

**METAPACE**

**API Reference Guide**  
**METAPACE SDK**  
**for UPOS Compliant**  
**Android**

---

**Ver. 2.00**

# Table of Contents

<b>Copyright .....</b>	<b>4</b>
<b>1. About This Manual .....</b>	<b>5</b>
<b>2. Support OS and Interface .....</b>	<b>6</b>
2-1 Operating System .....	6
2-2 Supported Devices and Interfaces .....	6
<b>3. Development Environment .....</b>	<b>7</b>
3-1 System Requirements.....	7
3-2 Connecting Android Device .....	8
3-2-1 Bluetooth .....	8
3-2-2 Network .....	9
3-2-3 USB.....	10
3-2-4 Setting Android Device Developer Options.....	11
<b>4. Package Contents .....</b>	<b>12</b>
4-1 Manual.....	12
4-2 Library .....	12
4-3 Sample source cord .....	12
<b>5. Constant Value (Defines).....</b>	<b>13</b>
5-1 JposException .....	13
5-2 Event.....	14
5-2-1 StatusUpdate Event.....	14
5-2-2 Error Event.....	14
5-2-3 OutputComplete Event .....	14
5-2-4 Data Event .....	14
5-2-5 DirectIO Event.....	14
5-3 EscapeSequence.....	17
5-4 Transaction Print .....	19
5-5 Alignment.....	19
5-6 Barcode type.....	20
5-7 Barcode Text Location .....	20
5-8 Device Model Name .....	21
5-9 Print Direction in Page Mode.....	22
5-12 Character Set .....	23
<b>6. Functions by Class .....</b>	<b>24</b>
6-1 ConfigLoader Class .....	24
6-1-1 openFile().....	24
6-1-2 newFile() .....	25
6-1-3 getEntries().....	26
6-1-4 addEntry().....	27
6-1-5 removeEntry() .....	29
6-1-6 saveFile().....	30
6-2 POSPrinter Class.....	31

6-2-1 open()	31
6-2-2 claim()	32
6-2-3 setDeviceEnabled()	33
6-2-4 release()	34
6-2-5 close()	35
6-2-6 checkHealth()	36
6-2-7 setAsyncMode()	37
6-2-8 setCharacterSet()	38
6-2-9 setCharacterEncoding()	39
6-2-10 cutPaper()	40
6-2-11 printBarCode()	41
6-2-12 printBitmap()	42
6-2-13 printBitmap()	44
6-2-14 printNormal()	46
6-2-15 setPageModePrintArea()	47
6-2-16 setPageModePrintDirection()	48
6-2-17 pageModePrint()	49
6-2-18 setPageModeHorizontalPosition()	50
6-2-19 setPageModeVerticalPosition()	52
6-2-20 transactionPrint()	54
6-2-21 displayString()	55
6-2-22 cleanScreen()	56
6-2-23 storeImageFile()	57
6-2-24 displayImage()	58
6-2-25 clearImage()	59
6-2-26 directIO()	60
6-2-27 markFeed()	62
6-2-28 setBitmap()	63
6-2-29 printSvg()	65
6-2-30 updateFirmware()	67
6-2-31 printPDFFile()	68
6-2-32 printPDFFile()	70
6-3 CashDrawer Class	72
6-3-1 open()	72
6-3-2 claim()	73
6-3-3 setDeviceEnabled()	74
6-3-4 release()	75
6-3-5 close()	76
6-3-6 openDrawer()	77
6-3-7 getDrawerOpened()	78
<b>7. Samples for Test</b>	<b>79</b>
7-1 Text print	79
7-2 Image print	81
7-3 Page mode print	83

## Copyright

© METAPACE Co., Ltd. All rights reserved.

This user manual and all property of the product are protected under copyright law. It is strictly prohibited to copy, store, and transmit the whole or any part of the manual and any property of the product without the prior written approval of METAPACE Co., Ltd.

The information contained herein is designed only for use with this METAPACE product. METAPACE is not responsible for any direct or indirect damages, arising from or related to use of this information.

- The METAPACE logo is the registered trademark of METAPACE Co., Ltd.
- All other brand or product names are trademarks of their respective companies or organizations.

METAPACE 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. About This Manual**

This SDK manual describes the library required for developing applications for Android. It additionally describes how to use SDK, specifications, and restrictions.

## 2. Support OS and Interface

### 2-1 Operating System

- This software supports the following operating systems.
- Android 5.0 (Lollipop) or later is required.

### 2-2 Supported Devices and Interfaces

Models	Interface	DPI	Max Printable Width
T-3II	Ethernet / USB	180 dpi	512 dots
T-3III	Bluetooth/Ethernet/WLAN/ USB	180 dpi	512 dots

## **3. Development Environment**

### **3-1 System Requirements**

- Java Development Kit (JDK) 7
- Android Studio
- Android SDK Tools
- Reference: <http://developer.android.com/sdk/index.html>

### **3-2 Connecting Android Device**

- The following screen was captured from an Nexus 5 smart phone.  
The screen and field names might be different for different Android operating systems or devices.

#### **3-2-1 Bluetooth**

1. Select [Settings].
2. Bluetooth should be enabled and the printer power should be on.
3. Select [Bluetooth] for settings.
4. Select [Scan]. Search the printer to connect and perform pairing operation.
5. Enter PIN code. Default PIN code of METAPACE is "0000".



**3-2-2 Network**

1. Connect the printer to the network AP (Access Point) and assign an IP address or obtain one using DHCP. As METAPACE's printer is initially set to Ad-hoc, it needs to be set up first with our Net Configuration Tool. The Net Configuration Tool can be downloaded from the METAPACE website.  
(Refer to the Net Configuration Tool manual for details on settings)
2. Select [Settings].
3. Wi-Fi should be turned on.
4. Connect the device to the same network that the METAPACE printer is connected to.
5. Additional setting is not required to connect the Android device to the TCP/IP port of printer.

**3-2-3 USB**

1. Android version should 3.1 or higher to connect USB peripheral devices.
2. No special driver or printer software is required on Android device.
3. Required USB cable depends on specific smartphone or tablet.  
Be sure that the Android device to be used supports USB connection and use an appropriate cable.
4. The following message may pop up for some Android devices the first time the printer is connected to the device.
5. To connect a USB peripheral, the following code should be entered to AndroidManifest.xml and res/xml/device\_filter.xml in BXLTest provided as a sample.

[AndroidManifest.xml]

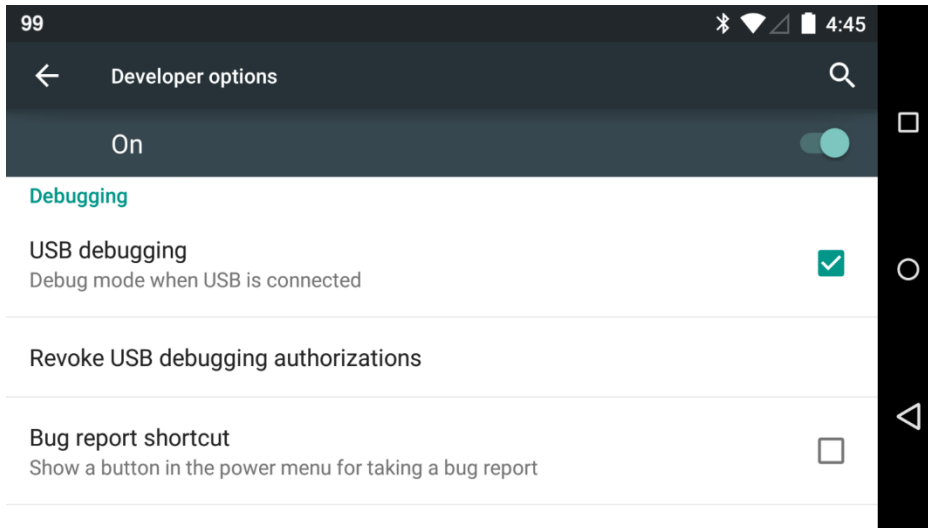
```
...  
<uses-feature android:name="android.hardware.usb.host" />  
...  
  
<intent-filter>  
    <action  
android:name="android.hardware.usb.action.USB_DEVICE_ATTACHED" />  
</intent-filter>  
  
<meta-data  
    android:name="android.hardware.usb.action.USB_DEVICE_ATTACHED"  
    android:resource="@xml/device_filter" />
```

[device\_filter.xml]

```
<?xml version="1.0" encoding="utf-8"?>  
<resources>  
  
    <usb-device  
        class="7"  
        protocol="2"  
        subclass="1" />  
  
</resources>
```

**3-2-4 Setting Android Device Developer Options**

1. Select [Settings].
2. Select [Developer options].
3. Enable [USB debugging].



## 4. Package Contents

### 4-1 Manual

Manual location/ Name	Description
docs/Manual_METAPACE SDK for Android_UPOS compliant API Reference Guide_english_Rev_x_xx	Manual in English

### 4-2 Library

Library location/ Name	Description
libs/metapace_printer_Vxxx.jar	Implementation of JavaPOS service component layers / printer setting library
libs/libcommon_Vxxx.jar	Printer control core library
libs/pdf/lib_pdf.aar	Library for PDF print
libs/jniLibs/ABI type/libcommon.so	Printer control native library

### 4-3 Sample source cord

Sample location/ Name	Description
samples/MetapaceSample	Printer/ CashDrawer control sample application

- ★ A sample reference class for better understanding of SDK usage is provided.
- Printer.java(src\main\java\com\metapace\sample\PrinterControl)

## 5. Constant Value (Defines)

### 5-1 JposException

- When an error occurs during execution of a specific function in a method, it throws a JposException exception. If an exception occurs, the contents of the error can be checked through the "printStackTrace" function.

[Example]

```
import jpos.JposConst;

try
{
    .....
}
catch(JposException e)
{
    // Error
    e.printStackTrace();
}
```

**5-2 Event**

- Each Event is defined in the JposConst and POSPrinterConst class.

**5-2-1 StatusUpdate Event**

A StatusUpdate event occurs whenever the printer status changes.

Code	Value	Description
JPOS_SUE_POWER_ONLINE	2001	Printer Power on
JPOS_SUE_POWER_OFF_OFFLINE	2004	Printer Power off
PTR_SUE_COVER_OPEN	11	Cover Open
PTR_SUE_COVER_OK	12	Cover OK
PTR_SUE_REC_EMPTY	24	Receipt Paper Empty
PTR_SUE_REC_NEAREMPTY	25	Receipt Paper Near Empty
PTR_SUE_REC_PAPEROK	26	Receipt Paper OK
PTR_SUE_IDLE	1001	Printer Idle
PTR_SUE_OFF_LINE	53	Printer off-line
PTR_SUE_ON_LINE	54	Printer on-line
PTR_SUE_BAT_OK	55	Printer battery normal
PTR_SUE_BAT_LOW	56	Printer battery low

**5-2-2 Error Event**

Code	Value	Description
JPOS_EPTR_COVER_OPEN	201	Cover Open
JPOS_EPTR_REC_EMPTY	203	Paper Empty
JPOS_EPTR_OFF_LINE	217	Printer off-line

**5-2-3 OutputComplete Event**

Generates a print completion event. However, it must be used in Async mode.

**5-2-4 Data Event**

Receives MSR Track information data.

**5-2-5 DirectIO Event**

Receives direct I / O response data.

## [Event Example]

```

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.addErrorListener(this);
    posPrinter.addStatusUpdateListener(this);
    posPrinter.addOutputCompleteListener(this);
    posPrinter.addDirectIOListener(this);
    posPrinter.setAsyncMode(true);
}
catch(JposException e)
{
    // Error
    e.printStackTrace();
}

@Override
public void outputCompleteOccurred(final OutputCompleteEvent e)
{
    runOnUiThread(new Runnable()
    {
        @Override
        public void run()
        {
            Toast.makeText(MainActivity.this, "complete print", Toast.LENGTH_SHORT).show();
        }
    });
}

@Override
public void directIOOccurred(DirectIOEvent directIOEvent) {
    runOnUiThread(new Runnable()
    {
        @Override
        public void run()
        {
            Toast.makeText(MainActivity.this, new String((byte[]) directIOEvent.getObject(),
            Toast.LENGTH_SHORT).show();
        }
    });
}

@Override
public void errorOccurred(final ErrorEvent arg0)
{
    // TODO Auto-generated method stub
    runOnUiThread(new Runnable()
    {
        @Override
        public void run()
        {
            switch (arg0.getErrorCodeExtended())
            {

```

```

        case POSPrinterConst.JPOS_EPTR_COVER_OPEN:
            return "Cover open";
        case POSPrinterConst.JPOS_EPTR_REC_EMPTY:
            return "Paper empty";
        case JposConst.JPOS_SUE_POWER_OFF_OFFLINE:
            return "Power off";
        default:
            return "Unknown";
    }
}
});
}

@Override
public void statusUpdateOccurred(final StatusUpdateEvent arg0)
{
    // TODO Auto-generated method stub
    runOnUiThread(new Runnable()
    {
        @Override
        public void run()
        {
            switch (arg0.getStatus())
            {
                case JposConst.JPOS_SUE_POWER_ONLINE:
                    return "Power on";
                case JposConst.JPOS_SUE_POWER_OFF_OFFLINE:
                    return "Power off";
                case POSPrinterConst.PTR_SUE_COVER_OPEN:
                    return "Cover Open";
                case POSPrinterConst.PTR_SUE_COVER_OK:
                    return "Cover OK";
                case POSPrinterConst.PTR_SUE_REC_EMPTY:
                    return "Receipt Paper Empty";
                case POSPrinterConst.PTR_SUE_REC_NEAREMPTY:
                    return "Receipt Paper Near Empty";
                case POSPrinterConst.PTR_SUE_REC_PAPEROK:
                    return "Receipt Paper OK";
                case POSPrinterConst.PTR_SUE_IDLE:
                    return "Printer Idle";
                default:
                    return "Unknown";
            }
        }
    });
}

```



### 5-3 EscapeSequence

- Values for specifying options such as font and thickness in the text to be printed.  
It is in string form and is added before print data.

