

# 打印机接口说明

打印机接口说明.....	1
一 概述.....	3
二 函数说明.....	5
Port Function.....	5
Port_OpenCOMIO.....	5
Port_OpenUSBIO.....	7
Port_OpenLPTIO.....	7
Port_OpenPRNIO.....	8
Port_OpenTCPIO.....	8
Port_EnumCOM.....	9
Port_EnumUSB.....	9
Port_EnumLPT.....	9
Port_EnumPRN.....	10
Port_SetPort.....	11
Port_ClosePort.....	11
Pos Function.....	11
Pos_Reset.....	11
Pos_SelfTest.....	12
Pos_FeedLine.....	12
Pos_FeedHot.....	12
Pos_Feed_N_Line.....	13
Pos_FeedNextLable.....	13
Pos_BlackMark.....	14
Pos_Align.....	14
Pos_SetLineHeight.....	15
Pos_Text.....	15
Pos_Beep.....	17
Pos_KiskOutDrawer.....	17
Pos_FullCutPaper.....	18
Pos_HalfCutPaper.....	18
Pos_Barcode.....	18
Pos_Qrcode.....	20
Pos_EscQrcode.....	20
Pos_DoubleQrcode.....	21
Pos_ImagePrint.....	22
Pos_PrintNVLogo.....	22
Pos_QueryPrinterErr.....	23
Pos_QueryStstus.....	24
Pos_SetPrinterBaudrate.....	26

Pos_SetPrinterBasic.....	26
Page Function.....	27
Page_SelectPageMode.....	27
Page_PrintPage.....	28
Page_ExitPageMode.....	28
Page_SetVerticalAbsolutePrintPosition.....	28
Page_SetHorizontalAbsolutePrintPosition.....	29
Page_SetVerticalRelativePrintPosition.....	29
Page_SetHorizontalRelativePrintPosition.....	30
Page_SetPageModeDrawDirection.....	30
Page_SetPageArea.....	30
Page_Text.....	31
Page_Barcode.....	32
Page_Qrcode.....	33
Page_ImagePrint.....	34

# 一 概述

- 1 CsnPrinterLibs 是在 Windows 平台用 C++编写的 DLL，DLL 导出 C 风格的函数。
- 2 C#使用 SDK 的时候，需要拷贝 C#打印 Demo 目录下面的 CsnPrinterLibs.cs 文件内容。同时需要拷贝 CsnPrinterLibs.dll 到 bin\Debug 或者 bin\Release 目录下，然后参考 Demo 编写自己的打印应用程序即可。

程序编写 3 个步骤调用 枚举端口-->打开端口-->设置端口-->打印需要的内容-->关闭端口

- 3 PrinterLibs 函数有以下几类

## A Port\_XXX

以 Port 开头的函数，主要是打开端口，关闭端口，枚举端口。  
支持通过串口，并口，USB 口，网口进行打印。

## B POS\_XXX

以 POS 开头的函数，主要是封装了 ESC/POS 指令，可以控制打印机打印。

- ① 进纸系列函数可以控制打印机进纸
- ② 设置系列函数可以设置打印的格式等
- ③ 打印系列函数可以打印文本，条码，QR 码，图片等
- ④ 查询系列函数可以查询打印机状态
- ⑤ 其他函数可以控制钱箱、切刀、蜂鸣器等

## C PAGE\_XXX

以 PAGE 开头的函数，封装了页模式指令，可以控制打印机以页模式的方式打印。

- ① PAGE\_PageEnter 进入页模式
- ② PAGE\_SetPrintArea 设置页模式打印区域
- ③ PAGE\_DrawXXX 系列函数在指定区域打印
- ④ PAGE\_PagePrint 打印整个页面
- ⑤ PAGE\_PageExit 退出页模式

备注：

- ②③可以重复调用

仅支持页模式的机型可以使用这些函数

- 4 修改历史

时间	修改

## 二 函数说明

### Port Function

### Port\_OpenCOMIO

### 打开串口

#### Syntax

```
void * Port_OpenCOMIO(const char *name, unsigned int baudrate = 9600, const int flowcontrol = 0, const int parity = 0, const int databits = 8, const int stopbits = 0);
```

#### Parameters

name

端口名称，可以由 EnumCOM()获取

例如：COM1，COM2，COM3，...COM11...

baudrate

波特率

一般取 9600,19200,38400,57600,115200.

