BIXOLON

API Reference Guide

Web Print SDK

Ver. 1.12

http://www.bixolon.com

Table of Contents

Copyright

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

Caution

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

1. Manual Guide

This manual contains the information needed to create applications in using BIXOLON's label printers, POS printers and mobile printers with the BIXOLON Web Print SDK.

Those who are using the Web Print SDK are recommended to carefully read the instructions in this manual prior to use.

2. Overview of Web Print SDK API

2-1 Features

• The purpose of this API is to make the control of BIXOLON printer more accessible on web application.

2-2 Terminology

• JSON: As an abbreviation of JavaScript Object Notation, this data format gives and receives a simple form of data

2-3 Supported OS

2-3-1 Windows

Microsoft Windows 7 (32bit/64bit)
Microsoft Windows 8 (32bit/64bit)
Microsoft Windows 10 (32bit/64bit)

2-3-2 iOS

iOS 10 and later

2-3-3 Android

Android 6.0 and later

2-4 Supported Interface

2-4-1 Windows

USB, Serial, Parallel, Bluetooth, Wi-Fi, Ethernet

2-4-2 iOS

Bluetooth, Wi-Fi, Ethernet

2-4-3 Android

Bluetooth, Wi-Fi, Ethernet, USB

Refer to the user manual for each OS for installation and settings of the Web Print SDK.

2-5 Supported Web Browser

2-5-1 Windows

jQuery and XmlHttpRequest Object-enabled web browsers

Recommended Browsers
- Chrome
- Edge
- Internet Explorer 11

2-5-2 iOS

Web Print SDK's Browser

2-5-3 Android
- Chrome Browser
- Web Print SDK's Browser

• If you use Chrome Browser, the Connection must be HTTPS after version 94.0.4606.61.

2-6 Supported Printer Models

2-6-1 POS Printer
SRP-S300 / SRP-S320
SRP-S200
SRP-Q300 / SRP-Q302
SRP-380 / SRP-382 / SRP-383
SRP-F310II / SRP-F312II / SRP-F313II
SRP-350plusIII / SRP-352plusIII
SRP-275III
SRP-S3000
SRP-330II / SRP-332II
SRP-Q200
SRP-QE300 / SRP-QE302
STP-103III
SRP-B300

2-6-2 Label Printer
SLP-TX400 / SLP-TX403
SLP-TX420 / SLP-TX423
SLP-TX220 / SLP-TX223
SLP-DX420 / SLP-DX423
SLP-DX220 / SLP-DX223
SLP-DL410 / SLP-DL413
XT5-40(RFID) / XT5-43(RFID) / XT5-46(RFID)
XL5-40CT / XL5-43CT
XD5-40d / XD5-43d
XD5-40t (RFID)/ XD5-43t(RFID)
SRP-S3000_LABEL
XQ-840 / 843(Android only)
XD3-40d / XD3-40t
SRP-E770III
XT3-40 / XT3-43

2-6-3 Mobile Receipt Printer
SPP-R200III
SPP-R310
SPP-R410
SPP-R210
SPP-R220
SPP-C200
SPP-C300

2-6-4 Mobile Label Printer
SPP-L3000
XM7-20
XM7-40(RFID)
SPP-L310
SPP-L410

2-6-5 B-gate
BGT-100P / BGT-102P
SRP-Q300H / SRP-Q302H
SRP-S300H / SRP-S320H
SRP-F310IIH / SRP-F312IIH

2-7 Supported function

2-7-1 Port Settings

The default value is 18080.

The reception port can be changed; in case of change, it is required to change the TCP port, which requests Web Print SDK to print on the web browser.

- When the listener port is overlapped with other APP used, it is necessary to change the reception port.
- When the listener port is overlapped, the Web Print SDK APP may operate abnormally.
- Without restart app after listener port is changed, the app may operate abnormally

2-7-2 Repetitive request processing function

If you use this feature, ignore duplicate print requests that occur in the Web Browser. Default setting of duplicate request processing is deactivated.

3. Common Reference

3-1 Overview

• bxlcommon.js file provides a function to request printing and check the printing result.

For communication with Web Print SDK App on the web browser, XmlHttpRequest and WebSocket object is used. The request for printing is implemented in requestPrint function, and checking for printing result is implemented in checkResult function.
MSR and output requests using Web Socket are implemented in WebSocketPrint function.

• For Windows version, a sample is automatically installed in the Web Print Server installation folder.
• JSON data uses UTF-8 encoding. If it is not UTF-8 encoding, the print may operate abnormally.

Sample composition

- Sample_PosPrinter.html: Receipt printer sample
- Sample_LabelPrinter.html: Label printer sample
- Sample_MSR.html: MSR sample
- js/bxlcommom.js: Request for printing and request for checking printing result
- js/bxlpos.js: JSON data generation API
- js/bxllabel.js: JSON data generation API

3-2 Request URL

Requirements	Method	URL
Request for printing	POST	/<PrinterName>
Request for printing result	POST	/<PrinterName>/checkStatus
Request for MSR Data	POST	/<PrinterName>/requestMSRData

3-3 bxlcommon.js

- Web Print SDK App request URL
  : http://127.0.0.1:18080/WebPrintSDK/
- It is declared as var serverURL in js.
- URL is case sensitive.
- WebSocket requests
  : ws://127.0.0.1:18080/WebPrintSDK/
- https / wss URL should be used for Secure Socket Layer (SSL) request.

3-3-1 Request for printing

Web Print SDK generates XmlHttpRequestObject or WebSocket and requests printing
The sequence of printing request is as follows:

1) Generate XmlHttpRequest or WebSocket object

```
//---------------------- XmlHttpRequest object generation code ----------------------
var httpRequest;
if (window.XMLHttpRequest)    {
      httpRequest = new XMLHttpRequest();
}

//------------------------ WebSocket object generation code ------------------------
var _websocket;
_websocket = new WebSocket(uri);
```

2) Request printing with the printing request URL of Web Print SDK.

3-3-2 Request for printing result

Web Print SDK generates XmlHttpRequestObject requests checking for printing result.
The sequence of checking for printing result is as follows:

1) Generate XmlHttpRequest object.

2) Create JSON Data to request checking for printing result.

```
function makeResultInquiryData(requestId, responseId, timeout) {
   return "{\"RequestID\":"+requestId+",” +
          ”\"ResponseID\":\""+responseId+"\",”+
          ”\"Timeout\":"+timeout+"}";
}
```

Request the printing result of Web Print SDK to the URL.
Timeout unit is second

Connection is maintained until print processing is completed for WebSocket request so printing results are not requested.

3-3-3 MSR Data request

Web Print SDK generates XmlHttpRequest or WebSocket Object requests checking for MSR Data.
The sequence of MSR data request is as follows:

1) XmlHttpRequest or WebSocket Object is generated

2) MSR Data request JSON Data is generated

```
var inquiryData = "{\"Timeout\":"+timeout+"}";
```

Timeout indicates ready-time(Sec) for the MSR reading

Only available on the printer equipped with MSR.

4. POSPrinter Reference

4-1 Overview

• bxlpos.js file provides a basic function for POS printer/mobile receipt printer device.
The structure of JSON data generated by bxlpos.js is as follows:

```
{
   “id”:1, //setId function
   “functions”:{     //printing function
            “func1”:{“function name”:[func1 parameters]},
            “func2”:{“function name”:[func2 parameters]},
             ....
             “funcN”:{“function name”:[funcN parameters]}
            }
}
```

4-2 bxlpos.js

4-2-1 getPosData

Gets the generated json data.

