printText:@" "])
    NSLog(@"printText Success");



pController.textSize = BXL_TS_1WIDTH| BXL_TS_1HEIGHT;
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 1x1"])
    NSLog(@"printText Success");



pController.textSize = BXL_TS_2WIDTH| BXL_TS_2HEIGHT;
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 2x2"])
    NSLog(@"printText Success");



pController.textSize = BXL_TS_2WIDTH| BXL_TS_4HEIGHT;
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 2x4"])
    NSLog(@"printText Success");



pController.textSize = BXL_TS_7WIDTH| BXL_TS_7HEIGHT;
If(BXL_SUCCESS == [pController printText:@"This is textSizeTest 7x7"])
    NSLog(@"printText Success");


```

## 5-2-10 internationalCharacterSet

Attribute to define the code page of special character area of printer

@property(readwrite) char internationalCharacterSet

### [Discussion]

Refer to 2-2 International Character Set. Default value is set to BXL\_ICS\_USA.

### [See Also]

[2-1-2 internationalCharacter Set](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
  
pController.characterSet = BXL_CS_437;  
If(BXL_SUCCESS == [pController printText:@" "])  
    NSLog(@"printText Success");
```

## 5-2-11 textEncoding

Attribute for setting Text Encoding type.

@property(readwrite) long textEncoding

### [Discussion]

The characterSet is extended code set that can be used in one byte font. Separate text encoding may be required in case of two byte font and some character sets.

### [See Also]

[2-1-3 Text Encoding](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
pController.textEncoding  
    = BXL_TEXTENCODING_KSC5601;
```

```
If(BXL_SUCCESS == [pController printText:@" "])  
    NSLog(@"한국어 출력 성공");
```

KSC5601

```
pController.textEncoding  
    = BXL_TEXTENCODING_SHIFT_JIS;
```

```
If(BXL_SUCCESS == [pController printText:@" "])  
    NSLog(@"にほんご出力が 完了しました。");
```

Shift-JIS

```
pController.textEncoding  
    = BXL_TEXTENCODING_GB18030;
```

```
If(BXL_SUCCESS == [pController printText:@" "])  
    NSLog(@"完成中國輸出");
```

GB18030

## 5-2-12 state

Printer status code

This value is updated when checkPrinter method of BXPrinterController is called.

@property(readonly) long state

### [See Also]

[2-1-12 State](#)

[5-3-17 checkPrinter](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
If (BXL_SUCCESS==  
    [pController checkStatus:BXL_MASK_COVEROPEN ] )  
{  
    If(pController.state & BXL_STS_COVEROPEN)  
        NSLog(@"printer Cover is open");  
}
```

Check cover  
status only

```
If (BXL_SUCCESS==  
    [pController checkStatus:BXL_MASK_All ] )  
{  
    If(pController.state & BXL_STS_COVEROPEN)  
        NSLog(@"printer Cover is open");  
  
    If(pController.state & BXL_STS_PAPEREMPTY)  
        NSLog(@"Paper is empty.");  
}
```

Check all status

## 5-2-13 power

Power status of printer

This value is updated when checkPrinter method of BXPrinterController is called.

@property(readonly) long state

### [Discussion]

BXL\_PWR\_HIGH is returned in case of models that do not use a battery.

### [See Also]

[2-1-11 Power](#)

[5-3-17 checkPrinter](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

If (BXL_SUCCESS==
    [pController checkStatus:BXL_MASK_POWERSTATUS] )
{
    switch(pController.power)
    {
        case BXL_PWR_HIGH:
            NSLog(@"Power Status is High");
            break;

        case BXL_PWR_MIDDLE:
            NSLog(@"Power Status is Middle");
            break;

        case BXL_PWR_LOW:
            NSLog(@"Power Status is Low");
            break;
    }
}
```

## 5-2-14 AutoConnection

Options on printer connection

@property(assign) int AutoConnection

### **[Discussion]**

When this attribute is set to BXL\_CONNECTIONMODE\_AUTO, the [5-3-6 connect](#) / [5-3-7 disconnect](#) methods cannot be used as the connection will be automatic. Print speed may become slower if connect and disconnect process are repeated, and this function is not supported in Bluetooth mode.

### **[See Also]**

[2-1-13 Connection Control](#)

[5-3-6 connect](#)

[5-3-7 disconnect](#)

## 5-2-15 drawerPin

Attribute of pin of Cash Drawer

@property(assign) int drawerPin

### **[Discussion]**

Check the attribute of pin as Cash Drawer may not work if pin attribute setting is incorrect.

### **[See Also]**

[2-1-14 Drawer kick-out connector pin](#)

## 5-2-16 drawerOpenLevel

Level to recognize Open of Cash Drawer

@property(assign) int drawerOpenLevel

### **[Discussion]**

Incorrect Open Level setting may reverse the Open/Close state of Cash Drawer. Check Open Level in this case.

### **[See Also]**

[2-1-15 Drawer open level](#)

## 5-3 Instance Methods

### 5-3-1 getInstance

This is a method to get the instance of BXPrinterController class.

#### [Syntax]

- (BXPrinterController)getInstance

#### [Return Value]

BXPrinterController class is created automatically and returned when this method is called first time, and existing BXPrinterController class is returned when this method is called again.

#### [Discussion]

Since BXPrinterController class uses only one instance in a process, create and use the class through this method instead of creating it directly by user.

#### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController == nil)  
    NSLog (@”getInstance Fail.”);
```

## 5-3-2 open

This method initializes various variables for using BXPrinterController class (memory allocation and background thread operation)

### [Syntax]

- (void)open

### [Discussion]

This should be called before calling main delegate of applications like (void)applicationDidBecomeActive:(UIApplication \*)application.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController == nil)  
    NSLog (@"getInstance Fail.");  
  
[pController open];
```

## 5-3-3 close

This method is used to release the resources allocated for using the BXPrinterController class.

### [Discussion]

This should be called before calling main delegate of applications like (void)applicationDidBecomeActive:(UIApplication \*)application.

If applications using BXPrinterController run in background mode without calling this method, use of BXPrinterController at the same time by other applications can be restricted.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController == nil)  
    NSLog (@"getInstance Fail.");  
  
[pController Open];  
  
//Do Something  
  
[pController close];
```

### 5-3-4 lookup

This method searches the printers in the same network as the iPhone is connected.

#### **[Syntax]**

- (void)lookup

#### **[Discussion]**

Start/end of search and list of searched printers can be obtained from `BXPrinterControlDelegate`. In case of iPhone, both 3G and Wi-Fi have two network adapters and lookup method performs search operation using only Wi-Fi. It has no effect if there is no connected Wi-Fi network.

#### **[See Also]**

[6-2-3 didFindPrinter](#)

[6-2-6 willLookupPrinters](#)

[6-2-7 didLookupPrinters](#)

[6-2-8 didNotLookup](#)

## [Example]

```
- (IBAction) buttonUp_Lookup:(id)sender
{
    NSLog(@" Lookup / Targetting Button Up.");
    BXPrinterController* pController = [BXPrinterController getInstance];

    If(pController != nil)
        NSLog (@"getInstance Fail.");
    [pController lookup];
}
```

```
// The following function is called when starting printer search.
- (void)willLookupPrinters:(BXPrinterController *)controller
{
    NSLog("will Lookup Printers.");
}
```

Search Start

```
// The following function is called when printer search operation is completed.
- (void)didLookupPrinters:(BXPrinterController *)controller
{
    NSLog("printer Lookup Finish.");
}
```

Search Completed

```
// This function is called every time printer is searched.
- (void)didFindPrinter:(BXPrinterController *)controller printer:(BXPrinter*) printer
{
    NSLog("printer Find.");
}
```

Searching

```
// This function is called every time printer is searched.
- (void)didNotLookup:(BXPrinterController *)controller withError:(NSError*) error
{
    NSLog("didNotLookup.");
}
```

Search  
Fail

## 5-3-5 selectTarget

This method is to initialize the object specified as target.

### [Syntax]

- (long)selectTarget
- (long)selectTarget : (int) modelID

### [Parameters]

*modelID*

- Select model type.
- It is assigned automatically if not specified. Refer to 2-14 Model ID.

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Discussion]

Target of BXPrinterController should be assigned before calling this method.

### [See Also]

[5-2-3 target](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

[pController Open];

// Object of searched printer can be used by using lookup function.
BXPrinter* printer = [BXPrinter new];
printer.connectionClass=BXL_CONNECTIONCLASS_BT;
printer.macAddress = @"XX:XX:XX:XX:XX:XX";

pController.target = printer;
If(BXL_SUCCESS == [pController selectTarget])
{
    NSLog(@"Select Target Success");
}
// ...
[printer release];
//...
```

## 5-3-6 connect

This method connects the target printer.

### [Syntax]

- (BOOL)connect

### [Discussion]

It has no effect if AutoConnection variable is set to BXL\_CONNECTIONMODE\_AUTO (default, == 0).

The target of BXPrinterController should be assigned before calling this method.

### [See Also]

[6-2-4 didConnect](#)

[6-2-5 didNotConnect](#)

[6-2-9 didBeBrokenConnection](#)

### [Return Value]

YES if the operation is successful. The return values contain the results of the Connect operation.

Refer to the description of delegate function to check the results.

### [Example]

```
- (void ) buttonUp_Disconnect:(id)sender
{
    BXPrinterController* pController = [BXPrinterController getInstance];

    If(pController != nil)
        NSLog (@”getInstance Fail.”);

    If(BXL_SUCCESS == [pController connect])
        NSLog(@”Connect Request is Success.”);
}
```

// The following function is called when “connect” operation is successful.

```
- (void)didConnect:(BXPrinterController *)controller printer:(BXPrinter*) printer
{
    NSLog(“Connect Complete”);
}
```

SUCCESS

// The following function is called when “connect” operation fails.

```
- (void)didNotConnect:(BXPrinterController *)controller withError:(NSError*) error
{
    NSLog(“Connect Fail”);
}
```

FAIL

## 5-3-7 disconnect

This method disconnects the connection with the printer.

### [Syntax]

- (void)disconnect

### [Discussion]

This method has no effect if AutoConnection variable is set to BXL\_CONNECTIONMODE\_AUTO (default, == 0).

### [See Also]

The following delegate function is called when disconnect operation is completed.

[6-2-10 didDisconnect](#)

### [Example]

```
- (void ) buttonUp_Disconnect:(id)sender
{
    // process
    [pController disconnect];
    // process
}
```

// The following function is called when "Disconnect" operation is completed.

```
- (void)didDisconnect:(BXPrinterController *)controller printer:(BXPrinter*) printer
{
    NSLog("Disconnect Complete");
}
```

**disconnect  
Completed**

## 5-3-8 enableLSB

This method activates the Last Status Back function.

### [Syntax]

- (long)enableLSB:(BOOL)bEnable

### [Parameters]

*bEnable*

LSB setting.

FALSE: LSB Disable

TRUE: LSB Enable

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [See Also]

[2-1-12 State](#)

[6-2-11 didUpdateStatus](#)

### [Example]

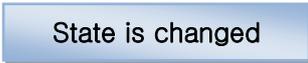
```
- (void) enableLSBTest
{
    // Enable
    If( BXL_SUCCESS == [pController enableLSB:YES];)
        NSLog(@"enableLSB:YES Success" );

    // disable
    If(BXL_SUCCESS == [pController enableLSB:NO])
        NSLog(@"enableLSB:NO Success");
}
```

// The following function is called when state is changed.

```
- (void)didUpdateStatus:(BXPrinterController *)controller status:(NSNumber*)status
{
    NSLog("did Update Status");

    If(status & BXL_STS_NORMAL)
        NSLog(@"Printer Status is Normal.");
    If(status & BXL_STS_PAPEREMPTY)
        NSLog(@"Paper is empty.");
    If(status & BXL_STS_COVEROPEN)
        NSLog(@"Printer Status is Normal.");
}
```



### 5-3-9 printText

This method prints text.  
It has no effect if there is no connected printer.

#### **[Syntax]**

(long)printText:(NSString \*)string

#### **[Parameters]**

*string*

Target text string, unicode data with null as terminator

#### **[Return Value]**

Refer to 2-1-18 Result Code

#### **[Discussion]**

Alignment and attributes of text should be assigned before calling this method.

#### **[See Also]**

[2-1-1 Character Set](#)

[2-1-2 internationalCharacterSet](#)

[2-1-4 Barcode/Image/Text Alignment](#)

[2-1-5 Text Size](#)

[2-1-6 Text Attribute](#)

[5-2-6 alignment](#)

[5-2-7 attribute](#)

[5-2-8 textSize](#)

[5-2-9 characterSet](#)

[5-2-10 internationalCharacterSet](#)

[5-2-11 textEncoding](#)

## [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
// Basic printing method
```

```
If(BXL_SUCCESS == [pController printText:@ "This is printText\r\n"])  
    NSLog(@"printText Success");
```

```
// Align to center, bold font
```

```
pController.alignment = BXL_ALIGNMENT_CENTER;           // Align to center  
pController.attribute = BXL_FT_BOLD;                   // Bold font
```

```
If(BXL_SUCCESS == [pController printText:@ "Center alignment and  
Bold Font."])  
    NSLog(@"printText Success");
```

```
// Align to right, underlined font
```

```
pController.alignment = BXL_ALIGNMENT_RIGHT;           // Align to right  
pController.attribute = BXL_FT_UNDERLINE;             // Underlined
```

```
If(BXL_SUCCESS == [pController printText:@ "Right alignment and  
Underline Font."])  
    NSLog(@"printText Success");
```

```
// Align to left, bold and underlined, magnify the font size by 2x in height and width
```

```
pController.alignment = BXL_ALIGNMENT_LEFT;           // Align to left  
pController.attribute = BXL_FT_BOLD|BXL_FT_UNDERLINE; // Bold and underlined  
pController.textSize = BXL_TS_1WIDTH|BXL_TS_1HEIGHT; // 2X in width and height
```

```
If(BXL_SUCCESS == [pController printText:@ "Left alignment and BOLD,  
Underline Font and Big Font."])  
    NSLog(@"printText Success");
```

## 5-3-10 printBox

This method prints box-shaped text.

### [Syntax]

(long)printText:(int)width height: (int)height;

### [Parameters]

*int*

Width of the box

1 == Width corresponding to one alphabet letter

*int*

Height of the box

1 == Height corresponding to one alphabet letter

### [Return Value]

Refer to 2-1-18 Result Code

### [Discussion]

Text alignment and attributes should be assigned before calling this method.

### [See Also]

[2-1-4 Barcode/Image/Text Alignment](#)

[2-1-5 Text Size](#)

[2-1-6 Text Attribute](#)

[5-2-6 alignment](#)

[5-2-7 attribute](#)

[5-2-8 textSize](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

If(BXL_SUCCESS == [pController printBox:43 height:3])
    NSLog(@"printBox Success.");
```

## 5-3-11 lineFeed

This method performs line feed.

### [Syntax]

- (long)linefeed:(int)lines

### [Parameters]

*lines*

Number of lines to feed

### [Return Value]

Refer to [2-1-18 Result Code](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
  
If(BXL_SUCCESS == [pController lineFeed:3])  
    NSLog(@"linefeed Success.");
```

## 5-3-12 nextPrintPos

This method feeds the paper to the starting point of the next label paper.

### [Syntax]

- (long)nextPrintPos

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Discussion]

This method is effective only in the label mode.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
  
If(BXL_SUCCESS == [pController nextPrintPos])  
    NSLog(@"nextPrintPos Success.");
```

## 5-3-13 printBarcode

This method prints one-dimensional or two-dimensional barcode.

### [Syntax]

```
(long)printBarcode:(char *)data
        symbology:(long)symbology
        width:(long)width
        height:(long)height
        alignment:(long)alignment
        textPosition:(long)textPosition
```

### [Parameters]

*data*

ANSI code data with null terminator to define barcode data for printing

*symbology*

Barcode type

Refer [2-1-8 Barcode Symbology](#)

*height*

Height of barcode in Dot unit with the range of 1~255

It has no effect in two-dimensional barcode.

*width*

Width of barcode with 2~7 steps in width

If barcode print area exceeds the size of paper, printing may stop or only part of the barcode is printed.

This setting has no effect to two-dimensional barcode.

*alignment*

Barcode alignment setting

Refer to [2-1-4 Barcode/Text/Image Alignment](#)

*textPosition*

Barcode text position setting

Refer to [2-1-7 Barcode Text Position](#)

### [Return Value]

Refer to [2-1-18 Result Code](#)

### [See Also]

[2-1-4 Barcode/Text/Image Alignment](#)

[2-1-7 Barcode Text Position](#)

[2-1-8 Barcode Symbology](#)

### [Example]

```
long IResult = BXL_SUCCESS;
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

// Print EAN13 barcode, one of 1D barcode type
IResult = [pController printBarcode:"123456789012"
            symbology: BXL_BCS_EAN13
            width: 3
            height:100;

If(BXL_SUCCESS == IResult)
    NSLog(@"printBarcode Success.");

// Print Code128 barcode, one of 1D barcode type
If(BXL_SUCCESS == [pController printBarcode:3])
    NSLog(@"linefeed Success.");

// Print QR barcode, one of 1D barcode type
If(BXL_SUCCESS == [pController printBarcode:3])
    NSLog(@"linefeed Success.");
```

## 5-3-14 printBitmap

This method prints image file.

### [Syntax]

```
- (long)printBitmap:(NSString *)path  
    width:(long)width  
    level:(long)level
```

### [Parameters]

*path*

Path of image file

*width*

Width of image file to convert and print, with the range of 0 ~ maximum width  
If the value is smaller than 0, image will be converted with the following conditions.  
Refer to [2-1-9 Image Width](#)

*level*

Image color level and diffusion option

Value	Description
0 ~ 100	Color level value
If fourth digit is 1	Enable diffusion processing
If fifth digit is 1	Print image using ESC * command

### [See Also]

[2-1-4 Barcode/Text/Image Alignment](#)

[7-1 Error Diffusion](#)

### [Return Value]

Refer to [2-1-18 Result Code](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
  
NSString *path = [[NSBundle mainBundle] pathForResource:@"Sample" ofType:@"png"];  
If (BXL_SUCCESS == [pController printBitmap:path  
                    width:BXL_WIDTH_FULL  
                    level:1050])  
  
    NSLog(@"printBitmap Success.");
```

## 5-3-15 printBitmapWithImage

This method prints image file.

### [Syntax]

```
- (long)printBitmapWithImage:(UIImage *) image  
                        width:(long)width  
                        level:(long)level
```

### [Parameters]

*path*

Path of image file

*width*

Width of image file to convert and print, with the range of 0 ~ maximum width  
If the value is smaller than 0, image will be converted with the following conditions.  
Refer to [2-1-9 Image Width](#)

*level*

Image color level and diffusion option

Value	Description
0 ~ 100	Color level value
If fourth digit is 1	Enable diffusion processing
If fifth digit is 1	Print image using ESC * command

### [See Also]

[2-1-4 Barcode/Text/Image Alignment](#)

[7-1 Error Diffusion](#)

### [Return Value]

Refer to [2-1-18 Result Code](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
  
UIImage* image = [UIImage imageNamed:@"sample.png"];  
If (BXL_SUCCESS == [pController printBitmapWithImage:image  
                    width:BXL_WIDTH_FULL  
                    level:1050])  
  
    NSLog(@"printBitmap Success.");
```

### 5-3-16 cutPaper

This method cuts the print paper after printing.

#### [Syntax]

- (long) cutPaper;

#### [Discussion]

Paper cutting position might be located in such a way that printed material is partially cut due to the difference in printer mechanism. In this case, adjust the cutting position using the line feed function.

#### [See Also]

[2-1-4 Barcode/Text/Image Alignment](#)

[7-1 Error Diffusion](#)

#### [Return Value]

Refer to [2-1-18 Result Code](#)

#### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

If (BXL_SUCCESS == [pController printText:@" printText\r\n"])
    NSLog(@"printText Success.");

If (BXL_SUCCESS == [pController lineFeed:3])
    NSLog(@"lineFeed Success.");

If (BXL_SUCCESS == [pController cutPaper])
    NSLog(@"cutPaper Success.");
```

## 5-3-17 checkPrinter

This method checks the printer status and returns the status in status attributes.

### [Syntax]

- (long)checkPrinter : (int) mask

### [Return Value]

Refer to [2-1-18 Result Code](#)

### [Availability]

SDK 0.6.0 and later

### [See Also]

[2-1-4 Barcode/Text/Image Alignment](#)

[2-1-10 Status check Mask](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
If (BXL_SUCCESS==  
    [pController checkStatus:BXL_MASK_COVEROPEN ] )  
{  
    If(pController.state & BXL_STS_COVEROPEN)  
        NSLog(@"printer Cover is open");  
}
```

Check Cover Status  
Only

```
If (BXL_SUCCESS==  
    [pController checkStatus:BXL_MASK_COVERSTATUS  
                    | BXL_MASK_PAPERSTATUS ] )  
{  
    If(pController.state & BXL_STS_COVEROPEN)  
        NSLog(@"printer Cover is open");  
  
    If(pController.state & BXL_STS_PAPEREMPTY)  
        NSLog(@"Paper is empty.");  
}
```

Check Cover/Paper  
Status

## 5-3-18 directIO

This method sends user-defined data or reads data from the printer.

### [Syntax]

- (long)directIO:(NSData \*)request  
response:(NSData \*\*)response

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Parameters]

*request*

It contains the ANSI CODE data to send to printer.

*response*

It contains the response from printer.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
If (BXL_SUCCESS == [pController msrReadReady] )  
    NSLog(@"msrReadReady Success.");
```

```
BYTE test1[] = {0x30, 0x31, 0x32, 0x0d, 0x0a};  
if(BXL_SUCCESS ==
```

```
    [pController directIO:[NSData dataWithBytes:test  
length:sizeof(test1)]  
requiredSize:0  
response:nil])
```

```
{  
    NSLog(@"directIO Success. ");  
}
```

Transmission Only

```
BYTE test2[] = {0x10, 0x04, 0x02};  
NSData* dataResponse = nil;  
if(BXL_SUCCESS ==
```

```
    [pController directIO:[NSData dataWithBytes:test  
length:sizeof(test2)]  
requiredSize:1  
response:&dataResponse]
```

```
{  
    NSLog(@"directIO Success. Response : %d", dataResponse.bytes);  
}
```

Transmission of Data  
with Response

## 5-3-19 nvImageList

This method obtains the list of address of images stored in NV area.

### [Syntax]

- (long)nvImageList:(NSArray \*\*)images

### [Parameters]

*images*

This parameter provides the address list.

Each address is provided as NSNumber\*. This object is automatically released and they do not have to be released explicitly by users.

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
If(pController != nil)
    NSLog(@"getInstance Fail.");

NSArray *images;

If(BXL_SUCCESS != [pController nvImageList:&images])
    NSLog(@"get List is Fail.");

for( NSNumber *n in images)
{
    NSLog(@"%d", [NSNumber intValue]);
}
```

## 5-3-20 downloadNVImage (Diffusion)

This method downloads the image data corresponding to the specified address in NV area.

### [Syntax]

```
- (long)downloadNVImage:(int)address  
    withImage:(UIImage *)image  
    width:(long)width  
    level:(long)level
```

### [Parameters]

*address*

It contains image address value ranging from 0 ~ 99. Existing image stored in the corresponding address is replaced by the new image.

*images*

It contains the target image object to be downloaded.

*width*

It is the width of target image printing. The image is resized to the maximum width when it is set to BXL\_WIDTH\_FULL.

Images that are narrower than the specified width are enlarged, and the images that are wider than the specified width are reduced.

*level*

It determines image color level and diffusion processing option.

Value	Description
0 ~ 100	Color Level Value
If fourth digit is 1	Enable diffusion processing
If fifth digit is 1	Print image using ESC * command

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Discussion]

Size of the images with a width larger than the maximum width of the printer is adjusted automatically.

### [See Also]

[7-1 Error Diffusion](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog (@"getInstance Fail.");

UIImage* image = [UIImage imageNamed:@"sample.png"];
If (BXL_SUCCESS == [pController downloadNVImage: 0
                    withImage:image
                    width:BXL_WIDTH_FULL
                    level:1050 ] )

    NSLog(@"downloadNVImage Success.");
```

## 5-3-21 downloadNVImage (Normal)

This method downloads the image data corresponding to the specified address in NV area.

### [Syntax]

- (long)downloadNVImage:(int)address  
withImage:(UIImage \*)image

### [Parameters]

*address*

It contains image address value ranging 0 ~ 99. Existing image stored in the corresponding address is replaced by the new image.

*images*

It contains the target image object to be downloaded.

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Discussion]

Size of the images with a width larger than the maximum width of the printer is adjusted automatically.

### [See Also]

[7-1 Error Diffusion](#)

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

UIImage* image = [UIImage imageNamed:@"sample.png"];
If (BXL_SUCCESS == [pController downloadNVImage: 0
                    withImage:image])
    NSLog(@"downloadNVImage Success.");
```

## 5-3-22 printNVImage

This method prints the image stored in the specified address of NV area.

### [Syntax]

- (long)printNVImage:(int)address

### [Parameters]

*address*

Useable image address ranging 0 ~ 99

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Discussion]

Image cannot be printed if it does not exist in the corresponding address.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

UIImage* image = [UIImage imageNamed:@"sample.png"];
If (BXL_SUCCESS == [pController printNVImage:0 ] )
    NSLog(@"printNVImage Success.");
```

## 5-3-23 removeNVImage

This method removes the image stored in the specified address in NV area.

### [Syntax]

- (long)removeNVImage:(int)address

### [Parameters]

*address*

Useable image address ranging 0 ~ 99

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Discussion]

No action is taken if image does not exist in the corresponding address.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];  
  
If(pController != nil)  
    NSLog(@"getInstance Fail.");  
  
UIImage* image = [UIImage imageNamed:@"sample.png"];  
If (BXL_SUCCESS == [pController removeNVImage:0])  
    NSLog(@"removeNVImage Success.");
```

## 5-3-24 removeAllNVImages

This method removes all images stored in the NV area.

### [Syntax]

- (long)removeAllNVImages

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

UIImage* image = [UIImage imageNamed:@"sample.png"];
If (BXL_SUCCESS == [pController removeAllNVImages])
    NSLog(@"removeAllNVImages Success.");
```

## 5-3-25 openDrawer

This method opens Cash Drawer.

### [Syntax]

- (long)openDrawer

### [Return Value]

Refer to [2-1-18 Result Code](#).

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

If(BXL_SUCCESS== [pController openDrawer])
    NSLog(@"OpenDrawer Success.");

else
    NSLog(@"OpenDrawer Fail.");
```

## 5-3-26 isSupport\_CashDrawer

This method checks the support of CashDrawer-related functions.

### [Syntax]

- (BOOL)isSupport\_CashDrawer

### [Return Value]

This method returns TRUE if CashDrawer-related functions are supported.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];
```

```
If(pController != nil)  
    NSLog(@"getInstance Fail.");
```

```
BOOL          isSupportCashDrawer=          [pController  
isSupport_CashDrawer];
```

```
If(isSupportCashDrawer)  
    NSLog(@"CashDrawerFunctions is Supported.");
```

```
else  
    NSLog(@"CashDrawerFunctions is not Supported.");
```

### 5-3-27 isSupport\_LSB

This method checks the support of LSB-related functions.

#### **[Syntax]**

- (BOOL)isSupport\_LSB

#### **[Return Value]**

This method returns TRUE if LSB-related functions are supported.

#### **[Example]**

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

BOOL isSupportLSB = [pController isSupport_LSB];

If(isSupportLSB)
    NSLog(@"LastStatusBack is Supported.");
else
    NSLog(@"LastStatusBack is not Supported.");
```

## 5-3-28 getBarcodeSupportTable

This method checks the support of Barcode.

### [Syntax]

- (NSMutableArray\*)getBarcodeSupportTable

### [Return Value]

The method returns NSMutableArray that contains BXBarcode.

### [Example]

```
BXPrinterController* pController = [BXPrinterController getInstance];

If(pController != nil)
    NSLog(@"getInstance Fail.");

NSMutableArray *tbl = [printerController getBarcodeSupportTable];

If( nil == tbl)
    NSLog(@"getBarcodeSupportTable Fail.");
for(BXBarcode *p in tbl)
{
    NSLog("barcode number: %d , p.barNumber");
    NSLog("barcode name : %@", priner.name);
    NSLog("is Support : %d", p.support);
}
```

## 6. BXPrinterControllerDelegate Protocol Reference

### 6-1 Overview

This protocol is for handling events occurring inside the BXPrinterController class.

### 6-2 Instance Methods

#### 6-2-1 didStart

This method is called when the use of class is started using the open method of BXPrinterControll.

It is called after connection with printer is established.

#### **[Syntax]**

- (void) didStart

#### **[Parameters]**

*controller*

It is a BXPrinterController object for handling events.

#### **[Discussion]**

It can be used to indicate the start of use of printer class to users.

## 6-2-2 didStop

This method is called when the use of class is stopped using the close method of BXPrinterControll.

### [Syntax]

- (void) didStop

### [Parameters]

*controller*

It is a BXPrinterController object for handling events.

### [Discussion]

It can be used to indicate the end of use of printer class to users.

## 6-2-3 didFindPrinter

This method is called for each printer searched and found in the same network.

### [Syntax]

- (void) didFindPrinter:(BXPrinterController \*)controller  
printer:(BXPrinter \*)printer

### [Parameters]

*controller*

It is a BXPrinterController object for handling events.

*printer*

Information of searched printer

### [Example]

```
// The following method is called when "connect" operation is successful.  
- (void) didFindPrinter:(BXPrinterController *)controller  
printer:(BXPrinter*) printer  
{  
    NSLog("did Find Printer, ");  
    NSLog("printer Name : %@", priner.name);  
    NSLog("printer Address : %@", priner.address);  
    NSLog("printer MacAddress : %@", priner.macAddress);  
}
```

## 6-2-4 didConnect

This method is called after connection with printer is completed.

### [Syntax]

```
- (void)didConnect:(BXPrinterController *)controller  
                printer:(BXPrinter*) printer
```

### [Parameters]

*controller*

It is a BXPrinterController object for handling events.

*printer*

It is a BXPrinter object for handling events.

### [Discussion]

It can be used to remove the status shown to users before connecting with printer. Refer to the target property of BXPrinterController if information of target connection printer is required.

### [See Also]

[5-3-6 connect](#)

### [Return Value]

YES if operation is successful.

It contains the result to the Connect request.

Refer to the description of delegate function to check the results of connection.

### [Example]

```
// The following function is called when "connect" is successful.  
- (void)didConnect:(BXPrinterController *)controller  
                printer:(BXPrinter*) printer  
{  
    NSLog("Connect Complete");  
}
```

## 6-2-5 didNotConnect

This function is called when connection to printer fails.

### [Syntax]

```
- (void)didNotConnect:(BXPrinterController *)controller  
    withError:(NSError *)error
```

### [Parameters]

*controller*

BXPrinterController object for handling events

*error*

Information about the cause of failure

### [Discussion]

This method cannot be used if there is an error during connection process.

### [Example]

```
// The following function is called when "Disconnect" is completed.  
- (void)didNotConnect::(BXPrinterController *)controller withError:(NSError*) error  
{  
    NSLog("cannot connect to the printer.");  
}
```

## 6-2-6 willLookupPrinters

This method is called before searching printers.

### [Syntax]

```
- (void)willLookupPrinters:(BXPrinterController *)controller
```

### [Parameters]

*controller*

BXPrinterController object for handling events.

### [Discussion]

It can be used to provide separate indication to users during printer search process.

### [Example]

```
// The following function is called when "Disconnect" is completed.  
- (void)willLookupPrinters::(BXPrinterController *)controller  
{  
    NSLog("the lookup did start. ");  
}
```

## 6-2-7 didLookupPrinters

This function is called when printer search process is completed.

### [Syntax]

- (void)didLookupPrinters:(BXPrinterController \*)controller

### [Parameters]

*controller*

BXPrinterController object for handling events

### [Discussion]

It can be used to provide separate indication to users during printer search process.

### [Example]

```
// The following function is called when "Disconnect" is completed.  
- (void)didNotLookup::(BXPrinterController *)controller withError:(NSError*) error  
{  
    NSLog("printer lookup fail.");  
}
```

## 6-2-8 didNotLookup

This method is called when printer search operation fails.

### [Syntax]

- (void)didNotLookup:(BXPrinterController \*)controller  
withError:(NSError \*)error

### [Parameters]

*controller*

BXPrinterController object for handling events

*error*

Information about cause of failure

### [Example]

```
// The following function is called when "Disconnect" is completed.  
- (void)didNotLookup::(BXPrinterController *)controller withError:(NSError*) error  
{  
    NSLog("printer lookup fail.");  
}
```

## 6-2-9 didBeBrokenConnection

This method is called when the printer gets disconnected.

### [Syntax]

```
- (void)didBeBrokenConnection:(BXPrinterController *)controller  
    withError:(NSError *)error
```

### [Parameters]

*controller*

BXPrinterController object for handling events

*error*

Information about cause of failure

### [Discussion]

This function is not called when the printer is disconnected by explicitly calling the close method of BXPrinterController by users. It is called only when the printer gets disconnected by external failure rather than user command.

Refer to the target property of BXPrinterController if the information of target printer is required.

### [Example]

```
// The following function is called when "Disconnect" is completed.  
- (void)didBeBrokenConnection::(BXPrinterController *)controller withError:(NSError*) error  
{  
    NSLog("connection is broken.");  
}
```

## 6-2-10 didDisconnect

This method is called when the process to disconnect the printer is completed.

### [Syntax]

```
- (void)didDisconnection:(BXPrinterController *)controller  
    printer:(BXPrinter*) printer
```

### [Parameters]

*controller*

BXPrinterController object for handling events

*printer*

BXPrinter object for handling events

### [See Also]

[5-3-7 disconnect](#)

### [Example]

```
// The following function is called when "Disconnect" is completed.  
- (void)didDisconnect:(BXPrinterController *)controller printer:(BXPrinter*) printer  
{  
    NSLog("Disconnect Complete");  
}
```

## 6-2-11 didUpdateStatus

This method is called when printer status is changed.

### [Syntax]

```
- (void)didUpdateStatus:(BXPrinterController *)controller  
    Status(NSNumber*) status
```

### [Parameters]

*controller*

BXPrinterController object for handling events

*status*

Current status of printer

### [See Also]

[2-1-12 State](#)

[5-3-8 enableLSB](#)

### [Example]

```
// After printer connection is completed  
BXPrinterController * pController = [BXPrinterController getInstance];  
[pController enableLSB:YES];  
  
// After that, the following delegate function is called when status is changed.  
- (void)didUpdateStatus:(BXPrinterController *)controller  
    status:(NSNumber*)status  
{  
    NSLog("did Update Status");  
  
    If(status & BXL_STS_NORMAL)  
        NSLog(@"Printer Status is Normal.");  
  
    If(status & BXL_STS_PAPEREMPTY)  
        NSLog(@"Paper is empty.");  
  
    If(status & BXL_STS_COVEROPEN)  
        NSLog(@"Printer Status is Normal.");  
}
```

**7. Appendix**

**7-1 Error Diffusion**

It is a technology to represent color image or black and white image with fewer bits or pixels. It has superior capability to preserve sharp images although obvious side effects like snake patterns can be seen in some specific types of images.

One of the disadvantages is long processing time, and this is because of the number of computations required to measure errors and distribute them to neighboring pixels.

It is recommended to use the error diffusion algorithm with this SDK.

[Original Image]



Application of Diffusion Algorithm	Result	
X	 <p data-bbox="900 1386 1046 1420">[Level: 50]</p>	
O	 <p data-bbox="632 1650 820 1684">[Level: 1020]</p>	 <p data-bbox="1125 1650 1313 1684">[Level: 1035]</p>
O	 <p data-bbox="619 1951 807 1984">[Level: 1050]</p>	 <p data-bbox="1125 1951 1313 1984">[Level: 1070]</p>