默认值 9600

需要和打印机波特率保持一致，建议使用高波特率以获得较好的打印速度

flowcontrol

流控制，各值定义如下：（默认值为 0）

值	定义
---	----

0	无流控
---	-----

1	DsrDtr
---	--------

2	CtsRts
---	--------

3	Xon/Xoff
---	----------

parity

校验位，各值定义如下：（默认值为 0）

值	定义
---	----

0	无校验
---	-----

1	奇校验
---	-----

2	偶校验
---	-----

3	标记校验
---	------

#### 4 空白校验

**databits**

数据位，范围[4,8]（默认值为 8）

**stopbits**

停止位，各值定义如下：（默认值为 0）

值      定义

0      1 位停止位

1      1.5 位停止位

2      2 位停止位

#### **Return value**

返回打开的端口句柄。非零表示打开成功，零表示打开失败。

#### **Remarks**

如果串口被占用，打开串口会失败。

如果波特率和打印机波特率不匹配，则无法打印。

## Port\_OpenUSBIO

打开 USB 端口

### Syntax

```
void * Port_OpenUSBIO(const char *name);
```

### Parameters

pName

端口名称。

可以通过 Port\_EnumUSB 来得到打印机的名称。

也可以使用任意其他字符串，这时候，如果找到 USB 打印机，会直接打开

### Return value

返回端口句柄 handle。非零表示打开成功，零表示打开失败。

### Remarks

USB 打印机接到电脑上，如果设备管理器中出现了 USB Printing Support，则可以使用该函数打开。

如果出现的是 Prolific USB-to-Serial Comm Port，则说明这是 USB 虚拟串口，需要使用 Port\_OpenCom。

## Port\_OpenLPTIO

打开并口

### Syntax

```
void * Port_OpenLPTIO(const char *name);
```

### Parameters

pName

端口名称。

例如：LPT1,LPT2,LPT3...