[Syntax]

function getPosData()

[Return Value]

json data

[Example]

```
function getPosData()
{
    var strSubmit = getPosData();
    console.log(strSubmit);
}
```

4-2-2 setPosId

When the repetitive request processing function is used, it is necessary to use this function.
When the repetitive request processing function is not used, it is not necessary to use this function.

[Syntax]
function setPosId(requestId)

[Parameters]
requestId: Print request ID value (Set by the user)

[Example]

```
function setPosId ()
{
    var issueID = 1;
    setPosId(issueID);

    issueID++;
}
```

4-2-3 checkPrinterStatus

To check the status of printer; when the printer is in error state, the function which is called afterwards is not processed.

[Syntax]
function checkPrinterStatus()

[Example]

```
function checkPrinterStatus ()
{
    checkPrinterStatus();
}
```

4-2-4 directPrintText

To send the text character string to the printer.

[Syntax]
function directPrintText(text)

[Parameters]
text: Character string to print out

[Example]

```
function directPrintText ()
{
    directPrintText (“Bixolon\n”);
}
```

4-2-5 directPrintHex

To convert the character string in a hexadecimal digit to hex value and send it to the printer.

[Syntax]
function directPrintHex(hexstring)

[Parameters]
hexstring: Character string in a hexadecimal digit to print out

[Example]

```
function directPrintHex ()
{
    directPrintHex(“3132330a”);
}
```

4-2-6 cutPaper

To cut the paper

[Syntax]

function cutPaper(type)

[Parameters]

type: Cutting type(0 : Cut in place, 1 : Cut after feeding)

[Example]

```
function cutPaper ()
{
    cutPaper (1);
}
```

4-2-7 setInternationalCharset

To set up the International CharacterSet of the printer

[Syntax]

function setInternationalCharset(ics)

[Parameters]

ics: International Character Set value

Value	Description
0	U.S.A
1	France
2	Germany
3	U.K
4	Denmark I
5	Sweden
6	Italy
7	Spain I
8	Japan
9	Norway
10	Denmark II
11	Spain II
12	Latin America
13	Korea

[Example]

```
function setInternationalCharset ()
{
    // set U.S.A
    setInternationalCharset(0);
}
```

4-2-8 setCharacterset

To set the printer code page; when the character string is printed, it is encoded as the same value as the printer code page

[Syntax]

function setCharacterset(charset)

[Parameters]

charset: codepage value

Value	Description
437	USA, Standard Europe (PC437)
1	Katakana
850	Multilingual (PC850)
860	Portuguese (PC860)
863	Canadian-French (PC863)
865	Nordic (PC865)
1252	Latin 1 (WPC1252)
866	Cyrillic #2 (PC866)
852	Latin 2 (PC852)
858	Euro (PC858)
862	Hebrew DOS code (PC862)
864	Arabic (Arabic)
1253	Greek (WPC1253)
1254	Turkish (WPC1254)
1257	Baltic (WPC1257)
27	Farsi
1251	Cyrillic (WPC1251)
737	Greek (PC737)
775	Baltic (PC775)
32	Hebrew Old code
1255	Hebrew New code (WPC1255)
34	Thai character code 11 (Thai 11)
35	Thai character code 18 (Thai 18)
855	Cyrillic (PC855)
857	Turkish (PC857)
928	Greek (PC928)
39	Thai character code 18 (Thai 16)
1256	Arabic
1258	Vietnamese (PC1258)
42	Khmer
1250	Czech
48	Vietnamese (TCVN-3)
49	Vietnamese (TCVN-3 Capital)
50	Vietnamese (VISCII)

949	Korean(KS5601)
932	Japanese(SHIFT_JIS)
950	Traditional Chinese(EUC-CN)
936	Simplified Chinese(GB2312)

[Example]

```
function setCharacterset ()
{
    // set PC437
    setCharacterset(437);
}
```

4-2-9 printText

To print out the character string

[Syntax]

function printText(text, horizontal, vertical, bold, invert, underline, fonttype, alignment)

[Parameters]

1) text: Character string to print out
2) horizontal: Horizontal magnification of character (0: Default size, ..., 7: 8 times)

※ *Only 0 or 1 is allowed for SRP-275III*

3) vertical: Vertical magnification of character (0: Default size, ..., 7: 8 times)

※ *Only 0 or 1 is allowed for SRP-275III*

4) bold: Bold(Not setup: 0 or false, setup: 1 or true)
5) invert: Inverse(Not setup: 0 or false, setup: 1 or true)
6) underline: Underline(Not setup: 0 or false, setup: 1 or true)
7) fonttype: font selection

Value	Description
0	Font A
1	Font B
2	Font C

8) alignment: Alignment

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

[Example]

```
function printText ()
{
   printText("\n\nMiraeAsset Venture Tower 685 \n\n", 0, 0, false, false, false, 0, 1);
   printText("----------------------------------------\n", 0, 0, false, false, false, 0, 0);
   printText("Item name                Q'ty      price \n", 0, 0, false, false, false, 0, 0);
   printText("Items 1                   1       100.00\n", 0, 0, false, false, false, 0, 0);
   printText("Items 2                   1       200.00\n", 0, 0, false, false, false, 0, 0);
   printText("Items 3                   1       300.00\n", 0, 0, false, false, false, 0, 0);
   printText("Items 4                   1       400.00\n", 0, 0, false, false, false, 0, 0);
```

```
    printText("Items 5                    1        500.00\n", 0, 0, false, false, false, 0, 0);
    printText("----------------------------------------\n", 0, 0, false, false, false, 0, 0);
}
```

4-2-10 print1DBarcode

Print 1D Barcode.

[Syntax]

function print1DBarcode(data, symbol, barWidth, height, hriPosition, alignment)

[Parameters]

1) data: Barcode Data
2) symbol: Barcode Type

Value	Description
0	UPC A
1	UPC E
2	JAN8/EAN8
3	JAN13/EAN13
4	CODE39
5	ITF
6	CODABAR
7	CODE93
8	CODE128

3) barWidth: Horizontal size of the bar code (2~6)
4) height: height of the bar code (1 ~255)
5) hriPosition: Selects the printing position of HRI

Value	Description
0	Not Printed
1	Above the barcode
2	Below the barcode

6) alignment: Alignment

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

[Example]

```
function print1DBarcode ()
{
    print1DBarcode("01234567890",0,4,70,2,1);
    print1DBarcode("01234567890",1,4,70,0,0);
    print1DBarcode("01234567890",8,4,70,0,2);
}
```

4-2-11 printPDF417

Print to PDF417 barcode

[Syntax]

function printPDF417(data, symbol, alignment, columnNumber, rowNumber, moduleWidth, moduleHeight, eccLevel)

[Parameters]

1) data: Barcode Data
2) symbol: PDF417 Barcode Type

Value	Description
0	PDF417
1	PDF417 Simplified

3) alignment: Alignment selection

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

4) columnNumber: The number of columns in the data area of PDF417 (0 ~ 30)
5) rowNumber: The number of rows in the data area of PDF417 (3 ~ 90)
6) moduleWidth: Module width (1 ~ 4)
7) moduleHeight: Module height (2 ~ 8)
8) eccLevel: The error correction level for PDF417 (0 ~ 8)