Define	Description
<b>String ESCAPE_CHARACTERS = new String(new byte[] {0x1b, 0x7c})</b>	<b>Escape Characters</b>
ESCAPE_CHARACTERS + "N"	Normal
ESCAPE_CHARACTERS + "aM"	Font A (12x24)
ESCAPE_CHARACTERS + "bM"	Font B (9x17)
ESCAPE_CHARACTERS + "cM"	Font C (9x24)
ESCAPE_CHARACTERS + "lA"	Left justify
ESCAPE_CHARACTERS + "cA"	Center
ESCAPE_CHARACTERS + "rA"	Right justify
ESCAPE_CHARACTERS + "bC"	Bold
ESCAPE_CHARACTERS + "!bC"	Disabled bold
ESCAPE_CHARACTERS + "uC"	Underline
ESCAPE_CHARACTERS + "!uC"	Disabled underline
ESCAPE_CHARACTERS + "rvC"	Reverse video
ESCAPE_CHARACTERS + "!rvC"	Disabled reverse video
ESCAPE_CHARACTERS + "1C"	Single high and wide
ESCAPE_CHARACTERS + "2C"	Double wide
ESCAPE_CHARACTERS + "3C"	Double high
ESCAPE_CHARACTERS + "4C"	Double high and wide
ESCAPE_CHARACTERS + "1hC"	Scale 1 time horizontally
ESCAPE_CHARACTERS + "2hC"	Scale 2 times horizontally
ESCAPE_CHARACTERS + "3hC"	Scale 3 times horizontally
ESCAPE_CHARACTERS + "4hC"	Scale 4 times horizontally
ESCAPE_CHARACTERS + "5hC"	Scale 5 times horizontally
ESCAPE_CHARACTERS + "6hC"	Scale 6 times horizontally
ESCAPE_CHARACTERS + "7hC"	Scale 7 times horizontally
ESCAPE_CHARACTERS + "8hC"	Scale 8 times horizontally
ESCAPE_CHARACTERS + "1vC"	Scale 1 time vertically
ESCAPE_CHARACTERS + "2vC"	Scale 2 times vertically
ESCAPE_CHARACTERS + "3vC"	Scale 3 times vertically
ESCAPE_CHARACTERS + "4vC"	Scale 4 times vertically
ESCAPE_CHARACTERS + "5vC"	Scale 5 times vertically
ESCAPE_CHARACTERS + "6vC"	Scale 6 times vertically
ESCAPE_CHARACTERS + "7vC"	Scale 7 times vertically
ESCAPE_CHARACTERS + "8vC"	Scale 8 times vertically
ESCAPE_CHARACTERS + "[code]B"	Print NV Image [code] : 0 ~ 255(image code)

ESCAPE_CHARACTERS + "[percentage]fP"	Feed Cut [percentage] : 100 Full cut [percentage] : 0~90 Partial cut
--------------------------------------	--

[Example]

```

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open(logicalDeviceName);
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true)

    String ESCAPE_CHARACTERS = new String(new byte[] {0x1b, 0x7c})

    // in bold
    String data = ESCAPE_SEQUENCE + "bC" + "METAPACE\n"
    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, data);

    // not in bold
    String data = ESCAPE_SEQUENCE + "!bC" + "METAPACE\n"
    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, data);

    // NV Image
    String data = ESCAPE_SEQUENCE + "0B";
    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, data);

    // Feed cut
    String data = ESCAPE_SEQUENCE + "90fP";
    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, data);
}
catch(JposException e)
{
    // Error
    e.printStackTrace();
}

```

**5-4 Transaction Print**

- Definitions of values that can be used when setting Transaction mode.

Code	Value	Description
PTR_TP_TRANSACTION	11	Initializes the buffer to empty status and starts Transaction Mode
PTR_TP_NORMAL	12	Terminates the transaction mode and outputs the accumulated data in the buffer

**5-5 Alignment**

- Definitions of the values required to specify alignment

[For Barcode]

Code	Value	Description
PTR_BC_LEFT	-1	Align to left
PTR_BC_CENTER	-2	Align to center
PTR_BC_RIGHT	-3	Align to right

[For Image]

Code	Value	Description
PTR_BM_LEFT	-1	Align to left
PTR_BM_CENTER	-2	Align to center
PTR_BM_RIGHT	-3	Align to right

### 5-6 Barcode type

- Definitions of the values required to specify barcode type when barcode is printed

Code	Value	Description	
PTR_BCS_UPCA	101	UPCA	
PTR_BCS_UPCE	102	UPCE	
PTR_BCS_JAN13	103	JAN13	
PTR_BCS_EAN13	103	EAN13	
PTR_BCS_JAN8	104	JAN8	
PTR_BCS_EAN8	104	EAN8	
PTR_BCS_Code93	105	Code93	
PTR_BCS_ITF	106	Interleaved 2 of 5	
PTR_BCS_Codabar	107	Codabar	
PTR_BCS_Code93	108	Code93	
PTR_BCS_Code128	109	Code 128 ※ Special Character of Code128Special Charaters	
		Special Charaters	Ascii Represntation
		Code A	{A
		Code B	{B
		Code C	{C
PTR_BCS_GS1DATABAR	110	GS1 DataBar Omnidirectional	
PTR_BCS_PDF417	201	PDF 417	
PTR_BCS_QRCODE	202	QR Code	
PTR_BCS_MAXICODE	203	MAXI Code	
PTR_BCS_DATAMATRIX	204	Data Matrix	
PTR_BCS_EAN128	120	EAN 128	

### 5-7 Barcode Text Location

- In case of barcodes supporting text printing, it specifies if the barcode text is printed, or the location of printing.

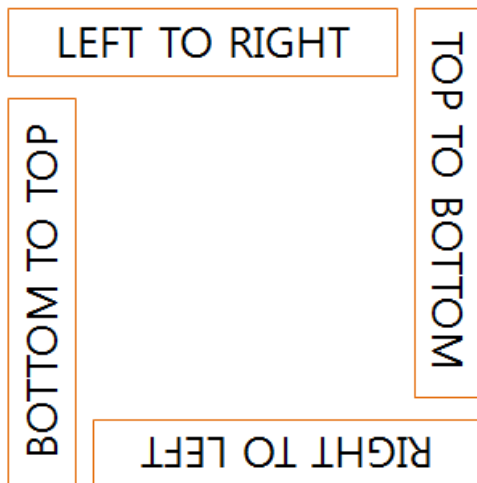
Code	Value	Description
PTR_BC_TEXT_NONE	-11	Does not print the text. Prints the barcode only.
PTR_BC_TEXT_ABOVE	-12	Prints the text at the top of the barcode.
PTR_BC_TEXT_BELOW	-13	Prints the text at the bottom of the barcode.

**5-8 Device Model Name**

Code	Description
PRODUCT_NAME_T_3II	T-3II
PRODUCT_NAME_T_4	T-4
PRODUCT_NAME_T_3III	T-3III

**5-9 Print Direction in Page Mode**

Code	Value	Description
PTR_PD_LEFT_TO_RIGHT	1	Prints left to right
PTR_PD_BOTTOM_TO_TOP	2	Prints from bottom to top
PTR_PD_RIGHT_TO_LEFT	3	Prints right to left
PTR_PD_TOP_TO_BOTTOM	4	Prints from top to bottom



**5-12 Character Set**

Code	Value	Description
CS_437_USA_STANDARD_EUROPE	437	PC437
CS_737_GREEK	737	PC737
CS_775_BALTIC	775	PC775
CS_850_MULTILINGUAL	850	PC850
CS_852_LATIN2	852	PC852
CS_855_CYRILLIC	855	PC855
CS_857_TURKISH	857	PC857
CS_858_EURO	858	PC858
CS_860_PORTUGUESE	860	PC860
CS_862_HEBREW_DOS_CODE	862	PC862
CS_863_CANADIAN_FRENCH	863	PC863
CS_864_ARABIC	864	PC864
CS_865_NORDIC	865	PC865
CS_866_CYRILLIC2	866	PC866
CS_928_GREEK	928	PC928
CS_1250_CZECH	1250	WPC1250
CS_1251_CYRILLIC	1251	WPC1251
CS_1252_LATIN1	1252	WPC1252
CS_1253_GREEK	1253	WPC1253
CS_1254_TURKISH	1254	WPC1254
CS_1255_HEBREW_NEW_CODE	1255	WPC1255
CS_1256_ARABIC	1256	WPC1256
CS_1257_BALTIC	1257	WPC1257
CS_1258_VIETNAM	1258	WPC1258
CS_FARSI	7065	FARSI
CS_KATAKANA	7565	KATAKANA
CS_KHMER_CAMBODIA	7572	KHMER
CS_THAI11	8411	THAI11
CS_THAI14	8414	THAI14
CS_THAI16	8416	THAI16
CS_THAI18	8418	THAI18
CS_THAI42	8442	THAI42
CS_KS5601	5601	KS5601
CS_BIG5	6605	BIG5
CS_GB2312	2312	GB2312
CS_SHIFT_JIS	8374	SHIFT-JIS
CS_TCVN_3_1	3031	TCVN-3(1)
CS_TCVN_3_2	3032	TCVN-3(2)

## 6. Functions by Class

### 6-1 ConfigLoader Class

- This is a class to save device setting information to be connected. The setting information manages device information through the ConfigLoader Class. The setting information includes the device name, product name, interface, etc., and if the information is not saved normally, the device cannot be connected. Before calling the Open function, this class must be called to save the setting information.



The device which is not saved with ConfigLoader cannot be connected.

#### 6-1-1 openFile()

Opens existing saved setting file.

#### [Syntax]

void openFile() throws Exception

#### [Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;

try
{
    ConfigLoader configLoader = new ConfigLoader(this);
    configLoader.openFile();
}
catch(Exception e)
{
    // Error
    e.printStackTrace();
}
```



**6-1-2 newFile()**

Creates a new setting save file. Normally, it is performed when an openFile exception occurs.

**[Syntax]**

void newFile() throws Exception

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;

try
{
    ConfigLoader configLoader = new ConfigLoader(this);
    ConfigLoader.openFile();
}
catch(Exception e)
{
    e.printStackTrace();
    configLoader.newFile();
}
```

**6-1-3 getEntries()**

Obtains saved setting information.

**[Syntax]**

List<JposEntry> getEntries() throws Exception

**[Return Values]**

Value	Description
List container of JposEntry	Listing of objects

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;

try
{
    for (Object entry : configLoader.getEntries())
    {
        JposEntry jposEntry = (JposEntry)entry;
        String strLogicalname = jposEntry.getLogicalName();
    }
}
catch(Exception e)
{
    e.printStackTrace();
}
```

**6-1-4 addEntry()**

Adds device connection information.

**[Syntax]**

void addEntry(String logicalName, int deviceCategory, String productName,  
int deviceBus, String address) throws IllegalArgumentException

**[Parameters]**

- String logicalName : Enter the logical name(nickname) of the device.
- int deviceCategory : Select the type of device.

Code	Value	Description
DEVICE_CATEGORY_CASH_DRAWER	0	Cash Drawer
DEVICE_CATEGORY_POS_PRINTER	2	POS Printer

- String productName : Device model name(Refer to "5-8 Device Model Name")
- int deviceBus : Select the interface type.

Code	Value	Description
DEVICE_BUS_BLUETOOTH	0	Bluetooth
DEVICE_BUS_ETHERNET	1	Ethernet
DEVICE_BUS_USB	2	USB
DEVICE_BUS_WIFI	3	WiFi

- String address : Enter the MAC or IP address of the device.  
(BT : MAC Address, Netwrok : IP Address)

[Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;

try
{
    ConfigLoader configLoader = new ConfigLoader(this);
    configLoader.addEntry("T-3II",
                        ConfigLoader.DEVICE_CATEGORY_POS_PRINTER,
                        ConfigLoader.PRODUCT_NAME_T_3II,
                        ConfigLoader.DEVICE_BUS_ETHERNET,
                        "192.168.0.123");
}
catch(Exception e)
{
    e.printStackTrace();
}
```

**6-1-5 removeEntry()**

Deletes saved setting information.

**[Syntax]**

boolean removeEntry(String logicalName)

**[Parameters]**

- String logicalName : Enter the logical name (nickname) of the device.

**[Return Values]**

Value	Description
true	Returned on success
false	Returned on failure

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;

try
{
    for (Object entry : configLoader.getEntries())
    {
        JposEntry jposEntry = (JposEntry)entry;
        String strLogicalname = jposEntry.getLogicalName();

        configLoader.removeEntry(strLogicalname);
    }
}
catch(Exception e)
{
    e.printStackTrace();
}
```

**6-1-6 saveFile()**

Saves changed information through addEntry and removeEntry.

**[Syntax]**

void saveFile()

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;

try
{
    for (Object entry : configLoader.getEntries())
    {
        JposEntry jposEntry = (JposEntry)entry;
        String strLogicalname = jposEntry.getLogicalName();

        configLoader.removeEntry(strLogicalname);
    }

    configLoader.addEntry("T-3II",
                        ConfigLoader.DEVICE_CATEGORY_POS_PRINTER,
                        ConfigLoader.PRODUCT_NAME_T_3II,
                        ConfigLoader.DEVICE_BUS_ETHERNET,
                        "192.168.0.123");

    configLoader.saveFile();
}
catch(Exception e)
{
    e.printStackTrace();
}
```



The device added by addEntry function must be saved using the saveFile function.

**6-2 POSPrinter Class**

- This is a class for POS printer control. Using this class, operations can be performed such as connecting/disconnecting printer and executing print jobs. It generates a JposException when an error occurs while performing a specific function. (Refer to "5-1 JposException")

**6-2-1 open()**

It Initiates the use of printer class and includes initialization operations such as memory allocation. It must be first performed to call a Method above Claim.  
Devices not saved via the ConfigLoader Class will not be opened.

**[Syntax]**

void open(String logicalDeviceName) throws JposException

**[Parameters]**

- String logicalDeviceName : Enter the name of the device to be opened.

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-2 claim()**

It attempts to actually open the port included in the device information, and includes some initialization operations such as memory allocation initialization.

It must be first performed to enable the use of device.

**[Syntax]**

void claim(int timeout) throws JposException

**[Parameters]**

- int timeout : Attempt to open the port for the time specified in this parameter.