### Return value

返回端口句柄 handle。非零表示打开成功，零表示打开失败。

### Remarks

并口只有单向通讯，只可写不可读。

一切查询状态的函数，对并口来说均是无效的。

## Port\_OpenPRNIO

打开打印机驱动端口

### Syntax

```
void * Port_OpenPRNIO(const char *name);
```

### Parameters

pName

打印机名称。

例如：POS58 Printer

### Return value

返回端口句柄 handle。非零表示打开成功，零表示打开失败。

### Remarks

## Port\_OpenTCPIO

打开网口

### Syntax

```
void * Port_OpenTCPIO(const char *ip, const unsigned short port);
```

### Parameters

szIp

IP 地址

例如：192.168.1.87

nPort

端口号

例如：9100

### Return value

返回端口句柄 handle。非零表示打开成功，零表示打开失败。

### Remarks

PC 和打印机需要同网段的才可以连接



## Port\_EnumCOM

枚举串口

### Syntax

```
size_t Port_EnumCOM(char *buffer, size_t length);
```

### Parameters

buffer

用来保存端口列表的缓冲区

length

缓冲区字节数

### Return value

枚举到的端口数量

### Remarks

## Port\_EnumUSB

枚举 USB

### Syntax

```
size_t Port_EnumUSB(char *buffer, size_t length);
```

### Parameters

buffer

用来保存端口列表的缓冲区

length

缓冲区字节数

### Return value

枚举到的端口数量

### Remarks

## Port\_EnumLPT

枚举并口

### Syntax

```
size_t Port_EnumLPT(char *buffer, size_t length);
```

**Parameters**

buffer  
用来保存端口列表的缓冲区

length  
缓冲区字节数

**Return value**  
枚举到的端口数量

**Remarks**

**Port\_EnumPRN**

枚举打印机驱动

**Syntax**  
size\_t Port\_EnumCOM(char \*buffer, size\_t length);

**Parameters**

buffer  
用来保存端口列表的缓冲区

length  
缓冲区字节数

**Return value**  
枚举到的端口数量

**Remarks**

## Port\_SetPort

设置打印机通讯端口

### Syntax

```
bool Port_SetPort(void *handle);
```

### Parameters

handle

端口句柄。（Port\_OpenXXX 系列函数的返回值）

### Return value

返回 true 代表设置成功，false 代表设置失败。

### Remarks

## Port\_ClosePort

关闭端口

### Syntax

```
void Port_ClosePort(void *handle);
```

### Parameters

handle

端口句柄。（Port\_OpenXXX 系列函数的返回值）

### Return value

### Remarks

关闭端口

## Pos Function

## Pos\_Reset

重置打印机

### Syntax

```
bool Pos_Reset();
```

#### **Parameters**

#### **Return value**

如果指令写入成功，返回 true。否则，返回 false

#### **Remarks**

## **Pos\_SelfTest**

打印自检页

#### **Syntax**

```
bool Pos_SelfTest();
```

#### **Parameters**

#### **Return value**

如果指令写入成功，返回 true。否则，返回 false

#### **Remarks**

## **Pos\_FeedLine**

进纸一行

#### **Syntax**

```
bool Pos_FeedLine();
```

#### **Parameters**

#### **Return value**

如果指令写入成功，返回 true。否则，返回 false

#### **Remarks**

## **Pos\_FeedHot**

按 n 个加热点数进纸(一个点 0.125mm)

**Syntax**

```
bool Pos_FeedHot(int n);
```

**Parameters**

n

需要进纸的点数

**Return value**

如果指令写入成功，返回 true。否则，返回 false

**Remarks**

## Pos\_Feed\_N\_Line

按行数进纸

**Syntax**

```
bool Pos_Feed_N_Line(int n);
```

**Parameters**

n

需要进纸的行数

**Return value**

如果指令写入成功，返回 true。否则，返回 false

**Remarks**

## Pos\_FeedNextLable

进纸到下一个标签处

**Syntax**

```
bool Pos_FeedNextLable();
```

**Parameters****Return value**

如果指令写入成功，返回 true。否则，返回 false

### Remarks

该 API 只针对非标准标签指令的打印机，即执行 ESC/POS 的指令的标签机。如型号：LPM-261

## Pos\_BlackMark

进纸到下一个黑标处

### Syntax

```
bool Pos_BlackMark();
```

### Parameters

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

该 API 只针对有黑标功能的打印机。

## Pos\_Align

设置对齐方式

### Syntax

```
bool Pos_Align(int value);
```

### Parameters

value

设置对齐方式，各值定义如下：

值	定义
0	左对齐
1	居中对齐
2	右对齐

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

该 API 仅用于对齐指令操作，又需要对齐的 API，做于补充。

## Pos\_SetLineHeight

设置行高

### Syntax

```
bool Pos_SetLineHeight(int value);
```

### Parameters

value

设置行高点数(一个点 0.125mm)，范围[0,255]

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

## Pos\_Text

打印文本

### Syntax

```
bool Pos_Text(const wchar_t *prnText, int nLan, int nOrgx, int nWidthTimes, int nHeightTimes, int  
FontType, int nFontStyle);
```

### Parameters

prnText

需要打印的文本

nLan

打印的文本编码类型，各值定义如下：

值	定义
---	----

0	GBK
---	-----

1	UTF-8
---	-------

3	BIG-5
---	-------

4	SHIFT-JIS
---	-----------

5	EUC-KR
---	--------

nOrgx

打印的文本位置，各值定义如下：

值	定义
---	----

-1	左对齐
----	-----

-2	居中对齐
----	------

-3	右对齐
----	-----

>=0	在第 n 点位置开始打印
-----	--------------

**nWidthTimes**

字符宽度放大的倍数，范围[0,7]

**nHeightTimes**

字符高度放大的倍数，范围[0,7]

**FontType**

打印的字体类型，各值定义如下：

值	定义
---	----

0	12*24
---	-------

1	9*17
---	------

**nFontStyle**

打印的字体类型，各值定义如下：

值	定义
---	----

0x00	正常
------	----

0x08	加粗
------	----

0x80	1 点下划线
------	--------

0x100	2 点下划线
-------	--------

0x200	倒置打印
-------	------

0x400	反显、黑底白字
-------	---------

0x1000	每个字符顺时针旋转 90 度
--------	----------------

**Return value**

如果指令写入成功，返回 true。否则，返回 false

**Remarks**

nFontStyle 数值是可以通过 | 操作来实现多种风格同时出现的。



## Pos\_Beep

蜂鸣器鸣叫

### Syntax

```
bool Pos_Beep(unsigned char nCount, unsigned char nMillis);
```

### Parameters

nBeepCount

鸣叫次数

nMillis

蜂鸣毫秒时间，取值范围[100,900]。取整到百毫秒。

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

请确认打印机是否有蜂鸣器功能。

## Pos\_KiskOutDrawer

打开钱箱

### Syntax

```
bool Pos_KiskOutDrawer(int nId, int nHightTime = 20, int nLowTime = 60);
```

### Parameters

nId

打开钱箱，值范围[0,1]

nHightTime

高电平毫秒时间

nLowTime

低电平毫秒时间

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

请确认打印机是否有打开钱箱功能。

## Pos\_FullCutPaper

执行全切

### Syntax

```
bool Pos_FullCutPaper();
```

### Parameters

### Return value

如果指令写入成功，返回 **true**。否则，返回 **false**

### Remarks

请确认打印机是否有全切功能。

## Pos\_HalfCutPaper

执行半切

### Syntax

```
bool POS_HalfCutPaper();
```

### Parameters

### Return value

如果指令写入成功，返回 **true**。否则，返回 **false**

### Remarks

请确认打印机是否有半切功能。

## Pos\_Barcode

打印条码

### Syntax

```
bool Pos_Barcode(const char * BarcodeData, int nBarcodeType, int nOrgx, int nUnitWidth, int nUnitHeight, int nFontStyle, int FontPosition);
```

### Parameters

#### BarcodeData

打印的条码内容

#### nBarcodeType

打印的条码类型，各值定义如下：

值	类型
0x41	UPC-A
0x42	UPC-E
0x43	EAN13
0x44	EAN8
0x45	CODE39
0x46	ITF
0x47	CODABAR
0x48	CODE93

#### nOrgx

打印的条码位置，各值定义如下：

值	定义
-1	左对齐
-2	居中对齐
-3	右对齐
>=0	在第 n 点位置开始打印

#### nUnitWidth

打印的条码宽度，值范围[1,6]

#### nUnitHeight

打印的条码高度，值范围[1,255]

#### nFontStyle

可读字符（HRI）的字体类型，各值定义如下：

值	定义
0	12*24
1	9*17

#### FontPosition

可读字符（HRI）的打印位置，各值定义如下：

值	定义
0	不打印
1	条码上方
2	条码下方
3	条码上方和下方

#### Return value

返回值仅指示指令是否写入成功。返回 **true** 表示写入成功，返回 **false** 表示写入失败。

#### Remarks

**nUnitWidth** 如果超出打印边界，则不打印。

## Pos\_Qrcode

打印二维码

### Syntax

```
bool Pos_Qrcode(const wchar_t *QrcodeData, int nWidth = 2, int nVersion = 0, int nErrlevel = 4);
```

### Parameters

QrcodeData

二维码的内容

nWidth

二维码的宽度,取值范围[1,6]

二维码单元宽度越大,QR 码越大。

nVersion

二维码的规格,取值范围[0,16],0 表示自动计算版本。

二维码码版本越大，能编码的字符就越多，QR 码也越大。

nErrlevel

二维码纠错等级,取值[1,4]

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

如果打印的二维码超出打印边界，则不打印。

## Pos\_EscQrcode

ESC/POS 版本二维码

### Syntax

```
bool Pos_EscQrcode(const wchar_t *QrcodeData, int nWidth = 4, int nErrlevel = 4);
```

### Parameters

QrcodeData

二维码内容。

nWidth

二维码的宽度,取值范围[1,16]

二维码单元宽度越大,QR 码越大。

nErrlevel

二维码纠错等级,取值[1,4]。

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

请确认二维码大小,因为该 API 超出也会打印,会导致无法正常扫描。

## Pos\_DoubleQrcode

打印双二维码

### Syntax

```
bool Pos_DoubleQrcode(const wchar_t *QrcodeData1,int QR1Position,int QR1Version, int QR1Ecc, const wchar_t *QrcodeData2, int QR2Position, int QR2Version, int QR2Ecc, int ModuleSize);
```

### Parameters

QrcodeData1

第一个二维码的内容。

QR1Position

第一个二维码开始打印的位置

QR1Version

第一个二维码的规格，范围[0，19]

QR1Ecc

第一个二维码的纠错等级，范围[0,3]

QrcodeData2

第二个二维码的内容。

QR2Position

第二个二维码开始打印的位置

QR2Version

第二个二维码的规格。[0,19]

QR2Ecc

第二个二维码的纠错等级,范围[0,3]

ModuleSize

二维码模块的大小，范围[1，8]

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

如果打印失败，请注意检查第二个二维码打印的位置是否与第一个重叠或者打。

## Pos\_ImagePrint

打印图片

### Syntax

```
bool Pos_ImagePrint(const wchar_t *FileName, int nWidth = 384, int nBinaryAlgorithm = 0);
```

### Parameters

FileName

图片路径。

nWidth

指定打印机打印该图片的宽度(像素)

2 寸打印机最大值为 384

3 寸打印机最大值为 576

nBinaryAlgorithm

图片计算的模式，各值定义如下：

值	定义
---	----

0	抖动算法      该算法对彩色图片打印效果较好
---	--------------------------

1	平均阈值算法    该算法对纯文字图片打印效果较好
---	---------------------------

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

## Pos\_PrintNVLogo

打印预载的 Logo

### Syntax

```
bool Pos_PrintNVLogo(unsigned short nLogo, unsigned short nWidth = 0);
```

### Parameters

n

打印第 n 张 Logo,范围[1,9]

nMode

指定打印 Logo 的模式，值定义如下：

值	定义
---	----

0	普通
---	----

1	倍宽
---	----

2	倍高
---	----

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

如果打印机中无预载的 Logo 则不打印。请用打印机工具预载。

## Pos\_QueryPrinterErr

查询打印机错误

### Syntax

```
int Pos_QueryPrinterErr(unsigned long nTimeout = 3000);
```

### Parameters

nTimeout

超时的毫秒数。

### Return value

返回错误的值，各值的定义为：

值	类型
1	打印机正常
-1	打印机脱机
-2	打印机上盖打开
-3	打印机缺纸
-4	打印机切刀异常
-5	打印机头片温度过高
-6	查询失败

### Remarks

该 API 无法一次返回多个异常状态。如需获取多个异常，请使用 Pos\_QueryStstus 函数自行实现。

# Pos\_QueryStstus

查询打印机状态

## Syntax

```
bool Pos_QueryStstus(char *rBuffer, int type, unsigned long nTimeout);
```

## Parameters

rBuffer

储存打印机返回的状态

type

查询的数据表。[1,4]。具体表单请看说明文档。

各 type 含义见下表:

type=1: 打印机状态

位	0/ 1	十六进制码	十进制码	功能
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2	0	00	0	一个或两个钱箱打开 (没有钱箱的机器该位固定为零)
	1	04	4	两个钱箱都关闭
3	0	00	0	联机
	1	08	8	脱机
4	1	10	16	固定为 1
5, 6		--	--	未定义
7	0	00	00	纸已撕走
	1	80	96	纸未撕走

type=2: 传送脱机状态

位	0/ 1	十六进制码	十进制码	功能
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2	0	00	0	上盖关
	1	04	4	上盖开
3	0	00	0	未按走纸键
	1	08	8	按下走纸键
4	1	10	16	固定为 1
5	0	00	0	打印机不缺纸



	1	20	32	打印机缺纸
6	0	00	00	没有出错情况
	1	40	64	有错误情况
7	0	00	0	固定为 0

type=3: 传送错误状态

位	0/ 1	十六进制码	十进制码	功能
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2		--	--	未定义
3	0	00	0	切刀无错误
	1	08	8	切刀有错误
4	1	10	16	固定为 1
5	0	00	0	无不可恢复错误
	1	20	32	有不可恢复错误
6	0	00	00	打印头温度和电压正常
	1	40	64	打印头温度或电压超出范围
7	0	00	0	固定为 0

type=4: 传送纸传感器状态

位	0/ 1	十六进制码	十进制码	功能
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2, 3	0	00	0	有纸
	1	0C	12	纸将近
4	1	10	16	固定为 1
5, 6	0	00	0	有纸
	1	60	96	纸尽
7	0	00	0	固定为 0

nTimeout

超时的毫秒数。

**Return value**

如果指令写入成功，返回 true。否则，返回 false

**Remarks**

## Pos\_SetPrinterBaudrate

设置打印机波特率

### Syntax

```
bool Pos_SetPrinterBaudrate(int nBaudrate);
```

### Parameters

nBaudrate

设置打印机的波特率。

例如：9600 19200 38400 57600 115200

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

如果用串口连接时，设置完波特率后，请重启打印机后，重新执行 OpenCOM。

## Pos\_SetPrinterBasic

设置打印机基本参数

### Syntax

```
bool Pos_SetPrinterBasic(int nFontStyle, int nDensity, int nLine, int nBeep, int nCut);
```

### Parameters

nFontStyle

设置字体规格,各值定义如下:

值	字体
0	9*17
1	12*24
2	9*24
3	16*18

nDensity

设置浓度,各值定义如下:

值	字体
0	微淡
1	正常
2	微浓

3            高浓度

#### nLine

设置进纸模式,各值定义如下:

值	模式
---	----

0	0x0A
---	------

1	0x0D
---	------

#### nBeep

是否启用蜂鸣器,各值定义如下:

值	开关
---	----

0	关闭蜂鸣器
---	-------

1	开启蜂鸣器
---	-------

#### nCut

是否启用蜂鸣器,各值定义如下:

值	开关
---	----

0	关闭切刀功能
---	--------

1	开启切刀功能
---	--------

#### Return value

如果指令写入成功, 返回 **true**。否则, 返回 **false**

#### Remarks

字体规格仅设置非双字节文本,中文日文韩文等字体均为 **24\*24** 无法修改。

蜂鸣器开关请确认使用的机型是否带有蜂鸣器功能

切刀开关请确认使用的机型是否带有切刀功能

## Page Function

### Page\_SelectPageMode

选择页模式

#### Syntax

```
bool Page_SelectPageMode();
```

#### Parameters

#### Return value

如果指令写入成功, 返回 **true**。否则, 返回 **false**

#### Remarks

请确认打印机是否有页模式功能,页模式下不会直接打印,需要在数据填充后调用 **Page\_PrintPage**。

## Page\_PrintPage

打印页模式下的内容

### Syntax

```
bool Page_PrintPage();
```

### Parameters

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

请确认打印机是否有页模式功能,页模式下不会直接打印,需要在数据填充后调用 Page\_PrintPage。

## Page\_ExitPageMode

退出页模式

### Syntax

```
bool Page_ExitPageMode();
```

### Parameters

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

## Page\_SetVerticalAbsolutePrintPosition

设置纵向绝对打印位置

### Syntax

```
bool Page_SetVerticalAbsolutePrintPosition(unsigned short nPosition);
```

### Parameters

nPosition

打印位置

**Return value**

如果指令写入成功，返回 **true**。否则，返回 **false**

**Remarks**

## Page\_SetHorizontalAbsolutePrintPosition

设置横向绝对打印位置

**Syntax**

```
bool Page_SetHorizontalAbsolutePrintPosition(unsigned short nPosition);
```

**Parameters**

nPosition

打印位置

**Return value**

如果指令写入成功，返回 **true**。否则，返回 **false**

**Remarks**

## Page\_SetVerticalRelativePrintPosition

设置纵向相对打印位置

**Syntax**

```
bool Page_SetVerticalRelativePrintPosition(unsigned short nPosition);
```

**Parameters**

nPosition

打印位置

**Return value**

如果指令写入成功，返回 **true**。否则，返回 **false**

**Remarks**

## Page\_SetHorizontalRelativePrintPosition

设置横向相对打印位置

### Syntax

```
bool Page_SetHorizontalRelativePrintPosition(unsigned short nPosition);
```

### Parameters

nPosition

打印位置

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

## Page\_SetPageModeDrawDirection

页模式下设置打印方向

### Syntax

```
bool Page_SetPageModeDrawDirection(unsigned short nPosition);
```

### Parameters

nPosition

打印区域方向，各值定义如下：

- 0 从左到右
- 1 从下到上
- 2 从右到左
- 3 从上到下

### