PrinterSDK API Documentation

> Version: 1.6.3
> Package: `com.hspos.printerSDK`
> Entry class: `PrinterSDK`
> Constants: `PrinterConstants`
> Companion guides: PrinterSDK_使用说明书.md, PrinterSDK_函数方法说明.md
> Chinese edition: PrinterSDK_API_Documentation_zh.md

Table of Contents

1. SDK Introduction
2. Frequently Asked Questions
3. Enum & Macro (`PrinterConstants`)
4. Interface Specification

- Lifecycle & Permissions
- getInstance
- release
- onRequestPermissionsResult
- getCommunicationManager
- Port / Connection Function
- getUsbDevices
- connectUsbPrinter
- getBluetoothPairedDevices
- connectBluetoothPrinter
- connectNetworkPrinter
- connectNetworkPrinterAsync
- getSerialDevices
- connectSerialPrinter
- disconnectPrinter
- getBluetoothDevices
- getBluetoothPrinters
- startBluetoothDiscovery
- stopBluetoothDiscovery
- refreshBluetoothDevices
- getBluetoothAvailableDevices
- hasUsbPermission
- requestUsbPermission
- addBluetoothListener

- removeBluetoothListener
- Printer Status Function
- clearPrinterBuffer
- cancelPrint
- queryRTStatus
- queryTransmitStatus
- getPrinterStatus
- isPrinterOnline
- hasPaper
- isCoverOpen
- POS Function
- initPrinter
- resetPrinter
- printText
- printTextLineLeftRight
- printTextLineLeftRight
- printTextInUTF8
- printTextInGBK
- printTextInBIG5
- printTextInShiftJIS
- printTextInEUCKR
- setAlignment
- setBold
- setUnderline
- setReverse
- setLineSpacing
- setCharSpacing
- printBarcode
- printQRCode
- printQRCodeStandard
- printDoubleQRCode
- printPDF417
- printRasterImageFromFile
- printRasterImageFromFile
- printRasterImageFromFile
- printRasterImageFromData
- printRasterImageFromData
- printRasterImageFromData
- printRasterImageFromPixels
- printRasterImageFromPixels
- printRasterImageFromPixels
- cutPaper
- feedAndCutPaper
- feedLine
- feedPaper
- feedPaper

- reverseFeed
- horizontalTab
- setTabPositions
- openCashDrawer
- beep
- beep
- setPrintMode
- setSingleByteMode
- setCharacterSet
- setCharacterCodepage
- setChineseMode
- setChineseDoubleSize
- setChineseDoubleSize
- setDoubleByteEncoding
- setTextScale
- setAsciiTextFontType
- setTextUpsideDown
- setTextRotate
- setTextLineHeight
- setAsciiTextCharRightSpacing
- setKanjiTextCharSpacing
- setPrintPosition
- setPrintAreaLeftMargin
- setPrintAreaWidth
- setHorizontalAbsolutePrintPosition
- setHorizontalRelativePrintPosition
- setVerticalAbsolutePrintPosition
- setVerticalRelativePrintPosition
- setBarcodeHRIPosition
- setBarcodeHeight
- setBarcodeWidth
- setBarcodeReadableTextFontType
- setBarcodeReadableTextPosition
- printHorizontalLine
- fixedHorizontalLine
- printAndFeedDots
- sendSelfTest
- sendCommand
- clearBuffer
- cancelPrintCmd
- Page Mode Function
- selectPageMode
- exitPageMode
- selectPageModeEx
- selectPageModeEx
- selectPageModeEx

- pagePrintPage
- pageClearPage
- setMovementUnits
- pageSetPageArea
- pageSetPageDrawDirection
- setPrintAreaInPageMode
- setPrintAreaDirection
- drawLineInPageMode
- drawLineInPageMode
- estimateCode128WidthDots
- drawRectangleInPageMode
- drawRectangleBorderInPageMode
- drawFilledRectangleInPageMode
- drawTextInPageMode
- estimateTextWidthDots
- drawTextCenteredInPageMode
- drawTextCenteredInPageMode
- drawTextRightInPageMode
- drawTextRightInPageMode
- drawTextLineLeftRightInPageMode
- drawTextLineLeftRightInPageMode
- printNvBitmap
- drawNvBitmapInPageMode
- drawNvBitmapCenteredInPageMode
- drawTextUTF8InPageMode
- drawTextGBKInPageMode
- drawTextBIG5InPageMode
- drawTextShiftJISInPageMode
- drawTextEUCKRInPageMode
- drawBarcodeInPageMode
- drawBarcodeInPageMode
- drawBarcodeInPageModeWithCenteredHri
- drawQRCodeInPageMode
- drawImageFromFileInPageMode
- drawImageFromDataInPageMode
- drawImageFromPixelsInPageMode
- Label Function
- enableLabelMode
- disableLabelMode
- labelPageBegin
- backPaperToPrintPosition
- labelPagePrint
- labelPagePrint
- printLabelText
- printLabelText
- printLabelTextUTF8

- printLabelTextUTF8
- printLabelTextGBK
- printLabelTextGBK
- printLabelBarcode
- printLabelBarcode
- setLabelBarcodeTextPosition
- printLabelQRCode
- printLabelQRCode
- printLabelPDF417Code
- printLabelPDF417Code
- drawLabelImageFromFile
- drawLabelImageFromData
- drawLabelImageFromPixels
- drawLabelImageAdvanced
- drawLabelLine
- drawLabelLine
- drawLabelRect
- drawLabelRect
- drawLabelBox
- Other Function

SDK Introduction

1. Run the sample / test app first and confirm printing works before integrating into your product.
2. PrinterSDK supports thermal receipt printing (ESC/POS), page mode, and JPL label printing.
3. All public APIs are exposed through `PrinterSDK.getInstance(context)`.
4. Major modules:
   - **Port / Connection**: USB, Bluetooth, Network (TCP), Serial
   - **POS**: text, barcode, QR, image, cut, drawer, beep
   - **Page Mode**: absolute X/Y drawing
   - **Label**: JPL label layout and print
5. Typical flow: **connect → init/reset → print APIs → feed/cut → disconnect**.
6. Most print APIs return `boolean`: `true` means bytes were written to the port successfully, not that the printer mechanically finished.

```
import com.hspos.printerSDK.PrinterSDK;
import com.hspos.printerSDK.PrinterConstants;
import com.hspos.printerSDK.communication.PrinterConnection;

PrinterSDK sdk = PrinterSDK.getInstance(context);
PrinterConnection conn = sdk.connectUsbPrinter(device);
sdk.initPrinter(conn);
sdk.printTextInGBK(conn, "Hello\n");
sdk.feedAndCutPaper(conn);
sdk.disconnectPrinter(conn);
```

Frequently Asked Questions

1. **Which APIs should I use for receipt vs label paper?**
   Receipt / continuous paper → POS / Page Mode APIs. Label gap paper → enable label mode, then Label APIs.

2. **Why does send succeed but nothing prints?**
   `true` only means the write succeeded. Check paper type, label mode on/off, power, and whether `pagePrintPage` / `labelPagePrint` was called.

3. **Barcode TYPE selection**
   For ESC/POS receipt printing use `PrinterConstants.BARCODE_TYPE_*` (e.g. `BARCODE_TYPE_CODE128`). For JPL label printing use `LABEL_BARCODE_TYPE_*` (different numbering).

4. **Page mode content overlaps**
   Some firmwares ignore the high byte of `GS $` / `ESC $`. Keep coordinates within 0–255 when possible, and remember text Y is often baseline while barcode Y is top.

5. **Self-test does nothing**
   Current SDK uses `GS ( A` (`1D 28 41 02 00 00 02`). Old `DC2 T` is kept only for compatibility.