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(3000);
}
catch(JposException e)
{
    e.printStackTrace();
}
```



**6-2-3 setDeviceEnabled()**

It includes whether the device will be used or not.

If the value of DeviceEnabled is disabled, the function may not be available.

**[Syntax]**

void setDeviceEnabled(boolean deviceEnabled) throws JposException

**[Parameters]**

- boolean deviceEnabled : Enter the device activation status.

Value	Description
true	Enabled
false	Disabled

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-4 release()**

Physically terminates use of the port of the claimed Device.  
Operations such as freeing up memory can be performed.

**[Syntax]**

void release() throws JposException

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    posPrinter.release();
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-5 close()**

Terminates use of the opened device.

Some operations such as freeing up memory may be performed.

**[Syntax]**

void close() throws JposException

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    posPrinter.close();
    posPrinter.setDeviceEnabled(false);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-6 checkHealth()**

Checks that the device is operating normally.

It can be used when Open / Claim / DeviceEnabled is normally performed.

**[Syntax]**

void checkHealth(int level) throws JposException

**[Parameters]**

- int level : fixed Value JposConst.JPOS\_CH\_INTERNAL

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.checkHealth(JposConst.JPOS_CH_INTERNAL);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

### 6-2-7 setAsyncMode()

Includes whether or not to use asynchronous mode.

If the asynchronous mode is true, the related method operates in asynchronous mode.

If false, the related method operates in synchronous mode.

#### **[Syntax]**

void setAsyncMode(boolean asyncMode) throws JposException

#### **[Parameters]**

- boolean asyncMode : Select whether or not to use asynchronous mode.

Value	Description
true	asynchronous mode
false	synchronous mode

#### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.setAsyncMode(true);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

### 6-2-8 setCharacterSet()

Sets the character set of the printer.

#### **[Syntax]**

void setCharacterSet(int characterSet) throws JposException

#### **[Parameters]**

- int characterSet : Select the character set to be set in the the printer.  
(Refer to "5-12 Character Set")

#### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.setCharacterSet(Const.CS_437_USA_STANDARD_EUROPE);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

### 6-2-9 setCharacterEncoding()

Sets the encoding of the data to be sent to the printer.

#### **[Syntax]**

void setCharacterEncoding(int characterEncoding) throws JposException

#### **[Parameters]**

- int characterEncoding : Select the data encoding type.

Code	Value	Description
CE_ASCII	0	ASCII(default)
CE_UTF8	1	UTF-8

#### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.setCharacterEncoding(Const.CE_ASCII);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-10 cutPaper()**

Cuts the paper in the models with Auto Cutter.

**[Syntax]**

void cutPaper(int percentage) throws JposException

**[Parameters]**

- int percentage : Select the Full cut / Partial cut.

It works only in the models with Auto Cutter.

Value	Description
100	Full cut
90	Partial cut

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.cutPaper(100);
}
catch(JposException e)
{
    e.printStackTrace();
}
```



**6-2-11 printBarcode()**

Prints barcodes.

**[Syntax]**

void printBarcode(int station, String data, int symbology, int height, int width, int alignment, int textPosition) throws JposException

**[Parameters]**

- int station : fixed Value PTR\_S\_RECEIPT
- String data : The data to be included in the barcode. The data allowed by the barcode type may differ.
- int symbology : Select the type of barcode. (Refer to "5-6 Barcord type")
- int height : Specify the height of the barcode.
- int width : Specify the width of the barcode.
- int alignment : Select the alignment of the barcode. (Refer to "5-5 Alignment")
- int textPosition : Determine the postion of the text to be printed with the barcode. (Refer to "5-7 Barcode Text Location")

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.printBarcode(POSPrinterConst.PTR_S_RECEIPT,
                           "123456789",
                           POSPrinterConst.PTR_BCS_QRCODE,
                           8,
                           8,
                           POSPrinterConst.PTR_BC_CENTER,
                           POSPrinterConst.PTR_BC_TEXT_BELOW);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-12 printBitmap()**

Prints image. (file printing)

**[Syntax]**

void printBitmap(int station, String fileName, int width, int alignment) throws JposException

**[Parameters]**

- int station : Set image printing options (total of 4 bytes)

Index	Description
First byte	Fixed Value : PTR_S_RECEIPT
Second byte	brightness : 0 ~ 100
Third byte	Whether to use compression algorithm (0x00 : None, 0x01 : RLE, 0x02 : LZMA)
Fourth byte	Whether to use dither 0x00 : None, 0x01 : FloydSteinberg

- String filename : Specify the path to the image file.
- int width : Specify the image width.
- int alignment : Select the image alignment. (Refer to “5-5 Alignment”)

[Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    posPrinter.printBitmap(buffer.getInt(0),
                           imagePath,
                           posPrinter.getRecLineWidth(),
                           POSPrinterConst.PTR_BM_LEFT);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-13 printBitmap()**

Prints image. (Bitmap data)

**[Syntax]**

void printBitmap(int station, Bitmap bitmap, int width, int alignment) throws JposException

**[Parameters]**

- int station : Set image printing options (total of 4 bytes)

Index	Description
First byte	Fixed Value : PTR_S_RECEIPT
Second byte	brightness : 0 ~ 100
Third byte	Whether to use compression algorithm (0x00 : None, 0x01 : RLE, 0x02 : LZMA)
Fourth byte	Whether to use dither 0x00 : None, 0x01 : FloydSteinberg

- Bitmap bitmap : Type the image data in the bitmap format.
- int width : Specify the image width.
- int alignment : Select the image alignment. (Refer to "5-5 Alignment")

[Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    posPrinter.printBitmap(buffer.getInt(0),
                           BitmapData,
                           posPrinter.getRecLineWidth(),
                           POSPrinterConst.PTR_BM_LEFT);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-14 printNormal()**

Prints text.

**[Syntax]**

void printNormal(int station, String data) throws JposException

**[Parameters]**

- int station : fixed Value PTR\_S\_RECEIPT
- String data : Specify the data to be printed. Printable characters and escape sequences, carriage returns, line feeds Data are allowed.

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, "Print Data\n");
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-15 setPageModePrintArea()**

Specifies the page mode area.

**[Syntax]**

void setPageModePrintArea(String area) throws JposException

**[Parameters]**

- String area : Specifies the print area.  
"X coordinate of area, y coordinate of area, width of area, height of area"  
ex) "0, 0, 576, 1600"

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.setPageModePrintArea("0, 0, 576, 1600");
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-16 setPageModePrintDirection()**

Specifies the direction of printing in Page Mode.

**[Syntax]**

void setPageModePrintDirection(int direction) throws JposException

**[Parameters]**

- int direction : Specify the direction of printing.(Refer to “Print Direction in Page Mode”)

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.setPageModePrintArea("0, 0, 576, 1600");
    posPrinter.setPageModePrintDirection(POSPrinterConst.PTR_PD_LEFT_TO_RIGHT);
}
catch(JposException e)
{
    e.printStackTrace();
}
```



## 6-2-17 pageModePrint()

Sets the printer to page mode.

### **[Syntax]**

void pageModePrint(int control) throws JposException

### **[Parameters]**

- int control : Specify the page mode

Code	Value	Description
PTR_PM_PAGE_MODE	1	Enable page mode
PTR_PM_NORMAL	2	Change to the normal mode and the data stored in the page mode buffer is printed.

### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.setPageModePrintArea("0, 0, 576, 1600");
    posPrinter.setPageModePrintDirection(POSPrinterConst.PTR_PD_LEFT_TO_RIGHT);
    posPrinter.setPageModePrint(POSPrinterConst.PTR_PM_PAGE_MODE);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

## 6-2-18 setPageModeHorizontalPosition()

Specifies the print start position (Horizontal).

### **[Syntax]**

void setPageModeHorizontalPosition(int position) throws JposException

### **[Parameters]**

- int position : print start position (Horizontal)

### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    // specifying of print area
    posPrinter.setPageModePrintArea("0, 0, 576, 1600");
    // specifying of print direction
    posPrinter.setPageModePrintDirection(POSPrinterConst.PTR_PD_LEFT_TO_RIGHT);
    // change to printer page mode
    posPrinter.setPageModePrint(POSPrinterConst.PTR_PM_PAGE_MODE);

    // specifying of width/height print position
    posPrinter.setPageModeHorizontalPosition(0);
    posPrinter.setPageModeVerticalPosition(0);
    // transmitting of data to be printed (image)
    posPrinter.printBitmap(buffer.getInt(0), imagePath, 384, POSPrinterConst.PTR_BM_LEFT);

    // specifying of width/height position
    posPrinter.setPageModeHorizontalPosition(100);
    posPrinter.setPageModeVerticalPosition(100);
    // transmitting of data to be printed (text)
    posPrinter.printNormar(POSPrinterConst.PTR_S_RECEIPT, "Print Data\n");

    // print start
    posPrinter.setPageModePrint(POSPrinterConst.PTR_PM_PAGE_NORMAL);
}
```

```
}  
catch(JposException e)  
{  
    e.printStackTrace();  
}
```

## 6-2-19 setPageModeVerticalPosition()

Specifies the print start position (Vertical).

### **[Syntax]**

void setPageModeVerticalPosition(int position) throws JposException

### **[Parameters]**

- int position : print start position (Vertical)

### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    // specifying of print area
    posPrinter.setPageModePrintArea("0, 0, 576, 1600");
    // specifying of print direction
    posPrinter.setPageModePrintDirection(POSPrinterConst.PTR_PD_LEFT_TO_RIGHT);
    // change to printer page mode
    posPrinter.setPageModePrint(POSPrinterConst.PTR_PM_PAGE_MODE);

    // specifying of width/height print position
    posPrinter.setPageModeHorizontalPosition(0);
    posPrinter.setPageModeVerticalPosition(0);
    // transmitting of data to be printed (image)
    posPrinter.printBitmap(buffer.getInt(0), imagePath, 384, POSPrinterConst.PTR_BM_LEFT);

    // specifying of width/height position
    posPrinter.setPageModeHorizontalPosition(100);
    posPrinter.setPageModeVerticalPosition(100);
    // transmitting of data to be printed (text)
    posPrinter.printNormar(POSPrinterConst.PTR_S_RECEIPT, "Print Data\n");
}
```

```
// print start
posPrinter.setPageModePrint(POSPrinterConst.PTR_PM_PAGE_NORMAL);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

## 6-2-20 transactionPrint()

Prints using the Transaction Mode.

### **[Syntax]**

public void transactionPrint(int station, int control) throws JposException

### **[Parameters]**

- int station : fixed Value PTR\_S\_RECEIPT
- int control : Transaction Mode(Refer to "5-4 Transaction Print")

### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    // Transaction mode start
    posPrinter.transactionPrint(POSPrinterConst.PTR_TP_TRANSACTION);

    // transmitting of data to be printed (image)
    posPrinter.printBitmap(buffer.getInt(0), imagePath, 384, POSPrinterConst.PTR_BM_LEFT);
    // transmitting of data to be printed (text)
    posPrinter.printNormar(POSPrinterConst.PTR_S_RECEIPT, "Print Data\n");

    // Transaction mode Termination (Start printing)
    posPrinter.transactionPrint(POSPrinterConst.PTR_TP_NORMAL);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-21 displayString()**

Print the text with BCD-3000.

This time, BCD-3000 should be connected to the Printer's DK port.

**[Syntax]**

public void displayString(String data) throws JposException

**[Parameters]**

- String data: Text data to be printed with BCD-3000.

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.displayString("Customer Display");
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-22 cleanScreen()**

Clean BCD-3000 screen.

This time, BCD-3000 should be connected to the Printer's DK port.

**[Syntax]**

public void cleanScreen() throws JposException

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.displayString("Customer Displa");
    posPrinter.cleanScreen();
}
catch(JposException e)
{
    e.printStackTrace();
}
```



**6-2-23 storeImageFile()**

Save the image data in the image buffer of BCD-3000.  
This time, BCD-3000 should be connected to the Printer's DK port.

**[Syntax]**

public void storeImageFile(String filename, int width, int height, int imageNumber)  
throws JposException

**[Parameters]**

- String filename: Specify the path of image file.
- int width: Specify the width of image (1 ~ 160)
- int height: Specify the height of image (1 ~ 32)
- int imageNumber: Specify the number of the image data to be saved (1 ~ 5)

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.storeImageFile(imagePath, 160, 32, 1);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-24 displayImage()**

Print the image saved in the image buffer of BCD-3000.

This time, BCD-3000 should be connected to the Printer's DK port.

**[Syntax]**

public void displayImage(int imageNumber, int xPos, int yPos) throws JposException

**[Parameters]**

- int imageNumber: Specify the number of the image data to be printed (1 ~ 5)
- int xPos: Input X coordinate to print the image on (0 ~ 159)
- int yPos: Input Y coordinate to print the image on (0 ~ 31)
- ※ The image will not be displayed when it is out of the printing area.

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.storeImageFile(imagePath, 160, 32, 1);
    posPrinter.displayImage(1, 0, 0);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

## 6-2-25 clearImage()

Delete the image saved in the image buffer of BCD-3000.

This time, BCD-3000 should be connected to the Printer's DK port.

### **[Syntax]**

public void clearImage(boolean isAll, int imageNumber) throws JposException

### **[Parameters]**

- boolean isAll: Select image buffer clear mode.

Value	Description
true	Delete all the images in image buffer.
false	Delete only the image of the specified number. The image number to be deleted should be input in imageNumber

- int imageNumber: Specify the number of the image data to be deleted (1 ~ 5)

### **[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.storeImageFile(imagePath, 160, 32, 1);
    posPrinter.displayImage(1, 0, 0);
    posPrinter.clearImage(false, 1);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

## 6-2-26 directIO()

Transmits custom data to the printer.

Transfers the data-driven response value to the directIOOccurred Event.

### [Syntax]

public void directIO(int command, int[] data, Object object) throws JposException

### [Parameters]

- int command: Selects the type of direct I/O command.

Value	Description
1	Transmit custom data.
2	Check battery status.
3	Power off time (mobile printer only): 0 to 90 (seconds)

- int[] data: Specifies the power off time (command = 3)
- Object object: Enters a custom command.

### [Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.addDirectIOListener(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    byte[] data = new byte{0x1d, 0x49, 0x43};
    posPrinter.directIO(1, null, data);
}
catch(JposException e)
{
    e.printStackTrace();
}

@Override
public void directIOOccurred(DirectIOEvent directIOEvent) {
    runOnUiThread(new Runnable()
    {
        @Override
        public void run()
        {
            Toast.makeText(MainActivity.this, new String((byte[]) directIOEvent.getObject(),
            Toast.LENGTH_SHORT).show();
        }
    });
}
```

```
    }  
    });  
}
```

**6-2-27 markFeed()**

Feeds the paper to the next print position.