[Example]

```
function printPDF417 ()
{
    printPDF417("012345",0,2,20,3,1,2,4);
}
```

4-2-12 printQRCode

Print to QR CODE

[Syntax]

function printQRCode(data, model, alignment, moduleSize, eccLevel)

[Parameters]

1) data: Barcode Data
2) model: QR CODE Type

Value	Description
0	Model 1
1	Model 2

3) alignment: Alignment selection

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

4) moduleSize: The Size of module (1 ~ 7)
5) eccLevel: the error correction level (0 ~ 3)

Value	Description
0	Level L 7%
1	Level M 15%
2	Level Q 25%
3	Level H 30%

[Example]

```
function printQRCode ()
{
    printQRCode("012345",0,1,6,1);
}
```

4-2-13 printGS1Databar

Print to GS1Databar

[Syntax]

function printGS1Databar(data, symbol, alignment, moduleSize)

[Parameters]

1) data: Barcode Data
2) symbol: Barcode Type

Value	Description
0	GS1 DataBar
1	GS1 DataBar Omnidirectional

3) alignment: Alignment selection

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

4) moduleSize: The Size of module (1 ~ 8)

[Example]

```
function printGS1Databar ()
{
    printGS1Databar("012345",0,0,8);
}
```

4-2-14 printDataMatrix

Print to DataMatrix

[Syntax]

function printDataMatrix(data, alignment, moduleSize)

[Parameters]

1) data: Barcode Data
2) alignment: Alignment selection

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

3) moduleSize: Module Size (2 ~ 3)

[Example]

```
function printDataMatrix ()
{
    printDataMatrix("012345",0,2);
}
```

4-2-15 printCompositeBarcode

Print to Composite Barcode

[Syntax]

function printCompositeBarcode (data, cData, symbol, cSymbol, alignment, moduleSize)

[Parameters]

1) data: First Barcode Data
2) cData: Second Barcode Data
3) symbol: The barcode type of first parameter

Value	Description
0	EAN8
1	EAN13
2	UPC-A
3	UPC-E
4	GS1 DataBar Omnidirectional
5	GS1 DataBar Turncated
6	GS1 DataBar Stacked
7	GS1 DataBar Stacked Omnidirectional
8	GS1 DataBar Limited
9	GS1 DataBar Expanded
10	GS1-128

4) cSymbol: The barcode type of second parameter

Value	Description
0	Automatic
1	Fixed to CC-C (Only GS1-128)

5) alignment: Alignment

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

6) moduleSize The Size of module (1 ~ 8)

[Example]

```
function printCompositeBarcode ()
{
    printCompositeBarcode("012345", "012345",2,0,1,7);
}
```

4-2-16 printBitmap

Prints to image. Image data must be encoded by base64.

[Syntax]

function printBitmap(imagedata, width, alignment, dither)

[Parameters]

1) imagedata: Image data(base64 encoding)
2) width: Image width

When printing as much as the original size of the image -2

3) alignment: Alignment selection

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

4) dither: Dithering Or Not(Not setup: 0 or false, Setup: 1 or true)

[Example]

```
function printBitmap ()
{
    var imgData = canvas.toDataURL();
    printBitmap(imgData, 400, 1, 1);
}
```

4-2-17 printBitmapFile

Print the image file.

[Syntax]

function printBitmapFile(filepath, width, alignment, dither)

[Parameters]

1) filepath: Image file path

Windows : fullpath
iOS : filename,(filename in Document path)
Android : filename(File absolute path and file name)

2) width: Image width

When printing as much as the original size of the image -2

3) alignment: Print the image file.

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

4) dither: Dithering Or Not (Not Used: 0 or false, Used: 1 or true)

If the width of the image is larger than the width of the receipt, the image may not be printed.

[Example]

```
function printBitmapFile ()
{
    //imageFileWindows must use the full path of the local pc.
    var imageFileWindows = "C:\\BIXOLON\\Web Print SDK\\logo.bmp";
    printBitmapFile(imageFileWindows,-2,1,false);

    //imageFileiOS must use the filename in Document.
    var imageFileiOS = "BIXOLON.bmp";
    printBitmapFile(imageFileiOS,-2,1,false);

    //imageFileAndroid must use the File absolute path and file name
    Var imageFileAndroid=" /storage/emulated/0/bixolon/logo.bmp";
    printBitmapFile(imageFileAndroid,-2,1,false);

}
```

4-2-18 printPDFFile

Print the PDF file.

[Syntax]

function printPDFFile(filepath, pageNumber, width, alignment, dither)

[Parameters]

1) filepath: Image PDF path

Windows : not support
iOS : filename(filename in Document path)
Android : not support

2) pageNumber: Print Page

1st page of PDF is starts with Parameter 0
If the page number is non-existed, printing is not proceeded
Total page printing parameter is Total PDF page -1

3) width: Image width

4) alignment

Value	Description
0	Left-aligned
1	Centered
2	Right-aligned

5) dither: Dithering Or Not (Not Used: 0 or false, Used: 1 or true)

If the width of the image is larger than the width of the receipt, the image may not be printed.

[Example]

```
function printPDFFile ()
{
    var filePath = "Test.pdf";
    printPDFFile(filePath,-1,550,0);
}
```

4-2-19 pagemodeBegin

Enters 'Page' mode

[Syntax]
function pagemodeBegin()

4-2-20 pagemodePrintArea

Sets the printing area of page mode.

[Syntax]
function pagemodePrintArea(width, height)

[Parameters]
1) width: Printing area Width
2) height: Printing area Height

4-2-21 pagemodePrintPosition

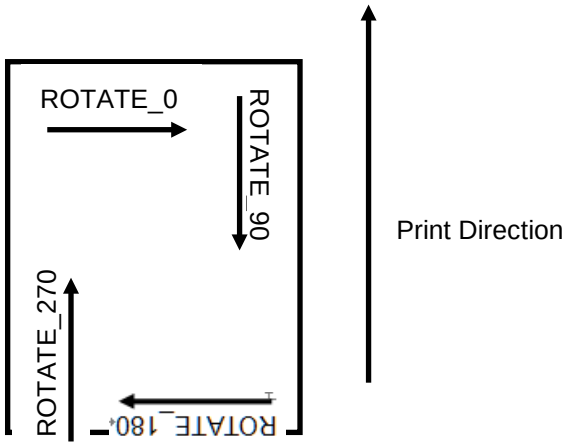
Sets the X and Y coordinates of the printing target in page mode.

[Syntax]
function pagemodePrintPosition(x,y)

[Parameters]
1) x: x-coordinate
2) y: y-coordinate

4-2-22 pagemodePrintDirection

Sets the printing direction in page mode



[Syntax]

function pagemodePrintDirection(direction)

[Parameters]

direction: Print Direction

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
4	270 degrees

4-2-23 pagemodeEnd

Leaves 'Page' mode, and then sends the data in the buffer to the printer.

[Syntax]

function pagemodeEnd()

[Example]

```
function PrintPagemode (rotation)
{

    pagemodeBegin();

    pagemodePrintArea(512, 1200);

    pagemodePrintDirection(rotation);

    if (rotation == 0) {//Normal
        pagemodePrintPosition(0, 80);
        printText("Web Print SDK!\n", 0, 1, false, false, false, 0, 0);
        pagemodePrintPosition(100, 150);
        printText("Test Print!!\n", 0, 0, false, false, false, 0, 0);
        pagemodePrintPosition(300, 350);
        printQRCode("http://www.bixolon.com", 0, 0, 5, 0);

    } else if(rotation == 1 || rotation == 3) {    //Left90 or Right90
        pagemodePrintPosition(300, 100);
        printText("Web Print SDK\n", 0, 1, false, false, false, 0, 0);
        pagemodePrintPosition(300, 170);
        printText("Test Print!!\n", 0, 0, false, false, false, 0, 0);
        pagemodePrintPosition(400, 350);
        printQRCode("http://www.bixolon.com", 0, 0, 5, 0);
    }

    else{//Rotate180
        pagemodePrintPosition(0, 0);
        printText("\n\nWeb Print SDK!\n", 0, 1, false, false, false, 0, 0);
        pagemodePrintPosition(200, 200);
        printText("Test Print!!\n", 0, 0, false, false, false, 0, 0);
        pagemodePrintPosition(300, 300);
        printText(pagemode_direct.value, 0, 0, false, false, false, 0, 0);
    }
    pagemodeEnd();
}
```

4-2-24 openDrawer

Opens Cash Drawer.

[Syntax]

function openDrawer(pinNumber)

[Parameters]

1) pinNumber: Drawer Pin Number(0 : PIN2, 1 : PIN5)

[Example]

```
function openDrawer(pinNumber)
{
    openDrawer(0);
}
```

4-3 Samples

* sample code (Sample code is based on 3-inch paper.)

```
{
 setPosId(1);
 checkPrinterStatus();
 printBitmapFile("C:\\BIXOLON Web Print SDK\\logo.bmp",300,1,0);
 printText("\n\nMiraeAsset Venture Tower 685,\nSampyeong-dong, Bundang-
 gu,\nSeongnam-si, Gyeonggi-do,\n463-400, Korea\n\n\n", 0, 0, false, false, false, 0,
 1);
 printText("----------------------------------------\n", 0, 0, false, false, false, 0, 0);
 printText("Item name                Q'ty                price \n", 0, 0, false, false, false, 0,
0);
 printText("Items 1                  1                  100.00\n", 0, 0, false, false, false, 0,
0);
 printText("Items 2                  1                  200.00\n", 0, 0, false, false, false, 0,
0);
 printText("Items 3                  1                  300.00\n", 0, 0, false, false, false, 0,
0);
 printText("Items 4                  1                  400.00\n", 0, 0, false, false, false, 0,
0);
 printText("Items 5                  1                  500.00\n", 0, 0, false, false, false, 0,
0);
 printText("----------------------------------------\n", 0, 0, false, false, false, 0, 0);
 printText("                                Sub-Total 895.0\n", 0, 1, true, false, false, 0, 0);
 printText("                                Discount      5.0\n", 0, 1, true, false, false, 0, 0);
 printText("                                ----------------\n", 0, 0, false, false, false, 0, 0);
 printText("                                Tax Total 200.0\n", 0, 1, true, false, false, 0, 0);
 printText("                                ----------------\n", 0, 0, false, false, false, 0, 0);
 printText("                                Total    1,000.0\n", 0, 1, true, false, false, 0, 0);
 printText("----------------------------------------\n", 0, 0, false, false, false, 0, 0);
 printText("Tel : 000 - 0000 - 0000\n", 0, 0, true, false, false, 0, 0);
 printText("Homepage : www.bixolon.com\n\n\n\n\n\n\n\n", 0, 0, false, false, false, 0,
0);
 printQRCode("www.bixolon.com",0,1,7,0);
 print1DBarcode("01234567890",0,4,70,2,1);
 printText("\n\n\n\n\n", 0, 0, false, false, false, 0, 0);
 cutPaper();
 var strSubmit = getPosData();
 console.log(strSubmit);
 requestPrint(p_name.value, strSubmit, viewResult);
 return true;
}
```

5. Label Printer Reference

5-1 Overview

• bxllabel.js file provides a basic function for Label printer device.

The structure of JSON data generated by bxlpos.js is as follows:

```
{
   “id”:1, //setId function
   “functions”:{     //printing function
            “func1”:{“function name”:[func1 parameters]},
            “func2”:{“function name”:[func2 parameters]},
             ....
             “funcN”:{“function name”:[funcN parameters]}
            }
}
```

5-2 bxllabel.js

5-2-1 getLabelData

Gets the generated json data.

[Syntax]

function getLabelData()

[Return Value]

json data

[Example]

```
function getLabelData()
{
    var strSubmit = getLabelData();
    console.log(strSubmit);
}
```

5-2-2 setLabelId

When the repetitive request processing function is used, it is necessary to use this function.
When the repetitive request processing function is not used, it is not necessary to use this function.

[Syntax]
function setLabelId (requestId)

[Parameters]
requestId: Id value requesting printing (Set by the user)

[Example]

```
function setLabelId()
{
    var issueID = 1;
    setLabelId(issueID);
    issueID++;
}
```

5-2-3 checkLabelStatus

To check the status of printer; when the printer is in error state, the function which is called afterwards is not processed.

[Syntax]
function checkLabelStatus()

[Example]

```
function checkLabelStatus ()
{
    checkLabelStatus();
}
```

5-2-4 clearBuffer

To initialize the printer buffer

[Syntax]

function clearBuffer()

[Example]

```
function clearBuffer()
{
    clearBuffer();
}
```

5-2-5 printBuffer

Starts printing the content of the printer buffer.

[Syntax]

function printBuffer()

[Example]

```
function printBuffer()
{
    printBuffer();
}
```

5-2-6 directDrawText

Sends custom string data to the printer. It is available if you are using the label command(SLCS Command). If there are no print commands within the string, It will not be
print.

[Syntax]
function directDrawText(text)

[Parameters]
text: Print to text

[Example]

```
function directDrawText ()
{
    directDrawText("Bixolon\n");
}
```

5-2-7 directDrawHex

Sends hexadecimal strings to the printer by converting to hexadecimal values If there are no print commands within the string, It will not be print.

[Syntax]
function directDrawHex(hexstring)

[Parameters]
hexstring: String of hexadecimal type to Print

[Example]

```
function directDrawHex ()
{
    directDrawHex("50302C30");
}
```

5-2-8 setCharacterset

Sets the printer Code page. When you print string, it is encoded to the same value as the printer code page.

[Syntax]

function setCharacterset(ics, charset)

[Parameters]

1) ics: International Character Set Value

Value	Description
0	U.S.A
1	France
2	Germany
3	U.K
4	Denmark I
5	Sweden
6	Italy
7	Spain I
8	Norway
9	Denmark II
10	Japan
11	Spain II
12	Latin America
13	Korea
14	Slovenia/Croatia
15	China

2) charset: codepage Value

Value	Description
0	U.S.A(CP437)
1	Latin1 (CP850)
2	Latin2 (CP 852)
3	Portuguese (CP 860)
4	Canadian French (CP 863)
5	Nordic (CP 865)
6	Latin I (WCP 1252)
7	European Combined (CP 865 + WCP 1252)
8	Turkish (CP 857)
9	Greek (CP 737)
10	Latin 2 (WCP 1250)
11	Greek (WCP 1253)
12	Turkish (WCP 1254)
13	Cyrillic (CP 855)
14	Hebrew (CP 862)
15	Cyrillic (CP 866)
16	Cyrillic (WCP 1251)
17	Hebrew (WCP 1255)
18	Greek (CP 928)
19	Arabic (CP 864)
20	Baltic (CP 775)
21	Baltic (WCP1257)
22	Latin 1 + Euro (CP858)

[Example]

```
function setInternationalCharset ()
{
    // set U.S.A
    setInternationalCharset(0, 0);
}
```

5-2-9 drawDeviceFont

To input the character string by using the device font of the printer buffer

[Syntax]

function drawDeviceFont (text, x, y, fontType, widthEnlarge, heightEnlarge, rotation, invert, bold, alignment)

[Parameters]

1) text: Character string to print out
2) x: Location of x coordinate
3) y: Location of y coordinate
4) fontType: Font type

Value	Font Size	Width × Height(dots)
'0'	6	9 × 15
'1'	8	12 × 20
'2'	10	16 × 25
'3'	12	19 × 30
'4'	15	24 × 38
'5'	20	32 × 50
'6'	30	48 × 76
'7'	14	22 × 34
'8'	18	28 × 44
'9'	24	37 × 58
'a'	KOREAN 1	16 × 16 (ascii 9×15)
'b'	KOREAN 2	24 × 24 (ascii 12×24)
'c'	KOREAN 3	20 × 20 (ascii 12×20)
'd'	KOREAN 4	26 × 26 (ascii 16×30)
'e'	KOREAN 5	20 × 26 (ascii 16×30)
'f'	KOREAN 6	38 × 38 (ascii 22×34)
'm'	GB2312	24 × 24 (ascii 12×24)
'n'	BIG5	24 × 24 (ascii 12×24)
'j'	Shift JIS	24 × 24 (ascii 12×24)

5) widthEnlarge: Horizontal multiplier (1~4)
6) heightEnlarge: Vertical multiplier (1~4)
7) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

8) invert: Inverse (Not setup: 0 or false, Setup: 1 or true)
9) bold: Bold (Not setup: 0 or false, Setup: 1 or true)

10) alignment: Text Alignment

Value	Description
0	Left Alignment
1	Right Alignment
2	Write text sting form right to left.

[Example]

```
function drawDeviceFont ()
{
    drawDeviceFont("1234567890",10,10,"0",2,2,3,0,0,0);
}
```

5-2-10 drawVectorFont

To input the character string by using the vector font of the printer buffer

[Syntax]

function drawVectorFont (text, x, y, fontType, fontWidth, fontHeight, rightSpacing , bold, invert , italic ,rotation, alignment, rtol)

[Parameters]

1) text: Character string to print out
2) x: Location of x coordinate
3) y: Location of y coordinate
4) fontType: Font type

Value	Description
'U'	ASCII
'K'	KS5601
'B'	BIG5
'G'	GB2312
'J'	Shift-JIS
'a'	OCR-A
'b'	OCR-B

5) fontWidth: Font width (dots)
6) fontHeight: Font height (dots)
7) rightSpacing: Right-side character spacing(+, - option can be used : 5, +3, -10..)
8) bold: Bold (Not setup: 0 or false, Setup: 1 or true)
9) invert: Inverse (Not setup: 0 or false, Setup: 1 or true)
10) italic: Italic (Not setup: 0 or false, Setup: 1 or true)
11) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

12) alignment: Text Alignment

Value	Description
0	Left Alignment
1	Right Alignment
2	Centered

13) rtol: Character string printing direction (true: right to left, false: left to right)

[Example]

```
function drawVectorFont ()
{
    drawVectorFont("1234567890",10,10,'U',15,15,0,0,0,0,0,false);
}
```

5-2-11 drawTrueTypeFont

To input the character string by using the true type font of the printer buffer

Only Windows version is supported.

[Syntax]

function drawTrueTypeFont (text, x, y, fontname, fontsize, rotation, italic, bold, underline, compression)

[Parameters]

1) text: Character string to print out
2) x: Location of x coordinate
3) y: Location of y coordinate
4) fontname: Font name
5) fontsize: Font size
6) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

7) italic: Italic (Not setup: 0 or false, Setup: 1 or true)
8) underline: Underline (Not setup: 0 or false, Setup: 1 or true)
9) bold: Bold (Not setup: 0 or false, Setup: 1 or true)
10) compression: Image compression (Not setup: 0 or false, Setup: 1 or true)

[Example]

```
function drawTrueTyperFont ()
{
    drawTrueTypeFont("1234567890",10,10,'Arial',10,0,0,0,0,0);
}
```

5-2-12 draw1DBarcode

To input one-dimension barcode on the printer buffer

[Syntax]

function draw1DBarcode (data, x, y, symbol, narrowbar, widebar, height, rotation, hri)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) symbol: Barcode type

Value	Description
0	Code39
1	Code128
2	Interleaved 2of5
3	Codabar
4	Code93
5	UPC-A
6	UPC-E
7	EAN13
8	EAN8
9	UCC/EAN128
10	Code11
11	Planet
12	Industrial 2of5
13	Standard 2of5
14	Logmars
15	UPC/EAN Extensions
16	Postnet

5) narrowbar: Narrow bar width
6) widebar: Wide bar width
7) height: Barcode height
8) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

9) hri: HRI font size and location

Value	Description
0	Not Printed
1	Below the barcode(Font Size: 1)
2	Above the barcode(Font Size: 1)
3	Below the barcode(Font Size: 2)
4	Above the barcode(Font Size: 2)
5	Below the barcode(Font Size: 3)
6	Above the barcode(Font Size: 3)
7	Below the barcode(Font Size: 4)
8	Above the barcode(Font Size: 4)

[Example]

```
function draw1DBarcode ()
{
    draw1DBarcode("1234567890",10,10,1,3,2,96,0,3);
}
```

5-2-13 drawMaxiCode

To input MaxiCode on the printer buffer

[Syntax]

function drawMaxiCode(data, x, y, mode)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) mode: Mode selection (0, 2, 3, 4)

Value	Description
0	Mode0
2	Mode2
3	Mode3
4	Mode4

[Example]

```
function drawMaxiCode ()
{
    drawMaxiCode("1234567890",10,10,0);
}
```

5-2-14 drawPDF417

To input PDF417 on the printer buffer

[Syntax]

function drawPDF417 (data, x, y, maxRowCount, maxColumnCount, eccLevel, dataCompressionMethod, hri, barcodeOriginPoint, moduleWidth, barHeight, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) maxRowCount: Maximum Row Count (3~90)
5) maxColumnCount: Maximum Column Count (1~30)
6) eccLevel: Error Correction level (0 ~ 8)
7) dataCompressionMethod: Data compression method

Value	Description
0	Text
1	Numeric
2	Binary

8) hri: Print HRI or not (Not setup: 0 or false, Setup: 1 or true)
9) barcodeOriginPoint: Barcode origin point

Value	Description
0	Center of barcode
1	Upper left corner of barcode

10) moduleWidth: Module Width (2~9)
11) barHeight: Bar Height (4~99)
12) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawPDF417 ()
{
    drawPDF417 ("1234567890",10,10,3,10,2,0,0,1, 2, 4, 0);
}
```

5-2-15 drawQRCode

To input QRCode on the printer buffer
For an effective QRCode size value for each model, please refer to the product SLCS Command manual.

[Syntax]

function drawQRCode (data, x, y, model, eccLevel, size, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) model

Value	Description
0	Model1
1	Model2

5) eccLevel: Error Correction level (L,M,Q,H)

Value	Description
'L'	7%
'M'	15%
'Q'	25%
'H'	30%

6) size: Barcode size(1~9)
7) rotation: Rotation

Value	Rotation
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawQRCode ()
{
    drawQRCode("1234567890",10,10,0,'L',4,0);
}
```

5-2-16 drawDataMatrix

To input DataMatrix on the printer buffer

[Syntax]

function drawDataMatrix (data, x, y, size, invert, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) size: Barcode size(1~4)
5) invert: Inverse(Not setup: 0 or false, setup: 1 or true)
6) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawDataMatrix()
{
    drawDataMatrix("1234567890",10,10,3,0);
}
```

5-2-17 drawAztec

To input Aztec barcode on the printer buffer

[Syntax]

function drawAztec (data, x, y, size, extendedChannel, eccSymbol, menuSymbol, numberOfSymbols, optionalID, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) size: Barcode size(1~10)
5) extendedChannel: Extended channel interpretation code (Not setup: 0 or false, Setup: 1 or true)
6) eccSymbol: Error control and symbol size/type

Value	Description
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”

7) menuSymbol: Menu Symbol (Not setup: 0 or false, Setup: 1 or true)
8) numberOfSymbols: Number of symbols for structured append (1~26)
9) optionalID: Optional ID field for structured append (1~24)
10) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawAztec()
{
    drawAztec("1234567890",10,10,5,0,10,0,10,1,0);
}
```

5-2-18 drawCode49

To input Code49 barcode on the printer buffer

[Syntax]

function drawCode49 (data, x, y, narrowbar, widebar, height, hri, starting, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) narrowbar: Narrow bar width
5) widebar: Wide bar width
6) height: Barcode height
7) hri: HRI location(0~2)

Value	Description
0	Not Printed
1	Below the barcode
2	Above the barcode

8) starting

Value	Description
0	Regular Alphanumeric Mode
1	Multiple Read Alphanumeric
2	Regular Numeric Mode
3	Group Alphanumeric Mode
4	Regular Alphanumeric Shift 1
5	Regular Alphanumeric Shift 2
7	Automatic Mode

9) rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawCode49()
{
    drawCode49 ("1234567890",10,10,2,4,96,1,0,0);
}
```

5-2-19 drawCODABLOCK

To input CODABLOCK barcode on the printer buffer

[Syntax]

function drawCODABLOCK (data, x, y, narrowbar, widebar, height, security, dataColumns, mode, rowsEncode)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) narrowbar: Narrow bar width
5) widebar: Wide bar width
6) height: Barcode height
7) security: Security (Not setup: 0 or false, Setup: 1 or true)
8) dataColumns: (2~62)
9) mode

Value	Description
'A'	CODABLOCK A mode uses the Code 39 character set
'E'	CODABLOCK E mode uses the Code 128 character set
'F'	CODABLOCK F mode uses the Code 128 character set and Automatically adds Function 1.(FNC1)

10)rowsEncode

Value	Description
1 ~ 18	When user select mode 'A'.
2 ~ 4	When user select mode 'E', 'F'.

[Example]

```
function drawCODABLOCK()
{
    drawCODABLOCK("1234567890",10,10,2,4,96,0,5,'A',3);
}
```

5-2-20 drawMicroPDF

To input MicroPDF barcode on the printer buffer

[Syntax]

function drawMicroPDF (data, x, y, moduleWidth, height, mode, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) moduleWidth: Module width (2~8)
5) height: Barcode height (1~99)
6) mode: Barcode mode (0~33)
7) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawMicroPDF()
{
    drawMicroPDF("1234567890",10,10,2,96,1,0);
}
```

5-2-21 drawIMB

To input IMB barcode on the printer buffer

[Syntax]

function drawIMB (data, x, y, roration, hri)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

5) hri: HRI Set (Not setup: 0 or false, Setup: 1 or true)

[Example]

```
function drawIMB()
{
    drawIMB("1234567890",10,10,0,0);
}
```

5-2-22 drawMSIBarcode

To input MSI barcode on the printer buffer

[Syntax]

function drawMSIBarcode (data, x, y, narrowbar, widebar, height, checkdigit, checkdigitHri, rotation, hri)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) narrowbar: Narrow bar width
5) widebar: Wide bar width
6) height: Barcode height
7) checkdigit: Check digit selection

Value	Description
0	No check digits
1	1 Mod 10
2	2 Mod 10
3	1 Mod 11 and 1 Mod 10

8) checkdigitHri: Check digit printing on HRI (Not setup: 0 or false, Setup: 1 or true)
9) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

10) hri

Value	Description
0	Not Printed
1	Below the barcode
2	Above the barcode

[Example]

```
function drawMSIBarcode()
{
    drawMSIBarcode("1234567890",10,10,2,4,96,0,0,0,0);
}
```

5-2-23 drawPlesseyBarcode

To input Plessey barcode on the printer buffer

[Syntax]

function drawPlesseyBarcode (data, x, y, narrowbar, widebar, height, checkdigitHri, rotation, hri)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) narrowbar: Narrow bar width
5) widebar: Wide bar width
6) height: Barcode height
7) checkdigitHri: Print check digit (Not setup: 0 or false, Setup: 1 or true)
8) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

9) hri

Value	Description
0	Not Printed
1	Below the barcode
2	Above the barcode

[Example]

```
function drawPlesseyBarcode()
{
    drawPlesseyBarcode("1234567890",10,10,2,4,96,0,0,0);
}
```

5-2-24 drawTLC39Barcode

To input TLC39 barcode on the printer buffer.

[Syntax]

function drawTLC39Barcode (data, x, y, narrowbar, widebar, height, pdf417Height, pdf417narrowbar, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) narrowbar: Narrow bar width
5) widebar: Wide bar width
6) height: Barcode height
7) pdf417Height: Row height of the Micro PDF417 (1~255)
8) pdf417narrowbar: Narrow bar width of the Micro PDF417 (1~10)
9) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawTLC39Barcode()
{
    drawTLC39Barcode("1234567890",10,10,2,4,96,50,1,0);
}
```

5-2-25 drawRSSBarcode

To input RSS barcode on the printer buffer.

[Syntax]

function drawRSSBarcode (data, x, y, rssType, magnification, separatorHeight, barcodeHeight, segmentWidth, rotation)

[Parameters]

1) data: Barcode data
2) x: Location of x coordinate
3) y: Location of y coordinate
4) rssType: RSS Barcode type (0~11)

Value	Description
0	RSS14
1	RSS14 Truncated
2	RSS14 Stacked
3	RSS14 Stacked Omnidirectional
4	RSS Limited
5	RSS Expanded
6	UPC-A
7	UPC-E
8	EAN-13
9	EAN-8
10	UCC/EAN-128 and CC-A/B
11	UCC/EAN-128 and CC-C

5) magnification: Magnification (1~10)
6) separatorHeight: separator height (1~2)
7) barcodeHeight: Barcode height (This parameter only affects the UCC/EAN barcode type and CC-A/B/C barcode type)
8) segmentWidth: segmentWidth (0~22 Even numbers only, this parameter only affects the RSS Expanded barcode type.)
9) rotation: Rotation

Value	Description
0	No rotation
1	90 degrees
2	180 degrees
3	270 degrees

[Example]

```
function drawRSSBarcode()
{
    drawRSSBarcode("1234567890",10,10,0,1,1,0,0,0);
}
```

5-2-26 drawBitmap

To input image data on the printer buffer; the image data should be encoded as base64. Any image on the web application server can be printed by base64 encoding.

[Syntax]

function drawBitmap(data,x,y, width, dither)

[Parameters]

1) data: Image data encoded as base64.
2) x: Location of x coordinate
3) y: Location of y coordinate
4) width: Image width
5) dither: Dither or not (Not setup: 0 or false, Setup: 1 or true)

[Example]

```
function drawBitmap ()
{
    var imgData = canvas.toDataURL();
    draw Bitmap(imgData, 10, 10, 400, 0);
}
```

5-2-27 drawBitmapFile

To input an image file on the printer buffer; only Windows version is supported.

[Syntax]

function drawBitmapFile(filepath,x, y, width, dither)

[Parameters]

1) filepath: Path of image file to print out

Windows : fullpath
iOS : filename,(filename in Document path)
Android : filename(File absolute path and file name)

2) x: Location of x coordinate
3) y: Location of y coordinate
4) width: Image width
5) dither: Dither or not (Not setup: 0 or false, Setup: 1 or true)

[Example]

```
function drawBitmapFile ()
{
    //imageFileWindows must use the full path of the local pc.
    var imageFileWindows = "C:\\BIXOLON\\Web Print SDK\\logo.bmp";
    drawBitmapFile(imageFileWindows,10,10,400,false);

    //imageFileiOS must use the filename in Document.
    var imageFileiOS = "BIXOLON.bmp";
    drawBitmapFile(imageFileiOS,10,10,400,false);

    //imageFileAndroid must use the File absolute path and file name
    Var imageFileAndroid=" /storage/emulated/0/bixolon/logo.bmp";
    drawBitmapFile(imageFileAndroid,10,10,400,false);
}
```

5-2-28 drawCircle

To input circle data on the printer buffer

[Syntax]

function drawCircle (startHorizontal, startVertical, circleSize, muliplier)

[Parameters]

1) startHorizontal: Location of x coordinate
2) startVertical: Location of y coordinate
3) circleSize: Circle Size Selection (1~6)

Value	Description (Circle diameter)
1	5 mm
2	7 mm
3	9 mm
4	11 mm
5	13 mm
6	21 mm

4) muliplier: Circle Multiplier (1~4)

[Example]

```
function drawCircle()
{
    drawCircle(10,10,2,1);
}
```

5-2-29 drawBlock

To input line, diagonal line and square data

[Syntax]

function drawBlock(startHorizontal, startVertical, endHorizontal, endVertical, option, thickness)

[Parameters]

1) startHorizontal: Location of x coordinate starting in the horizontal direction
2) startVertical: Location of y coordinate starting in the horizontal direction
3) endHorizontal: Location of x coordinate ending in the horizontal direction
4) endVertical: Location of y coordinate ending in the horizontal direction
5) option:

Value	Description
'O'	Line Overwriting (Overwrite an overlapped part)
'E'	Line Exclusive OR (Not print out an overlapped part)
'D'	Line Delete (Delete an overlapped part)
'S'	Slope(Diagonal line)
'B'	Box(Square)

6) thickness: Thickness of diagonal line or square
(This parameter is required to have "diagonal line ("S")" or "square ("B")" as "optionvalue.)

[Example]

```
function drawBlock()
{
    drawBlock(10,10,50,50,'B',5);
}
```

5-2-30 setPrintingType

To select the printing type as Thermal Direct or Thermal Transfer mode.

If you make an invalid setup, the printer may not function properly.

[Syntax]

function setPrintingType (type)

[Parameters]

type: Printing type selection

Value	Description
'd'	Direct Thermal
't'	Thermal Transfer

[Example]

```
function setPrintingType()
{
    setPrintingType('d');
}
```

5-2-31 setMargin

To designate the margin of image buffer

[Syntax]

function setMargin(horizontal, vertical)

[Parameters]

1) horizontal: Horizontal margin
2) vertical: Vertical margin

[Example]

```
function setMargin()
{
    setMargin(10,10);
}
```

5-2-32 setLength

To set up the length of paper, length of gap/black mark and paper type of the printer
In case of continuous paper, it is necessary to use this instruction. In case of black mark or gap paper, it is not necessary to use the instruction.

[Syntax]

function setLength(labelLength, gapLength, mediaType, offset)

[Parameters]

1) labelLength: Label length
2) gapLength: Gap length or thickness of black line
3) mediaType: Media Type

Value	Description
'G'	Gap
'C'	Continuous
'B'	Black Mark

4) offset: offset Length between Black Mark(or Gap) and perforation line

[Example]

```
function setLength()
{
    setLength(1200,0,'C',0);
}
```

5-2-33 setWidth

To set up the width of paper of the printer

[Syntax]

function setWidth(width)

[Parameters]

width: The reference point to calculate the width and width of label is the center.

[Example]

```
function setWidth()
{
    setWidth(832);
}
```

5-2-34 setSpeed

To set up the speed of printing; it is ignored when an ineffective value for each model is used. For an effective value for each model, please refer to the product SLCS Command manual.

[Syntax]

function setSpeed(speed)

[Parameters]

speed: Speed set value (0~12)

Value	Description
0	2.5 ips
1	3.0 ips
2	4.0 ips
3	5.0 ips
4	6.0 ips
5	7.0 ips
6	8.0 ips
7	9.0 ips
8	10.0 ips
9	11.0 ips
10	12.0 ips
11	13.0 ips
12	14.0 ips

[Example]

```
function setSpeed()
{
    setSpeed(4);
}
```

5-2-35 setDensity

To set up the concentration of printing; it is ignored when an ineffective value for each model is used. For an effective value for each model, please refer to the product manual.

[Syntax]

function setDensity(density)

[Parameters]

density: Density Level (0~20)

[Example]

```
function setDensity()
{
    setDensity(15);
}
```

5-2-36 setOrientation

To set up the direction of printing

[Syntax]

function setOrientation(orientation)

[Parameters]

orientation: Printing direction

Value	Description
'T'	To print out from top to bottom
'B'	To print out from bottom to top

[Example]

```
function setOrientation()
{
    setOrientation('T');
}
```

5-2-37 setOffset

To set up the length of offset between gap or black mark and cutoff line; the position of starting printing is adjusted.

[Syntax]
function setOffset(offset)

[Parameters]
offset: offset value (-100 ~ 100)

[Example]

```
function setOffset()
{
    setOffset(50);
}
```

5-2-38 setTearoffPosition

To adjust the position of tearing off the label after printing

[Syntax]
function setTearoffPosition(position)

[Parameters]
position: Position (-100~100)

[Example]

```
function setTearoffPosition ()
{
    setTearoffPosition (50);
}
```

5-2-39 setAutoCutter

To set up whether the auto cutter is used or not

[Syntax]

function setAutoCutter(enable, period)

[Parameters]

1) enable: Whether the cutter is used or not (not used: 0 or false, used: 1 or true)
2) period: To determine an interval of pages for cutting

Example]

```
function setAutoCutter ()
{
    setAutoCutter (1, 3);
}
```

5-2-40 printPDF

Print the PDF file.

[Syntax]

function printPDF(filepath, pageNumber, width, dither)

[Parameters]

1) filepath: Image PDF path

Windows : not support
iOS : filename,(filename in Document path)
Android : not support

2) pageNumber: Print Page

1st page of PDF is starts with Parameter 0
If the page number is non-existed, printing is not proceeded
Total page printing parameter is Total PDF page -1

3) width: Image width
4) dither: Dithering Or Not (Not Used: 0 or false, Used: 1 or true)

This function support PDF printing without calling printBuffer()

If other API is called to draw the printing contents before printPDF(), call printBuffer() or clearBuffer() to make buffer memory empty.

[Example]

```
function printPDFFile ()
{
    var filePath = "Test.pdf";
    printPDF(filePath,-1,550,0);
}
```

5-2-41 setupRFID

For setting the RFID transponder type, number of coding retries, number of labels upon retry, and sending/receiving power

[Syntax]

function setupRFID (rfidType, numberOfRetries, numberOfLabel, radioPower)

[Parameters]

1) rfidType : RFID Transponder Type (0~5)
2) numberOfRetries : Number of Coding Retries Upon Coding Failure (1~10)
3) numberOfLabel : Number of Labels Upon Retry Following RFID Label Writing Failure (1~5)
4) radioPower : Sending/Receiving Power Adjustment (0~30)

[Example]

```
function setupRFID ()
{
   setupRFID (5, 2, 2, 21);
}
```

5-2-42 calibrateRFID

For calculating and saving the optimal coding position (read/write position of the transponder) of the RFID label on the printer and printing.

[Syntax]

function calibrateRFID()

[Example]

```
function calibrateRFID ()
{
   calibrateRFID ();
}
```

5-2-43 setRFIDPosition

For setting the RFID label coding position.

[Syntax]

function setRFIDPosition(transPosition)

[Parameters]

transPosition : RFID Label Coding Position (Y-Axis Value)

[Example]

```
function setRFIDPosition ()
{
    setRFIDPosition (320);
}
```

5-2-44 setEPCDataStructure

Defines the EPC Data structure for writing EPC Data

[Syntax]

function setEPCDataStructure(totalSize, fieldSize)

[Parameters]

1) totalSize: Total amount of bits in field
2) fieldSize : amount of bits in each field

[Example]

```
function setEPCDataStructure ()
{
    setEPCDataStructure (64, “8,8,8,8,8,8,8,8”);
}
```

5-2-45 writeRFID

For writing RFID labels.

[Syntax]

function writeRFID(dataType, startingBlockNumber, dataLength, data)

[Parameters]

1) dataType: Data Type (A : Ascii, H : Hexadecimal, E : EPC, U : User field select)
2) startingBlockNumber : Starting Block Number (4~10)
3) dataLength : Number of Bytes for Reading or Writing (2~12)
4) data : Write data

[Example]

```
function writeRFID ()
{
    writeRFID ('E',4, 2, "1,2,3,4,5,6,7,8");
}
```

5-2-46 setRFIDPassword

Setting RFID Access Password and Kill Password.

[Syntax]

function setRFIDPassword(oldAccessPwd, oldKillPwd, newAccessPwd, newKillPwd)

[Parameters]

1) oldAccessPwd : Old Access Password
2) oldKillPwd : Old Kill Password
3) newAccessPwd : New Access Password
4) newKillPwd : New Kill Password

[Example]

```
function setRFIDPassword ()
{
    setRFIDPassword (“0000”, “0000”, “1111”, “1111”);
}
```

5-2-47 lockRFID

For locking kill, access, and EPC data via the access password

[Syntax]

function lockRFID();

[Example]

```
function lockRFID ()
{
    lockRFID ();
}
```

5-3 Samples

* sample code (Sample code is based on 4 x 6 inch paper.)

```
{
	setLabelId(issueID);
	checkLabelStatus();
	clearBuffer();
	drawDeviceFont("1234567890",32,1115,"0",2,2,3,0,0,0);
	drawTrueTypeFont("S63",60,880,"Arial",80,3,false,true,false,true);
	drawDeviceFont("-",110,930,"1",2,2,3,0,1,0);
	drawDeviceFont("0",80,900,"4",2,2,3,0,1,0);
	draw1DBarcode("1234567890",213,1165,1,3,2,96,3,3);
	drawDeviceFont("BIXOLON",359,1200,"a",2,2,3,0,1,0);
	drawDeviceFont("010-0000-0000",400,1200,"0",2,2,3,0,1,0);
	drawDeviceFont("031-000-0000",430,1200,"0",2,2,3,0,1,0);
	drawDeviceFont("Gyeonggi-do Seongnam-si Bundang-
gu",459,1200,"b",1,1,3,0,1,0);
	drawDeviceFont("(Sampyeong-dong,
			Mirae Asset Venture Tower 7F)",492,1200,"b",1,1,3,0,1,0);
	drawDeviceFont("1 / 1",630,1150,"b",2,2,3,0,1,0);
	drawBitmapFile("C:\\BIXOLON Web Print SDK\\BIXOLON.bmp",0,80,100,0);
	drawBitmapFile("C:\\BIXOLON Web Print SDK\\BIXOLON.bmp",100,60,200,0);
	drawBitmapFile("C:\\BIXOLON Web Print SDK\\BIXOLON.bmp",200,50,300,0);
	drawBitmapFile("C:\\BIXOLON Web Print SDK\\BIXOLON.bmp",300,50,350,0);
	drawBlock(10,10,800,1210,"B",5);
	printBuffer();
	var strSubmit = getLabelData();
	console.log(strSubmit);
	issueID++;
	requestPrint(p_name.value, strSubmit, viewResult);
}
```

Revision history

Ver.	Date	Description
1.00	2019-05-02	New
1.01	2019-05-17	- Information of supported printers added. (XL5-40, XL5-43,XD5-40d, XD5-43d) - Modifying support range that Size parameter of drawQRcode API 4 -> 9
1.02	2019-09-02	- App name changed. (mPrint Browser -> mPrint Server)
1.03	2019-11-29	- Information of supported printers added : XD5-40t, XD5-43t : XM7-40 - Information of supported printers Changed : XL5-40 -> XL5-40CT : XL5-43 -> XL5-43CT
1.04	2019-12-20	- Add API Example
1.05	2020-01-09	- Add a comment about exception handling(SRP-275III)
1.06	2020-03-11	- Information of supported printers added : XM7-20
1.07	2020-09-14	- Modify support OS(Windows) - Indication of RFID support - Supported RFID API Added : setupRFID, calibrateRFID, writeRFID, setRFIDPosition, setEPCDataStructure, setRFIDPassword, lockRFID
1.08	2021-01-14	- Information of supported printers added : SRP-B300, XQ-840
1.09	2021-05-06	- Information of supported printers added : SRP-S320, SRP-S200, SRP-S3000, SRP-330II/332II, SRP-Q200, SRP-QE300/QE302, STP-103III : SRP-S3000_LABEL, XQ-843, XD3-40d/43d, XD3-30t/43t, SRP-E770III : SPP-R210, SPP-R220 : SPP-L310, SPP-L410 : SRP-S320H - App name changed (mPrintServer -> WebPrintSDK) - Supported Web Socket, SSL (aOS)

1.10	2021-08-26	- Information of supported printers added : XT3-40, XT3-43, XM7-40R - Supported Web Socket, SSL (Windows, iOS)
1.11	2021-10-01	- Add a comment about HTTPS at Supported Browser
1.12	2021-11-23	- Information of supported printers added : SPP-C200, SPP-C300