Finding Bluetooth printers (common flow)

These APIs are under **Port / Connection Function** below:

Method	Purpose
`getBluetoothPairedDevices()`	Get **paired** devices (no scan needed)
`startBluetoothDiscovery()`	**Start scanning** nearby Bluetooth devices
`stopBluetoothDiscovery()`	Stop scanning
`refreshBluetoothDevices()`	Refresh list (paired + unpaired scan)
`getBluetoothAvailableDevices()`	Get scan results (includes unpaired)
`getBluetoothPrinters()` / `getBluetoothDevices()`	Bluetooth printer list helpers
`addBluetoothListener(listener)`	Receive `onDeviceFound` while scanning
`removeBluetoothListener(listener)`	Remove listener
`connectBluetoothPrinter(device)`	Connect selected device

```
List<BluetoothDevice> paired = sdk.getBluetoothPairedDevices();

sdk.addBluetoothListener(device -> {
    Log.d("BT", "found: " + device.getName() + " " + device.getAddress());
});
sdk.startBluetoothDiscovery();
// wait a few seconds...
List<BluetoothDevice> all = sdk.getBluetoothAvailableDevices();
sdk.stopBluetoothDiscovery();

PrinterConnection conn = sdk.connectBluetoothPrinter(selectedDevice);
```

> Or call `refreshBluetoothDevices()` then `getBluetoothAvailableDevices()`.

Enum & Macro (`PrinterConstants`)

Alignment & Cut

Syntax

```
public static final int ALIGNMENT_LEFT = 0;
public static final int ALIGNMENT_CENTER = 1;
public static final int ALIGNMENT_RIGHT = 2;
public static final int CUT_FULL = 0;
public static final int CUT_PARTIAL = 1;
// 58mm 纸有效打印宽度（点，8 dot/mm）
public static final int PAPER_WIDTH_58MM = 384;
// 80mm 纸有效打印宽度（点，8 dot/mm）
public static final int PAPER_WIDTH_80MM = 576;
```

NV Bitmap (printer memory logo, FS p)

Syntax

```
// NV 位图正常尺寸
public static final int NV_BITMAP_MODE_NORMAL = 0;
// NV 位图倍宽
public static final int NV_BITMAP_MODE_DOUBLE_WIDTH = 1;
// NV 位图倍高
public static final int NV_BITMAP_MODE_DOUBLE_HEIGHT = 2;
// NV 位图倍宽倍高
public static final int NV_BITMAP_MODE_DOUBLE_BOTH = 3;
// 不打印条码可读文字（HRI）
public static final int BARCODE_HRI_NONE = 0;
// 在条码下方打印可读文字（常用）
public static final int BARCODE_HRI_BELOW = 2;
```

Barcode Types (ESC/POS GS k m)

Syntax

```
// UPC-A，北美 12 位商品码，数据需 11~12 位数字
public static final int BARCODE_TYPE_UPC_A = 0x41;
// UPC-E，北美 8 位压缩商品码
public static final int BARCODE_TYPE_UPC_E = 0x42;
// EAN-13，国际 13 位商品码，数据需 12~13 位数字
public static final int BARCODE_TYPE_EAN13 = 0x43;
// EAN-8，国际 8 位商品码，数据需 7~8 位数字
public static final int BARCODE_TYPE_EAN8 = 0x44;
// Code 39，工业标识，支持数字、大写字母及 -. $/+%
public static final int BARCODE_TYPE_CODE39 = 0x45;
// ITF（交叉二五码），数据必须为偶数位数字
public static final int BARCODE_TYPE_ITF = 0x46;
// Codabar，以 A~D 开头和结尾
public static final int BARCODE_TYPE_CODABAR = 0x47;
// Code 93，高密度编码
public static final int BARCODE_TYPE_CODE93 = 0x48;
// Code 128，通用首选。测试条码功能时推荐使用此类型，支持字母和数字。 <p>示例：{
public static final int BARCODE_TYPE_CODE128 = 0x49;
public static final int BARCODE_CODE128 = BARCODE_TYPE_CODE128;
public static final int BARCODE_CODE39 = BARCODE_TYPE_CODE39;
public static final int BARCODE_EAN13 = BARCODE_TYPE_EAN13;
```

QR Code

Syntax

```
public static final int QRCODE_SIZE_SMALL = 1;
public static final int QRCODE_SIZE_NORMAL = 2;
public static final int QRCODE_SIZE_LARGE = 3;
public static final int QR_ECC_LEVEL_L = 1;
public static final int QR_ECC_LEVEL_M = 2;
public static final int QR_ECC_LEVEL_Q = 3;
public static final int QR_ECC_LEVEL_H = 4;
```

Page Mode Direction

Syntax

```
public static final int PAGE_DIRECTION_LEFT_TO_RIGHT = 0;
public static final int PAGE_DIRECTION_BOTTOM_TO_TOP = 1;
public static final int PAGE_DIRECTION_RIGHT_TO_LEFT = 2;
public static final int PAGE_DIRECTION_TOP_TO_BOTTOM = 3;
```

Printer Status Bits

Syntax

```
// 查询失败或无响应
public static final int STATUS_OFFLINE = 0x00;
// 在线（由 DLE EOT n=1 的 bit3=0 合成）
public static final int STATUS_ONLINE = 0x01;
// 缺纸（由 DLE EOT n=4 纸尽位合成）
public static final int STATUS_PAPER_OUT = 0x02;
// 开盖（由 DLE EOT n=2 的 bit2 合成）
public static final int STATUS_COVER_OPEN = 0x04;
// GS r / DLE EOT 查询类型：打印机状态
public static final int STATUS_TYPE_PRINTER = 1;
// GS r / DLE EOT 查询类型：脱机状态
public static final int STATUS_TYPE_OFFLINE = 2;
// GS r / DLE EOT 查询类型：错误状态
public static final int STATUS_TYPE_ERROR = 3;
// GS r / DLE EOT 查询类型：纸传感器状态
public static final int STATUS_TYPE_PAPER_SENSOR = 4;
```

Cash Drawer

Syntax

```
public static final int CASH_DRAWER_PIN2 = 0;
public static final int CASH_DRAWER_PIN5 = 1;
```

Beep Defaults

Syntax

```
public static final int BEEP_DEFAULT_COUNT = 1;
// 默认蜂鸣时长参数（ESC B 的 t 值，与 {
public static final int BEEP_DEFAULT_DURATION = 100;
```

Raster Image Compression

Syntax

```
// GS v 0 标准光栅，不压缩
public static final int RASTER_COMPRESSION_GS_V0 = 0;
// US ( P 逐行 RLE（压缩算法1）；需打印机固件支持，多数机型不兼容，勿用于 AUTO
public static final int RASTER_COMPRESSION_LINE_RLE = 1;
// US ( J JBIG（压缩算法2）
public static final int RASTER_COMPRESSION_JBIG = 2;
// 自动选择传输量最小的方案
public static final int RASTER_COMPRESSION_AUTO = 3;
```

Label Mode

Syntax

```
public static final int LABEL_FONT_HEIGHT_24 = 24;
// 标签模式 Code 128 条码类型（JPL 编号 8，与 ESC/POS 的 BARCODE_TYPE_* 不同）
public static final int LABEL_BARCODE_TYPE_CODE128 = 8;
public static final int LABEL_BARCODE_HEIGHT_DEFAULT = 80;
public static final int LABEL_BARCODE_UNIT_WIDTH_2 = 2;
```

Double-byte Encoding

Syntax

```
public static final int DOUBLE_BYTE_ENCODING_GBK = 0;
public static final int DOUBLE_BYTE_ENCODING_UTF8 = 1;
public static final int DOUBLE_BYTE_ENCODING_BIG5 = 3;
```

ASCII Font

Syntax

```
public static final int ASCII_FONT_TYPE_A = 0;
public static final int ASCII_FONT_TYPE_B = 1;
```

Line Spacing

Syntax

```
public static final int LINE_SPACING_DEFAULT = 0;
public static final int LINE_SPACING_1_8 = 1;
public static final int LINE_SPACING_1_6 = 2;
```

Print Mode (ESC ! n)

Syntax

```
public static final int PRINT_MODE_NORMAL = 0;
public static final int PRINT_MODE_DOUBLE_WIDTH = 1;
public static final int PRINT_MODE_DOUBLE_HEIGHT = 2;
public static final int PRINT_MODE_DOUBLE = 3;
public static final int PRINT_MODE_BOLD = 8;
public static final int PRINT_MODE_REVERSE = 16;
public static final int PRINT_MODE_UNDERLINE = 128;
```

Chinese Double Size (FS W n)

Syntax

```
public static final int CHINESE_DOUBLE_SIZE_NORMAL = 0;
public static final int CHINESE_DOUBLE_SIZE_ON = 1;
```

ESC/POS Command Bytes

Syntax

```
public static final byte[] CMD_INIT = new byte[] { 0x1B, 0x40 };
public static final byte[] CMD_CUT = new byte[] { 0x1D, 0x56, 0x00 };
public static final byte[] CMD_FEED_LINE = new byte[] { 0x0A };
public static final byte[] CMD_FEED_PAPER = new byte[] { 0x1B, 0x4A, 0x00 };
public static final byte[] CMD_CLEAR_BUFFER = new byte[] { 0x18 };
public static final byte[] CMD_CANCEL_PRINT = new byte[] { 0x18 };
public static final byte[] CMD_RESET_PRINTER = new byte[] { 0x1B, 0x40 };
public static final byte[] CMD_SELECT_PAGE_MODE = new byte[] { 0x1B, 0x4C };
public static final byte[] CMD_EXIT_PAGE_MODE = new byte[] { 0x1B, 0x53 };
// 自检：GS ( A pL pH n m — 打印十六进制转储（Epson 兼容，多数机型有效） 旧版 DC2 T（12 54）在部分机型上无动作，已弃用。
public static final byte[] CMD_SELF_TEST = new byte[] {
        0x1D, 0x28, 0x41, 0x02, 0x00, 0x00, 0x02
    };
// 兼容旧机型的 DC2 T 自检（部分国产机）
public static final byte[] CMD_SELF_TEST_DC2_T = new byte[] { 0x12, 0x54 };
```

JPL Label Commands

Syntax

```
public static final byte[] JPL_ENABLE_LABEL_MODE = new byte[] { 0x1C, 0x4C };
public static final byte[] JPL_DISABLE_LABEL_MODE = new byte[] { 0x1C, 0x4C, 0x00 };
public static final byte[] JPL_BACK_PAPER_TO_PRINT_POSITION = new byte[] { 0x1C, 0x42 };
// 标签页打印 1 份：1A 4F 01 n（与 pageBegin/draw* 同属 1A 指令族） 注意：旧值 FS P（1C 50 01）与绘制指令体系不一致，会导致“发送成功但不出纸”。
public static final byte[] JPL_PAGE_PRINT_SINGLE = new byte[] { 0x1A, 0x4F, 0x01, 0x01 };
// 兼容旧固件的 FS P 打印（少数机型）
public static final byte[] JPL_PAGE_PRINT_SINGLE_FS = new byte[] { 0x1C, 0x50, 0x01 };
public static final byte JPL_DRAW_TEXT_BASIC = 0x00;
public static final byte JPL_DRAW_TEXT_ADVANCED = 0x01;
```

Interface Specification

Lifecycle & Permissions

getInstance

Get the PrinterSDK singleton instance.

Syntax

```
public PrinterSDK getInstance(Context context);
```

```
// Get the PrinterSDK singleton instance.
//
//
// context
// Android Context (Application or Activity).
//
// return
// PrinterSDK singleton.
```

release

Release SDK resources

Syntax

```
public void release();
```

```
// Release SDK resources
//
//
// return
// No return value.
```

onRequestPermissionsResult

On Request Permissions Result

Syntax

```
public void onRequestPermissionsResult(int requestCode, String[] permissions, int[]
grantResults);
```

```
// On Request Permissions Result
//
//
// requestCode
// Permission request code.
//
// permissions
// Permission name array.
//
// grantResults
// Grant result array.
//
// return
// No return value.
```

getCommunicationManager

Get communication manager

Syntax

```
public CommunicationManager getCommunicationManager();
```

```
// Get communication manager
//
//
// return
// CommunicationManager instance.
```

Port / Connection Function

getUsbDevices

Get Usb Devices

Syntax

```
public List<UsbDevice> getUsbDevices();
```

```
// Get Usb Devices
//
//
// return
// Device list; empty list on failure.
```

connectUsbPrinter

Connect a USB printer device (requests permission if needed).

Syntax

```
public PrinterConnection connectUsbPrinter(UsbDevice device);
```

```
// Connect a USB printer device (requests permission if needed).
//
//
// device
// Target device object.
//
// return
// Connection handle on success; null on failure.
```

getBluetoothPairedDevices

Get paired Bluetooth devices

Syntax

```
public List<BluetoothDevice> getBluetoothPairedDevices();
```

```
// Get paired Bluetooth devices
//
//
// return
// Device list; empty list on failure.
```

connectBluetoothPrinter

Connect a Bluetooth SPP printer.

Syntax

```
public PrinterConnection connectBluetoothPrinter(BluetoothDevice device);
```

```
// Connect a Bluetooth SPP printer.
//
//
// device
// Target device object.
//
// return
// Connection handle on success; null on failure.
```

connectNetworkPrinter

Connect a network printer via TCP (blocking).

Syntax

```
public PrinterConnection connectNetworkPrinter(String ip, int port);
```

```
// Connect a network printer via TCP (blocking).
//
//
// ip
// Printer IP address.
//
// port
// TCP port (usually 9100).
//
// return
// Connection handle on success; null on failure.
```

connectNetworkPrinterAsync

Connect a network printer via TCP asynchronously.

Syntax

```
public void connectNetworkPrinterAsync(String ip, int port,
CommunicationManager.NetworkConnectionCallback callback);
```

```
// Connect a network printer via TCP asynchronously.
//
//
// ip
// Printer IP address.
//
// port
// TCP port (usually 9100).
//
// callback
// Async connection callback.
//
// return
// No return value.
```

getSerialDevices

Serial port

Syntax

```
public List<String> getSerialDevices();
```

```
// Serial port
//
//
// return
// Device list; empty list on failure.
```

connectSerialPrinter

Connect a serial printer.

Syntax

```
public PrinterConnection connectSerialPrinter(String devicePath, int baudRate);
```

```
// Connect a serial printer.
//
//
// devicePath
// Serial device path, e.g. /dev/ttyS1.
//
// baudRate
// Serial baud rate, e.g. 9600.
//
// return
// Connection handle on success; null on failure.
```

disconnectPrinter

Disconnect printer

Syntax

```
public void disconnectPrinter(PrinterConnection connection);
```

```
// Disconnect printer
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// No return value.
```

getBluetoothDevices

Get Bluetooth Devices

Syntax

```
public List<BluetoothDevice> getBluetoothDevices();
```

```
// Get Bluetooth Devices
//
//
// return
// Device list; empty list on failure.
```

getBluetoothPrinters

Get Bluetooth Printers

Syntax

```
public List<BluetoothDevice> getBluetoothPrinters();
```

```
// Get Bluetooth Printers
//
//
// return
// Device list; empty list on failure.
```

startBluetoothDiscovery

Bluetooth devices

Syntax

```
public boolean startBluetoothDiscovery();
```

```
// Bluetooth devices
//
//
// return
// If the command is written successfully, returns true; otherwise false.
```

stopBluetoothDiscovery

Bluetooth devices

Syntax

```
public void stopBluetoothDiscovery();
```

```
// Bluetooth devices
//
//
// return
// No return value.
```

refreshBluetoothDevices

Get paired Bluetooth devices

Syntax

```
public boolean refreshBluetoothDevices();
```

```
// Get paired Bluetooth devices
//
//
// return
// If the command is written successfully, returns true; otherwise false.
```

getBluetoothAvailableDevices

Get paired Bluetooth devices

Syntax

```
public List<BluetoothDevice> getBluetoothAvailableDevices();
```

```
// Get paired Bluetooth devices
//
//
// return
// Device list; empty list on failure.
```

hasUsbPermission

Has Usb Permission

Syntax

```
public boolean hasUsbPermission(UsbDevice device);
```

```
// Has Usb Permission
//
//
// device
// Target device object.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

requestUsbPermission

Request Usb Permission

Syntax

```
public boolean requestUsbPermission(UsbDevice device);
```

```
// Request Usb Permission
//
//
// device
// Target device object.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

addBluetoothListener

Bluetooth devices

Syntax

```
public void
addBluetoothListener(com.hspos.printerSDK.communication.bluetooth.BluetoothPrinterManager.Bluet
oothDeviceListener listener);
```

```
// Bluetooth devices
//
//
// listener
// Bluetooth discovery listener.
//
// return
// No return value.
```

removeBluetoothListener

Bluetooth devices

Syntax

```
public void
removeBluetoothListener(com.hspos.printerSDK.communication.bluetooth.BluetoothPrinterManager.Bl
uetoothDeviceListener listener);
```

```
// Bluetooth devices
//
//
// listener
// Bluetooth discovery listener.
//
// return
// No return value.
```

Printer Status Function

clearPrinterBuffer

Clear Printer Buffer

Syntax

```
public boolean clearPrinterBuffer(PrinterConnection connection);
```

```
// Clear Printer Buffer
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

cancelPrint

Cancel print

Syntax

```
public boolean cancelPrint(PrinterConnection connection);
```

```
// Cancel print
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

queryRTStatus

Query real-time status (DLE EOT)

Syntax

```
public byte[] queryRTStatus(PrinterConnection connection, int type, int timeout);
```

```
// Query real-time status (DLE EOT)
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// type
// Type constant (barcode / cut / status, etc.).
//
// timeout
// Read timeout in milliseconds.
//
// return
// Raw status bytes on success; null on timeout / no response.
```

queryTransmitStatus

Query transmit status (GS r)

Syntax

```
public byte[] queryTransmitStatus(PrinterConnection connection, int type, int timeout);
```

```
// Query transmit status (GS r)
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// type
// Type constant (barcode / cut / status, etc.).
//
// timeout
// Read timeout in milliseconds.
//
// return
// Raw status bytes on success; null on timeout / no response.
```

getPrinterStatus

Get Printer Status

Syntax

```
public int getPrinterStatus(PrinterConnection connection);
```

```
// Get Printer Status
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// Integer result; status APIs may return STATUS_OFFLINE on failure.
```

isPrinterOnline

Check if printer is online

Syntax

```
public boolean isPrinterOnline(PrinterConnection connection);
```

```
// Check if printer is online
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

hasPaper

Check if paper is present

Syntax

```
public boolean hasPaper(PrinterConnection connection);
```

```
// Check if paper is present
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

isCoverOpen

Check if cover is open

Syntax

```
public boolean isCoverOpen(PrinterConnection connection);
```

```
// Check if cover is open
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

POS Function

initPrinter

Initialize printer (ESC @)

Syntax

```
public boolean initPrinter(PrinterConnection connection);
```

```
// Initialize printer (ESC @)
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

resetPrinter

Reset Printer

Syntax

```
public boolean resetPrinter(PrinterConnection connection);
```

```
// Reset Printer
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printText

Print text

Syntax

```
public boolean printText(PrinterConnection connection, String text);
```

```
// Print text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextLineLeftRight

Print one receipt line with left text and right-aligned text.

Syntax

```
public boolean printTextLineLeftRight(PrinterConnection connection, String left, String right,
int areaWidthDots, int charWidthDots);
```

```
// Print one receipt line with left text and right-aligned text.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// left
// Rect left.
//
// right
// Rect right.
//
// areaWidthDots
// See method description.
//
// charWidthDots
// Estimated character width in dots (Font A ≈ 12).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextLineLeftRight

Print one receipt line with left text and right-aligned text.

Syntax

```
public boolean printTextLineLeftRight(PrinterConnection connection, String left, String right);
```

```
// Print one receipt line with left text and right-aligned text.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// left
// Rect left.
//
// right
// Rect right.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextInUTF8

Print text

Syntax

```
public boolean printTextInUTF8(PrinterConnection connection, String text);
```

```
// Print text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextInGBK

Print text

Syntax

```
public boolean printTextInGBK(PrinterConnection connection, String text);
```

```
// Print text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextInBIG5

Print text

Syntax

```
public boolean printTextInBIG5(PrinterConnection connection, String text);
```

```
// Print text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextInShiftJIS

Print text

Syntax

```
public boolean printTextInShiftJIS(PrinterConnection connection, String text);
```

```
// Print text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printTextInEUCKR

Print text

Syntax

```
public boolean printTextInEUCKR(PrinterConnection connection, String text);
```

```
// Print text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setAlignment

Set alignment

Syntax

```
public boolean setAlignment(PrinterConnection connection, int alignment);
```

```
// Set alignment
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// alignment
// ALIGNMENT_LEFT / CENTER / RIGHT.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setBold

Set bold

Syntax

```
public boolean setBold(PrinterConnection connection, boolean bold);
```

```
// Set bold
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// bold
// true = bold on.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setUnderline

Set underline

Syntax

```
public boolean setUnderline(PrinterConnection connection, boolean underline);
```

```
// Set underline
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// underline
// true = underline on.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setReverse

Set reverse (white on black)

Syntax

```
public boolean setReverse(PrinterConnection connection, boolean reverse);
```

```
// Set reverse (white on black)
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// reverse
// true = white-on-black.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setLineSpacing

Set line spacing

Syntax

```
public boolean setLineSpacing(PrinterConnection connection, int spacing);
```

```
// Set line spacing
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// spacing
// Spacing value in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setCharSpacing

Set Char Spacing

Syntax

```
public boolean setCharSpacing(PrinterConnection connection, int spacing);
```

```
// Set Char Spacing
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// spacing
// Spacing value in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printBarcode

Print a 1D barcode. Use PrinterConstants.BARCODE_TYPE_* for type (recommend CODE128).

Syntax

```
public boolean printBarcode(PrinterConnection connection, int type, String data, int hri, int
height, int width);
```

```
// Print a 1D barcode. Use PrinterConstants.BARCODE_TYPE_* for type (recommend CODE128).
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// type
// Type constant (barcode / cut / status, etc.).
//
// data
// Payload data string or bytes.
//
// hri
// Barcode HRI position (BARCODE_HRI_*).
//
// height
// Height in dots.
//
// width
// Width in dots / module width.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printQRCode

Print / draw QR code

Syntax

```
public boolean printQRCode(PrinterConnection connection, String content, int size);
```

```
// Print / draw QR code
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// content
// QR / text content.
//
// size
// Module size.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printQRCodeStandard

Print / draw QR code

Syntax

```
public boolean printQRCodeStandard(PrinterConnection connection, String content, int size, int align, int eccLevel);
```

```
// Print / draw QR code
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// content
// QR / text content.
//
// size
// Module size.
//
// align
// Alignment.
//
// eccLevel
// QR ECC level (QR_ECC_LEVEL_*).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printDoubleQRCode

Print / draw QR code

Syntax

```
public boolean printDoubleQRCode(PrinterConnection connection, int n, int qr1Pos, String qr1Data, int qr1Ecc, int qr1Ver, int qr2Pos, String qr2Data, int qr2Ecc, int qr2Ver);
```

```
// Print / draw QR code
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// n
// Numeric parameter (encoding / dots / copies, depending on API).
//
// qr1Pos
// Left QR horizontal position.
//
// qr1Data
// Left QR data.
//
// qr1Ecc
// Left QR ECC.
//
// qr1Ver
// Left QR version.
//
// qr2Pos
// Right QR horizontal position.
//
// qr2Data
// Right QR data.
//
// qr2Ecc
// Right QR ECC.
//
// qr2Ver
// Right QR version.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printPDF417

Print PDF417

Syntax

```
public boolean printPDF417(PrinterConnection connection, int columns, int rows, int
moduleWidth, int rowHeight, int errorCorrectionLevel, int option, String data);
```

```
// Print PDF417
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// columns
// PDF417 columns.
//
// rows
// PDF417 rows.
//
// moduleWidth
// Module width.
//
// rowHeight
// PDF417 row height.
//
// errorCorrectionLevel
// Error correction level.
//
// option
// PDF417 option.
//
// data
// Payload data string or bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromFile

Print Raster Image From File

Syntax

```
public boolean printRasterImageFromFile(PrinterConnection connection, String filePath);
```

```
// Print Raster Image From File
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// filePath
// Image file path.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromFile

Print Raster Image From File

Syntax

```
public boolean printRasterImageFromFile(PrinterConnection connection, String filePath, int targetWidth, int targetHeight);
```

```
// Print Raster Image From File
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// filePath
// Image file path.
//
// targetWidth
// Target print width in dots.
//
// targetHeight
// Target print height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromFile

Print Raster Image From File

Syntax

```
public boolean printRasterImageFromFile(PrinterConnection connection, String filePath, int targetWidth, int targetHeight, int compressionMode);
```

```
// Print Raster Image From File
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// filePath
// Image file path.
//
// targetWidth
// Target print width in dots.
//
// targetHeight
// Target print height in dots.
//
// compressionMode
// RASTER_COMPRESSION_* constant.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromData

Print Raster Image From Data

Syntax

```
public boolean printRasterImageFromData(PrinterConnection connection, byte[] imageData);
```

```
// Print Raster Image From Data
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// imageData
// Encoded image bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromData

Print Raster Image From Data

Syntax

```
public boolean printRasterImageFromData(PrinterConnection connection, byte[] imageData, int targetWidth, int targetHeight);
```

```
// Print Raster Image From Data
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// imageData
// Encoded image bytes.
//
// targetWidth
// Target print width in dots.
//
// targetHeight
// Target print height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromData

Print Raster Image From Data

Syntax

```
public boolean printRasterImageFromData(PrinterConnection connection, byte[] imageData, int targetWidth, int targetHeight, int compressionMode);
```

```
// Print Raster Image From Data
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// imageData
// Encoded image bytes.
//
// targetWidth
// Target print width in dots.
//
// targetHeight
// Target print height in dots.
//
// compressionMode
// RASTER_COMPRESSION_* constant.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromPixels

Print Raster Image From Pixels

Syntax

```
public boolean printRasterImageFromPixels(PrinterConnection connection, int[] pixels, int width, int height);
```

```
// Print Raster Image From Pixels
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pixels
// ARGB pixel array.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromPixels

Print Raster Image From Pixels

Syntax

```
public boolean printRasterImageFromPixels(PrinterConnection connection, int[] pixels, int width, int height, int targetWidth, int targetHeight);
```

```
// Print Raster Image From Pixels
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pixels
// ARGB pixel array.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// targetWidth
// Target print width in dots.
//
// targetHeight
// Target print height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printRasterImageFromPixels

Print Raster Image From Pixels

Syntax

```
public boolean printRasterImageFromPixels(PrinterConnection connection, int[] pixels, int width, int height, int targetWidth, int targetHeight, int compressionMode);
```

```
// Print Raster Image From Pixels
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pixels
// ARGB pixel array.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// targetWidth
// Target print width in dots.
//
// targetHeight
// Target print height in dots.
//
// compressionMode
// RASTER_COMPRESSION_* constant.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

cutPaper

Cut paper

Syntax

```
public boolean cutPaper(PrinterConnection connection, int cutType);
```

```
// Cut paper
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// cutType
// CUT_FULL or CUT_PARTIAL.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

feedAndCutPaper

Cut paper

Syntax

```
public boolean feedAndCutPaper(PrinterConnection connection);
```

```
// Cut paper
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

feedLine

Feed Line

Syntax

```
public boolean feedLine(PrinterConnection connection);
```

```
// Feed Line
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

feedPaper

Feed Paper

Syntax

```
public boolean feedPaper(PrinterConnection connection, int lines);
```

```
// Feed Paper
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// lines
// Number of lines / feed amount.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

feedPaper

Feed Paper

Syntax

```
public boolean feedPaper(PrinterConnection connection);
```

```
// Feed Paper
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

reverseFeed

Feed paper

Syntax

```
public boolean reverseFeed(PrinterConnection connection, int lines);
```

```
// Feed paper
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// lines
// Number of lines / feed amount.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

horizontalTab

Horizontal tab

Syntax

```
public boolean horizontalTab(PrinterConnection connection);
```

```
// Horizontal tab
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setTabPositions

Horizontal tab

Syntax

```
public boolean setTabPositions(PrinterConnection connection, int[] positions);
```

```
// Horizontal tab
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// positions
// Horizontal tab stop positions.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

openCashDrawer

Open cash drawer

Syntax

```
public boolean openCashDrawer(PrinterConnection connection, int drawerIndex);
```

```
// Open cash drawer
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// drawerIndex
// CASH_DRAWER_PIN2 or PIN5.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

beep

Beep

Syntax

```
public boolean beep(PrinterConnection connection, int count, int duration);
```

```
// Beep
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// count
// Beep count.
//
// duration
// Beep duration parameter.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

beep

Beep

Syntax

```
public boolean beep(PrinterConnection connection);
```

```
// Beep
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setPrintMode

Set print mode

Syntax

```
public boolean setPrintMode(PrinterConnection connection, int mode);
```

```
// Set print mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// mode
// Mode flag / NV bitmap mode.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setSingleByteMode

Set single-byte mode

Syntax

```
public boolean setSingleByteMode(PrinterConnection connection);
```

```
// Set single-byte mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setCharacterSet

Set character set

Syntax

```
public boolean setCharacterSet(PrinterConnection connection, int charset);
```

```
// Set character set
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// charset
// International character set id.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setCharacterCodepage

Set character code page

Syntax

```
public boolean setCharacterCodepage(PrinterConnection connection, int codepage);
```

```
// Set character code page
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// codepage
// Character code page id.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setChineseMode

Set Chinese Mode

Syntax

```
public boolean setChineseMode(PrinterConnection connection);
```

```
// Set Chinese Mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setChineseDoubleSize

Set Chinese double width/height

Syntax

```
public boolean setChineseDoubleSize(PrinterConnection connection, boolean enable);
```

```
// Set Chinese double width/height
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// enable
// Enable flag.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setChineseDoubleSize

Set Chinese double width/height

Syntax

```
public boolean setChineseDoubleSize(PrinterConnection connection, int n);
```

```
// Set Chinese double width/height
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// n
// Numeric parameter (encoding / dots / copies, depending on API).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setDoubleByteEncoding

Set double-byte encoding

Syntax

```
public boolean setDoubleByteEncoding(PrinterConnection connection, int n);
```

```
// Set double-byte encoding
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// n
// Numeric parameter (encoding / dots / copies, depending on API).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setTextScale

Set Text Scale

Syntax

```
public boolean setTextScale(PrinterConnection connection, int widthScale, int heightScale);
```

```
// Set Text Scale
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// widthScale
// Text width scale 0–7 (0=1x).
//
// heightScale
// Text height scale 0–7 (0=1x).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setAsciiTextFontType

Set Ascii Text Font Type

Syntax

```
public boolean setAsciiTextFontType(PrinterConnection connection, int fontType);
```

```
// Set Ascii Text Font Type
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// fontType
// ASCII / barcode font type.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setTextUpsideDown

Set upside-down text

Syntax

```
public boolean setTextUpsideDown(PrinterConnection connection, boolean upsideDown);
```

```
// Set upside-down text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// upsideDown
// true = upside-down text.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setTextRotate

Set text rotate

Syntax

```
public boolean setTextRotate(PrinterConnection connection, boolean rotate);
```

```
// Set text rotate
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// rotate
// Rotate flag / label rotate 0-3.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setTextLineHeight

Set text line height

Syntax

```
public boolean setTextLineHeight(PrinterConnection connection, int lineHeight);
```

```
// Set text line height
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// lineHeight
// Line height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setAsciiTextCharRightSpacing

Set Ascii Text Char Right Spacing

Syntax

```
public boolean setAsciiTextCharRightSpacing(PrinterConnection connection, int spacing);
```

```
// Set Ascii Text Char Right Spacing
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// spacing
// Spacing value in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setKanjiTextCharSpacing

Print one line with left and right text

Syntax

```
public boolean setKanjiTextCharSpacing(PrinterConnection connection, int leftSpacing, int rightSpacing);
```

```
// Print one line with left and right text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// leftSpacing
// Left character spacing.
//
// rightSpacing
// Right character spacing.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setPrintPosition

Set Print Position

Syntax

```
public boolean setPrintPosition(PrinterConnection connection, int position);
```

```
// Set Print Position
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// position
// Absolute print position / HRI position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setPrintAreaLeftMargin

Set Print Area Left Margin

Syntax

```
public boolean setPrintAreaLeftMargin(PrinterConnection connection, int leftMargin);
```

```
// Set Print Area Left Margin
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// leftMargin
// Left margin in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setPrintAreaWidth

Set print area width

Syntax

```
public boolean setPrintAreaWidth(PrinterConnection connection, int width);
```

```
// Set print area width
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// width
// Width in dots / module width.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setHorizontalAbsolutePrintPosition

Set print position

Syntax

```
public boolean setHorizontalAbsolutePrintPosition(PrinterConnection connection, int position);
```

```
// Set print position
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// position
// Absolute print position / HRI position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setHorizontalRelativePrintPosition

Set print position

Syntax

```
public boolean setHorizontalRelativePrintPosition(PrinterConnection connection, int offset);
```

```
// Set print position
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// offset
// Relative offset in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setVerticalAbsolutePrintPosition

Set print position

Syntax

```
public boolean setVerticalAbsolutePrintPosition(PrinterConnection connection, int position);
```

```
// Set print position
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// position
// Absolute print position / HRI position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setVerticalRelativePrintPosition

Set print position

Syntax

```
public boolean setVerticalRelativePrintPosition(PrinterConnection connection, int offset);
```

```
// Set print position
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// offset
// Relative offset in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setBarcodeHRIPosition

Print / draw barcode

Syntax

```
public boolean setBarcodeHRIPosition(PrinterConnection connection, int pos);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pos
// HRI position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setBarcodeHeight

Print / draw barcode

Syntax

```
public boolean setBarcodeHeight(PrinterConnection connection, int height);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setBarcodeWidth

Print / draw barcode

Syntax

```
public boolean setBarcodeWidth(PrinterConnection connection, int width);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// width
// Width in dots / module width.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setBarcodeReadableTextFontType

Print / draw barcode

Syntax

```
public boolean setBarcodeReadableTextFontType(PrinterConnection connection, int fontType);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// fontType
// ASCII / barcode font type.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setBarcodeReadableTextPosition

Print / draw barcode

Syntax

```
public boolean setBarcodeReadableTextPosition(PrinterConnection connection, int position);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// position
// Absolute print position / HRI position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printHorizontalLine

Print horizontal line

Syntax

```
public boolean printHorizontalLine(PrinterConnection connection, int start, int end);
```

```
// Print horizontal line
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// start
// Line start position.
//
// end
// Line end position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

fixedHorizontalLine

Print horizontal line

Syntax

```
public boolean fixedHorizontalLine(PrinterConnection connection);
```

```
// Print horizontal line
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printAndFeedDots

Print And Feed Dots

Syntax

```
public boolean printAndFeedDots(PrinterConnection connection, int n);
```

```
// Print And Feed Dots
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// n
// Numeric parameter (encoding / dots / copies, depending on API).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

sendSelfTest

Send self-test command (GS (A hex dump).

Syntax

```
public boolean sendSelfTest(PrinterConnection connection);
```

```
// Send self-test command (GS ( A hex dump).
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

sendCommand

Send raw command bytes

Syntax

```
public boolean sendCommand(PrinterConnection connection, byte[] command);
```

```
// Send raw command bytes
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// command
// Raw ESC/POS or JPL command bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

clearBuffer

Clear printer buffer

Syntax

```
public boolean clearBuffer(PrinterConnection connection);
```

```
// Clear printer buffer
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

cancelPrintCmd

Cancel print

Syntax

```
public boolean cancelPrintCmd(PrinterConnection connection);
```

```
// Cancel print
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

Page Mode Function

selectPageMode

Enter page mode with default area initialization.

Syntax

```
public boolean selectPageMode(PrinterConnection connection);
```

```
// Enter page mode with default area initialization.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

exitPageMode

Page mode

Syntax

```
public boolean exitPageMode(PrinterConnection connection);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

selectPageModeEx

Enter page mode and set motion units / print area.

Syntax

```
public boolean selectPageModeEx(PrinterConnection connection, int nHorizontalMovementUnit, int nVerticalMovementUnit, int x, int y, int width, int height);
```

```
// Enter page mode and set motion units / print area.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// nHorizontalMovementUnit
// Horizontal motion unit (GS P x).
//
// nVerticalMovementUnit
// Vertical motion unit (GS P y).
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

selectPageModeEx

Enter page mode and set motion units / print area.

Syntax

```
public boolean selectPageModeEx(PrinterConnection connection, int x, int y, int width, int height);
```

```
// Enter page mode and set motion units / print area.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

selectPageModeEx

Enter page mode and set motion units / print area.

Syntax

```
public boolean selectPageModeEx(PrinterConnection connection);
```

```
// Enter page mode and set motion units / print area.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

pagePrintPage

Print page-mode buffer (FF) and typically leave page mode.

Syntax

```
public boolean pagePrintPage(PrinterConnection connection);
```

```
// Print page-mode buffer (FF) and typically leave page mode.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

pageClearPage

Page mode

Syntax

```
public boolean pageClearPage(PrinterConnection connection);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setMovementUnits

Set movement units

Syntax

```
public boolean setMovementUnits(PrinterConnection connection, int x, int y);
```

```
// Set movement units
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

pageSetPageArea

Page mode

Syntax

```
public boolean pageSetPageArea(PrinterConnection connection, int x, int y, int width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

pageSetPageDrawDirection

Page mode

Syntax

```
public boolean pageSetPageDrawDirection(PrinterConnection connection, int direction);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// direction
// PAGE_DIRECTION_*.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setPrintAreaInPageMode

Page mode

Syntax

```
public boolean setPrintAreaInPageMode(PrinterConnection connection, int x, int y, int width,
int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setPrintAreaDirection

Page mode

Syntax

```
public boolean setPrintAreaDirection(PrinterConnection connection, int direction);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// direction
// PAGE_DIRECTION_*.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLineInPageMode

Page mode

Syntax

```
public boolean drawLineInPageMode(PrinterConnection connection, int x1, int y1, int x2, int y2);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x1
// Line start X.
//
// y1
// Line start Y.
//
// x2
// Line end X.
//
// y2
// Line end Y.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLineInPageMode

Page mode

Syntax

```
public boolean drawLineInPageMode(PrinterConnection connection, int x1, int y1, int x2, int y2, int lineType);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x1
// Line start X.
//
// y1
// Line start Y.
//
// x2
// Line end X.
//
// y2
// Line end Y.
//
// lineType
// 1=solid, 2=dashed, 3=dotted.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

estimateCode128WidthDots

Print one line with left and right text

Syntax

```
public int estimateCode128WidthDots(String data, int moduleWidth);
```

```
// Print one line with left and right text
//
//
// data
// Payload data string or bytes.
//
// moduleWidth
// Module width.
//
// return
// Integer result; status APIs may return STATUS_OFFLINE on failure.
```

drawRectangleInPageMode

Page mode

Syntax

```
public boolean drawRectangleInPageMode(PrinterConnection connection, int mode, int x, int y,
int width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// mode
// Mode flag / NV bitmap mode.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawRectangleBorderInPageMode

Page mode

Syntax

```
public boolean drawRectangleBorderInPageMode(PrinterConnection connection, int x, int y, int
width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawFilledRectangleInPageMode

Page mode

Syntax

```
public boolean drawFilledRectangleInPageMode(PrinterConnection connection, int x, int y, int width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextInPageMode

Page mode

Syntax

```
public boolean drawTextInPageMode(PrinterConnection connection, int x, int y, String text);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

estimateTextWidthDots

Page mode

Syntax

```
public int estimateTextWidthDots(String text, int charWidthDots);
```

```
// Page mode
//
//
// text
// Text to print.
//
// charWidthDots
// Estimated character width in dots (Font A ≈ 12).
//
// return
// Integer result; status APIs may return STATUS_OFFLINE on failure.
```

drawTextCenteredInPageMode

Page mode

Syntax

```
public boolean drawTextCenteredInPageMode(PrinterConnection connection, int pageWidth, int y,
String text);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextCenteredInPageMode

Draw Text Centered In Page Mode

Syntax

```
public boolean drawTextCenteredInPageMode(PrinterConnection connection, int pageWidth, int y,
String text, int charWidthDots);
```

```
// Draw Text Centered In Page Mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// charWidthDots
// Estimated character width in dots (Font A ≈ 12).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextRightInPageMode

Set alignment

Syntax

```
public boolean drawTextRightInPageMode(PrinterConnection connection, int pageWidth, int y,
String text);
```

```
// Set alignment
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextRightInPageMode

Draw Text Right In Page Mode

Syntax

```
public boolean drawTextRightInPageMode(PrinterConnection connection, int pageWidth, int y,
String text, int charWidthDots);
```

```
// Draw Text Right In Page Mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// charWidthDots
// Estimated character width in dots (Font A ≈ 12).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextLineLeftRightInPageMode

Print one line with left and right text

Syntax

```
public boolean drawTextLineLeftRightInPageMode(PrinterConnection connection, int pageWidth, int
y, String left, String right);
```

```
// Print one line with left and right text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// left
// Rect left.
//
// right
// Rect right.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextLineLeftRightInPageMode

Draw Text Line Left Right In Page Mode

Syntax

```
public boolean drawTextLineLeftRightInPageMode(PrinterConnection connection, int pageWidth, int y, String left, String right, int charWidthDots);
```

```
// Draw Text Line Left Right In Page Mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// left
// Rect left.
//
// right
// Rect right.
//
// charWidthDots
// Estimated character width in dots (Font A ≈ 12).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printNvBitmap

NV bitmap

Syntax

```
public boolean printNvBitmap(PrinterConnection connection, int nvNumber, int mode);
```

```
// NV bitmap
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// nvNumber
// NV bitmap slot number (FS p).
//
// mode
// Mode flag / NV bitmap mode.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawNvBitmapInPageMode

Page mode

Syntax

```
public boolean drawNvBitmapInPageMode(PrinterConnection connection, int x, int y, int nvNumber, int mode);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// nvNumber
// NV bitmap slot number (FS p).
//
// mode
// Mode flag / NV bitmap mode.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawNvBitmapCenteredInPageMode

Page mode

Syntax

```
public boolean drawNvBitmapCenteredInPageMode(PrinterConnection connection, int pageWidth, int y, int nvNumber, int mode, int imageWidthDots);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// pageWidth
// Page / paper width in dots.
//
// y
// Y coordinate in dots.
//
// nvNumber
// NV bitmap slot number (FS p).
//
// mode
// Mode flag / NV bitmap mode.
//
// imageWidthDots
// See method description.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextUTF8InPageMode

Print text in UTF-8

Syntax

```
public boolean drawTextUTF8InPageMode(PrinterConnection connection, int x, int y, String text);
```

```
// Print text in UTF-8
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextGBKInPageMode

Print text in GBK

Syntax

```
public boolean drawTextGBKInPageMode(PrinterConnection connection, int x, int y, String text);
```

```
// Print text in GBK
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextBIG5InPageMode

Print text in BIG5

Syntax

```
public boolean drawTextBIG5InPageMode(PrinterConnection connection, int x, int y, String text);
```

```
// Print text in BIG5
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextShiftJISInPageMode

Print text in Shift-JIS

Syntax

```
public boolean drawTextShiftJISInPageMode(PrinterConnection connection, int x, int y, String text);
```

```
// Print text in Shift-JIS
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawTextEUCKRInPageMode

Print text in EUC-KR

Syntax

```
public boolean drawTextEUCKRInPageMode(PrinterConnection connection, int x, int y, String text);
```

```
// Print text in EUC-KR
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// text
// Text to print.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawBarcodeInPageMode

Print / draw barcode

Syntax

```
public boolean drawBarcodeInPageMode(PrinterConnection connection, int x, int y, int barcodeType, String data);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// barcodeType
// ESC/POS BARCODE_TYPE_* or LABEL_BARCODE_TYPE_*.
//
// data
// Payload data string or bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawBarcodeInPageMode

Print / draw barcode

Syntax

```
public boolean drawBarcodeInPageMode(PrinterConnection connection, int x, int y, int barcodeType, String data, int hriPos, int height, int width);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// barcodeType
// ESC/POS BARCODE_TYPE_* or LABEL_BARCODE_TYPE_*.
//
// data
// Payload data string or bytes.
//
// hriPos
// See method description.
//
// height
// Height in dots.
//
// width
// Width in dots / module width.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawBarcodeInPageModeWithCenteredHri

Draw Code128 barcode in page mode with manually centered HRI text.

Syntax

```
public boolean drawBarcodeInPageModeWithCenteredHri(PrinterConnection connection, int x, int y, int barcodeType, String data, int height, int width, int hriGapDots);
```

```
// Draw Code128 barcode in page mode with manually centered HRI text.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// barcodeType
// ESC/POS BARCODE_TYPE_* or LABEL_BARCODE_TYPE_*.
//
// data
// Payload data string or bytes.
//
// height
// Height in dots.
//
// width
// Width in dots / module width.
//
// hriGapDots
// Gap between barcode bottom and HRI text (may be negative).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawQRCodeInPageMode

Print / draw QR code

Syntax

```
public boolean drawQRCodeInPageMode(PrinterConnection connection, int x, int y, int version,
int eccLevel, String data);
```

```
// Print / draw QR code
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// version
// QR version / module size hint.
//
// eccLevel
// QR ECC level (QR_ECC_LEVEL_*).
//
// data
// Payload data string or bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawImageFromFileInPageMode

Page mode

Syntax

```
public boolean drawImageFromFileInPageMode(PrinterConnection connection, int x, int y, String filePath, int width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// filePath
// Image file path.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawImageFromDataInPageMode

Page mode

Syntax

```
public boolean drawImageFromDataInPageMode(PrinterConnection connection, int x, int y, byte[] imageData, int width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// imageData
// Encoded image bytes.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawImageFromPixelsInPageMode

Page mode

Syntax

```
public boolean drawImageFromPixelsInPageMode(PrinterConnection connection, int x, int y, int[] pixels, int imgWidth, int imgHeight, int width, int height);
```

```
// Page mode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// pixels
// ARGB pixel array.
//
// imgWidth
// Source image width.
//
// imgHeight
// Source image height.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

Label Function

enableLabelMode

Enable JPL label mode (FS L).

Syntax

```
public boolean enableLabelMode(PrinterConnection connection);
```

```
// Enable JPL label mode (FS L).
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

disableLabelMode

Disable JPL label mode.

Syntax

```
public boolean disableLabelMode(PrinterConnection connection);
```

```
// Disable JPL label mode.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

labelPageBegin

Begin a JPL label page and set size / rotation.

Syntax

```
public boolean labelPageBegin(PrinterConnection connection, int x, int y, int width, int height, int rotation);
```

```
// Begin a JPL label page and set size / rotation.
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// rotation
// Label page rotation: 0 or 1.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

backPaperToPrintPosition

Set print position

Syntax

```
public boolean backPaperToPrintPosition(PrinterConnection connection);
```

```
// Set print position
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

labelPagePrint

Print the current label page buffer (1A 4F).

Syntax

```
public boolean labelPagePrint(PrinterConnection connection);
```

```
// Print the current label page buffer (1A 4F).
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// return
// If the command is written successfully, returns true; otherwise false.
```

labelPagePrint

Print the current label page buffer (1A 4F).

Syntax

```
public boolean labelPagePrint(PrinterConnection connection, int printNum);
```

```
// Print the current label page buffer (1A 4F).
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// printNum
// Number of label copies.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelText

Draw label text

Syntax

```
public boolean printLabelText(PrinterConnection connection, String text, int x, int y, int fontSize);
```

```
// Draw label text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// fontSize
// Label font height (16/20/24/96).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelText

Draw label text

Syntax

```
public boolean printLabelText(PrinterConnection connection, String text, int x, int y, int fontHeight, int fontType);
```

```
// Draw label text
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// fontHeight
// Label font height (16/20/24/96).
//
// fontType
// ASCII / barcode font type.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelTextUTF8

Print text in UTF-8

Syntax

```
public boolean printLabelTextUTF8(PrinterConnection connection, String text, int x, int y);
```

```
// Print text in UTF-8
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelTextUTF8

Print text in UTF-8

Syntax

```
public boolean printLabelTextUTF8(PrinterConnection connection, String text, int x, int y, int fontHeight, int fontType);
```

```
// Print text in UTF-8
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// fontHeight
// Label font height (16/20/24/96).
//
// fontType
// ASCII / barcode font type.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelTextGBK

Print text in GBK

Syntax

```
public boolean printLabelTextGBK(PrinterConnection connection, String text, int x, int y);
```

```
// Print text in GBK
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelTextGBK

Print text in GBK

Syntax

```
public boolean printLabelTextGBK(PrinterConnection connection, String text, int x, int y, int fontHeight, int fontType);
```

```
// Print text in GBK
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// text
// Text to print.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// fontHeight
// Label font height (16/20/24/96).
//
// fontType
// ASCII / barcode font type.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelBarcode

Print / draw barcode

Syntax

```
public boolean printLabelBarcode(PrinterConnection connection, int type, String data, int x, int y, int height, int width);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// type
// Type constant (barcode / cut / status, etc.).
//
// data
// Payload data string or bytes.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// height
// Height in dots.
//
// width
// Width in dots / module width.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelBarcode

Print / draw barcode

Syntax

```
public boolean printLabelBarcode(PrinterConnection connection, int x, int y, int barcodeType, int barcodeHeight, int unitWidth, int rotate, String data);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// barcodeType
// ESC/POS BARCODE_TYPE_* or LABEL_BARCODE_TYPE_*.
//
// barcodeHeight
// Barcode height in dots.
//
// unitWidth
// Barcode / QR unit width.
//
// rotate
// Rotate flag / label rotate 0–3.
//
// data
// Payload data string or bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

setLabelBarcodeTextPosition

Print / draw barcode

Syntax

```
public boolean setLabelBarcodeTextPosition(PrinterConnection connection, int position);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// position
// Absolute print position / HRI position.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelQRCode

Print / draw QR code

Syntax

```
public boolean printLabelQRCode(PrinterConnection connection, String data, int x, int y, int size);
```

```
// Print / draw QR code
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// data
// Payload data string or bytes.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// size
// Module size.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelQRCode

Print / draw QR code

Syntax

```
public boolean printLabelQRCode(PrinterConnection connection, int x, int y, int version, int eccLevel, int unitWidth, int rotate, String data);
```

```
// Print / draw QR code
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// version
// QR version / module size hint.
//
// eccLevel
// QR ECC level (QR_ECC_LEVEL_*).
//
// unitWidth
// Barcode / QR unit width.
//
// rotate
// Rotate flag / label rotate 0–3.
//
// data
// Payload data string or bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelPDF417Code

Print / draw barcode

Syntax

```
public boolean printLabelPDF417Code(PrinterConnection connection, String data, int x, int y);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// data
// Payload data string or bytes.
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

printLabelPDF417Code

Print / draw barcode

Syntax

```
public boolean printLabelPDF417Code(PrinterConnection connection, int x, int y, int colNum, int eccLevel, int lwRatio, int unitWidth, int rotate, String data);
```

```
// Print / draw barcode
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// colNum
// PDF417 column count.
//
// eccLevel
// QR ECC level (QR_ECC_LEVEL_*).
//
// lwRatio
// PDF417 length/width ratio.
//
// unitWidth
// Barcode / QR unit width.
//
// rotate
// Rotate flag / label rotate 0–3.
//
// data
// Payload data string or bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelImageFromFile

Draw label image

Syntax

```
public boolean drawLabelImageFromFile(PrinterConnection connection, int x, int y, String imagePath);
```

```
// Draw label image
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// imagePath
// Image file path.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelImageFromData

Draw label image

Syntax

```
public boolean drawLabelImageFromData(PrinterConnection connection, int x, int y, byte[]
imageData);
```

```
// Draw label image
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// imageData
// Encoded image bytes.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelImageFromPixels

Draw label image

Syntax

```
public boolean drawLabelImageFromPixels(PrinterConnection connection, int x, int y, int[] pixels, int width, int height);
```

```
// Draw label image
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// pixels
// ARGB pixel array.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelImageAdvanced

Draw label image

Syntax

```
public boolean drawLabelImageAdvanced(PrinterConnection connection, int x, int y, int[] pixels, int width, int height, int showType);
```

```
// Draw label image
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// x
// X coordinate in dots.
//
// y
// Y coordinate in dots.
//
// pixels
// ARGB pixel array.
//
// width
// Width in dots / module width.
//
// height
// Height in dots.
//
// showType
// Image show type.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelLine

Draw label line

Syntax

```
public boolean drawLabelLine(PrinterConnection connection, int startX, int startY, int endX,
int endY);
```

```
// Draw label line
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// startX
// Line start X.
//
// startY
// Line start Y.
//
// endX
// Line end X.
//
// endY
// Line end Y.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelLine

Draw label line

Syntax

```
public boolean drawLabelLine(PrinterConnection connection, int startX, int startY, int endX,
int endY, int width, int color);
```

```
// Draw label line
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// startX
// Line start X.
//
// startY
// Line start Y.
//
// endX
// Line end X.
//
// endY
// Line end Y.
//
// width
// Width in dots / module width.
//
// color
// Draw color (0/1 depending on firmware).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelRect

Draw rectangle

Syntax

```
public boolean drawLabelRect(PrinterConnection connection, int left, int top, int right, int bottom);
```

```
// Draw rectangle
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// left
// Rect left.
//
// top
// Rect top.
//
// right
// Rect right.
//
// bottom
// Rect bottom.
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelRect

Draw rectangle

Syntax

```
public boolean drawLabelRect(PrinterConnection connection, int left, int top, int right, int bottom, int width, int color);
```

```
// Draw rectangle
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// left
// Rect left.
//
// top
// Rect top.
//
// right
// Rect right.
//
// bottom
// Rect bottom.
//
// width
// Width in dots / module width.
//
// color
// Draw color (0/1 depending on firmware).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

drawLabelBox

Draw filled box

Syntax

```
public boolean drawLabelBox(PrinterConnection connection, int left, int top, int right, int bottom, int color);
```

```
// Draw filled box
//
//
// connection
// Printer connection handle returned by connectXxx().
//
// left
// Rect left.
//
// top
// Rect top.
//
// right
// Rect right.
//
// bottom
// Rect bottom.
//
// color
// Draw color (0/1 depending on firmware).
//
// return
// If the command is written successfully, returns true; otherwise false.
```

Other Function

Notes

- Do not mix ESC/POS page mode with JPL label mode in one job without switching modes correctly.
- Label font heights supported by many firmwares: **16 / 20 / 24 / 96**.
- For 58mm paper use `PAPER_WIDTH_58MM` (384); for 80mm use `PAPER_WIDTH_80MM` (576).