**[Syntax]**

public void markFeed(int type) throws JposException

**[Parameters]**

- int type: Specifies the mark type (fixed value: 0)

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.markFeed(0);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

## 6-2-28 setBitmap()

This method saves the image in the non-volatile memory area.

### [Syntax]

void setBitmap(int bitmapNumber, int station, String fileName, int width, int alignment)  
throws JposException

### [Parameters]

- int bitmapNumber : address of memory area to save image. (0 ~ 255)
- int station : Set image printing options (total of 4 bytes)

Index	Description
First byte	Fixed Value : PTR_S_RECEIPT
Second byte	brightness : 0 ~ 100
Third byte	Reserved(0x00)
Fourth byte	Reserved(0x00)

- String filename : Specify the path to the image file.
- int width : Specify the image width.
- int alignment : Fixed Value : -1

### [Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x00); // Reserved
    buffer.put((byte) 0x00); // Reserved

    // Define NV image : Memory code 0
    posPrinter.setBitmap(0,
                        buffer.getInt(0),
                        imagePath,
                        posPrinter.getRecLineWidth(),
                        -1);

    // Print NV image : Memory code 0
    String data = ESCAPE_SEQUENCE + "0B";
```

```
        posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, data);  
    }  
    catch(JposException e)  
    {  
        e.printStackTrace();  
    }
```



## 6-2-29 printSvg()

Prints SVG file.

### [Syntax]

void printSvg(int station, String fileName, int width, int alignment, int rotate) throws JposException

### [Parameters]

- int station : Set SVG printing options (total of 4 bytes)

Index	Description
First byte	Fixed Value : PTR_S_RECEIPT
Second byte	brightness : 0 ~ 100
Third byte	Whether to use compression algorithm (RLE) (0x01 : RLE)
Fourth byte	Reserved(0x00)

- String filename : Specify the path to the SVG file.
- int width : Specify the SVG width.
- int alignment : Select the SVG alignment. (Refer to “5-5 Alignment”)
- int rotate : Select the SVG rotate.

Value	Description
0	No rotate
90	90° rotate
180	180° rotate
270	270° rotate

### [Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 50); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    posPrinter.printSvg(buffer.getInt(0),
```

```
        imagePath,  
        posPrinter.getRecLineWidth(),  
        POSPrinterConst.PTR_BM_LEFT,  
        90);  
    }  
    catch(JposException e)  
    {  
        e.printStackTrace();  
    }
```

**6-2-30 updateFirmware()**

This method sends the specified firmware binary file to the printer and updates the printer firmware. Printer should be rebooted after updating printer firmware.

**[Syntax]**

void updateFirmware(String firmwareFileName) throws JposException

**[Parameters]**

- String firmwareFileName : Absolute path fo the firmware binary file to send to printer.

**[Example]**

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    posPrinter.updateFirmware("fw.flis");
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-31 printPDFFile()**

Print the PDF file.

**[Syntax]**

void printPDFFile(int station, Uri uri, int width, int alignment, int pgae) throws JposException

**[Parameters]**

- int station : Set the image print options (consisting of 4 bytes in total)

Index	Explanation
First byte	Fixed value : PTR_S_RECEIPT
Second byte	Brightness : 0 ~ 100
Third byte	Whether to use a compression algorithm 0x00 : no compression, 0x01: RLE, 2 : LZMA
Fourth byte	Whether to use dither 0x00 : not used, 0x01 : FloydSteinberg

- Uri uri : Enter the PDF file path in URI format.
- int width : Specifies the image width.
- int alignment : Choose image alignment. ("5-5 Alignment" Reference)
- int page : Specifies the PDF page number to be printed.

[Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    Uri uri = Uri.parse("/document/primary:Download/PDF_For_Print.pdf");
    posPrinter.printPDFFile(buffer.getInt(0),
                           uri,
                           posPrinter.getRecLineWidth(),
                           POSPrinterConst.PTR_BM_LEFT,
                           1);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-2-32 printPDFFile()**

Print the PDF file.

**[Syntax]**

void printPDFFile(int station, Uri uri, int width, int alignment, int startPgae, int endPage)  
throws JposException

**[Parameters]**

- int station : Set the image print options (consisting of 4 bytes in total)

Index	Explanation
First byte	Fixed value : PTR_S_RECEIPT
Second byte	Brightness : 0 ~ 100
Third byte	Whether to use a compression algorithm 0x00 : no compression, 0x01: RLE, 2 : LZMA
Fourth byte	Whether to use dither 0x00 : not used, 0x01 : FloydSteinberg

- Uri uri : Enter the PDF file path in URI format.
- int width : Specifies the image width.
- int alignment : Choose image alignment. ("5-5 Alignment" Reference)
- int startPage : Specifies the page number from which to start printing.
- int endPage : Specifies the page number to end printing.

[Example]

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;
import com.metapace.const;

try
{
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3II");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    Uri uri = Uri.parse("/document/primary:Download/PDF_For_Print.pdf");
    posPrinter.printPDFFile(buffer.getInt(0),
                           uri,
                           posPrinter.getRecLineWidth(),
                           POSPrinterConst.PTR_BM_LEFT,
                           1,
                           3);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-3 CashDrawer Class**

- This is a class for CashDrawer control. It generates a JposException when an error occurs while performing a specific function. (Refer to “5-1 JposException”)

**6-3-1 open()**

It initiates use of CashDrawer class and includes initialization operations such as memory allocation. It must be first performed to call a Method above Claim. Devices not saved via the ConfigLoader Class will not be opened.

**[Syntax]**

void open(String logicalDeviceName) throws JposException

**[Parameters]**

- String logicalDeviceName : device name to be opened(CashDrawer)

**[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    CashDrawer cashDrawer = new CashDrawer();
    cashDrawer.open("CashDrawer");
}
catch(JposException e)
{
    e.printStackTrace();
}
```



**6-3-2 claim()**

It attempts to actually open the port included in the device information, and includes some initialization operations such as memory allocation initialization.

It must be first performed to enable the use of device.

**[Syntax]**

void claim(int timeout) throws JposException

**[Parameters]**

- int timeout : Attempt to open the port for the time specified in this parameter.

**[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    CashDrawer cashDrawer = new CashDrawer();
    cashDrawer.open("CashDrawer");
    cashDrawer.claim(3000);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

### 6-3-3 setDeviceEnabled()

It includes whether the device will be used or not.

If the value of DeviceEnabled is disabled, the function may not be available.

#### **[Syntax]**

void setDeviceEnabled(boolean deviceEnabled) throws JposException

#### **[Parameters]**

- boolean deviceEnabled : Enter the device activation status.

Value	Description
true	Enabled
false	Disabled

#### **[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    CashDrawer cashDrawer = new CashDrawer();
    cashDrawer.open("CashDrawer");
    cashDrawer.claim(3000);
    cashDrawer.setDeviceEnabled(true);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-3-4 release()**

Physically terminates the use of the port of the claimed Device.  
Operations such as freeing up memory can be performed.

**[Syntax]**

void release() throws JposException

**[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    cashDrawer.release();
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-3-5 close()**

Terminates use of the opened device.

Operations such as freeing up memory can be performed.

**[Syntax]**

void close() throws JposException

**[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    cashDrawer.close();
    cashDrawer.setDeviceEnabled(false);
}
catch(JposException e)
{
    e.printStackTrace();
}
```

**6-3-6 openDrawer()**

Opens Cash Drawer.

**[Syntax]**

void openDrawer() throws JposException

**[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    CashDrawer cashDrawer = new CashDrawer();
    cashDrawer.open("CashDrawer");
    cashDrawer.claim(3000);
    cashDrawer.setDeviceEnabled(true);

    cashDrawer.openDrawer();
}
catch(JposException e)
{
    e.printStackTrace();
}
```

### 6-3-7 getDrawerOpened()

Returns the open/close status of Cash Drawer.

#### **[Syntax]**

boolean getDrawerOpened() throws JposException

#### **[Return Values]**

Value	Description
true	Cash drawer is opened.
false	Cash drawer is closed.

#### **[Example]**

```
import jpos.JposConst;
import jpos.JposException;
import jpos.CashDrawer;
import jpos.CashDrawerConst;

try
{
    CashDrawer cashDrawer = new CashDrawer();
    cashDrawer.open("CashDrawer");
    cashDrawer.claim(3000);
    cashDrawer.setDeviceEnabled(true);

    if(cashDrawer.getDrawerOpened())
    {
        // opened
    }
    else
    {
        // closed
    }
}
catch(JposException e)
{
    e.printStackTrace();
}
```

## 7. Samples for Test

### 7-1 Text print

```
import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    // ConfigLoader creating / setting file open
    ConfigLoader ConfigLoader = new ConfigLoader(this);
    ConfigLoader.openFile();
}
catch(JposException e)
{
    e.printStackTrace();
    ConfigLoader.newFile();
}

try
{
    // Adding device information
    ConfigLoader.addEntry("T-3II",
                        ConfigLoader.DEVICE_CATEGORY_POS_PRINTER,
                        ConfigLoader.PRODUCT_NAME_T_3II,
                        ConfigLoader.DEVICE_BUS_ETHERNET,
                        "192.168.0.123");

    // Saving setting file
    ConfigLoader.saveFile();

    // printer Open/Claim/DeviceEnabled
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3III");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    // text print
    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT, "Print Data\n");
    posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT,
                        EscapeSequence.getString(7) + "www.metapace.com\n");

    posPrinter.close();
}
catch(JposException e)
{
}
```

```
e.printStackTrace();  
}
```



**7-2 Image print**

```

import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    // ConfigLoader creating / setting file open
    ConfigLoader ConfigLoader = new ConfigLoader(this);
    ConfigLoader.openFile();
}
catch(JposException e)
{
    e.printStackTrace();
    ConfigLoader.newFile();
}

try
{
    // Adding device information
    ConfigLoader.addEntry("T-3II",
                        ConfigLoader.DEVICE_CATEGORY_POS_PRINTER,
                        ConfigLoader.PRODUCT_NAME_T_3II,
                        ConfigLoader.DEVICE_BUS_ETHERNET,
                        "192.168.0.123");

    // Saving setting file
    ConfigLoader.saveFile();

    // print Open/Claim/DeviceEnabled
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3III");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    // image print
    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    posPrinter.printBitmap(buffer.getInt(0),
                          "/storage/emulated/0/kicc/sample_image.png",
                          384,
                          POSPrinterConst.PTR_BM_LEFT);

```

```
        posPrinter.close();  
    }  
    catch(JposException e)  
    {  
        e.printStackTrace();  
    }
```

**7-3 Page mode print**

```

import com.metapace.config.editor.configLoader;
import jpos.config.JposEntry;
import jpos.POSPrinter;
import jpos.POSPrinterConst;

try
{
    // ConfigLoader creating / setting file open
    ConfigLoader ConfigLoader = new ConfigLoader(this);
    ConfigLoader.openFile();
}
catch(JposException e)
{
    e.printStackTrace();
    ConfigLoader.newFile();
}

try
{
    // Adding device information
    ConfigLoader.addEntry("T-3II",
                        ConfigLoader.DEVICE_CATEGORY_POS_PRINTER,
                        ConfigLoader.PRODUCT_NAME_T_3II,
                        ConfigLoader.DEVICE_BUS_ETHERNET,
                        "192.168.0.123");

    // Saving setting file
    ConfigLoader.saveFile();

    // printer Open/Claim/DeviceEnabled
    POSPrinter posPrinter = new POSPrinter(this);
    posPrinter.open("T-3III");
    posPrinter.claim(5000);
    posPrinter.setDeviceEnabled(true);

    ByteBuffer buffer = ByteBuffer.allocate(4);
    buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);
    buffer.put((byte) 80); // brightness
    buffer.put((byte) 0x01); // compress
    buffer.put((byte) 0x00);

    // Page mode print area setting
    posPrinter.setPageModePrintArea("0,0,384,1200");
    // Page mode print direction setting
    posPrinter.setPageModePrintDirection(POSPrinterConst.PTR_PD_LEFT_TO_RIGHT)
;
    // Page mode change

```

```

posPrinter.pageModePrint(POSPrinterConst.PTR_PM_PAGE_MODE);

// print location setting : width/height
posPrinter.setPageModeHorizontalPosition(0);
posPrinter.setPageModeVerticalPosition(0);
posPrinter.printBitmap(buffer.getInt(0),
                        "/storage/emulated/0/kicc/sample_image.png",
                        384,
                        POSPrinterConst.PTR_BM_LEFT);

posPrinter.setPageModeHorizontalPosition(130);
posPrinter.setPageModeVerticalPosition(200);
posPrinter.printNormal(POSPrinterConst.PTR_S_RECEIPT,
                        EscapeSequence.getString(0) + // Normal
                        EscapeSequence.getString(7) + // Bold
                        EscapeSequence.getString(18) + //Scale_2_time_horizontally
                        EscapeSequence.getString(26) + // Scale_2_time_vertically
                        "www.metapace.com\n");

posPrinter.setPageModeHorizontalPosition(340);
posPrinter.setPageModeVerticalPosition(550);
String data = "www. metapace.com\n";
posPrinter.printBarCode(POSPrinterConst.PTR_S_RECEIPT,
                        data,
                        POSPrinterConst.PTR_BCS_QRCODE,
                        0, 6/*1to8*/,
                        POSPrinterConst.PTR_BC_LEFT,
                        POSPrinterConst.PTR_BC_TEXT_NONE);

// Start of page mode print
posPrinter.pageModePrint(POSPrinterConst.PTR_PM_NORMAL);

posPrinter.close();
}
catch(JposException e)
{
    e.printStackTrace();
}

```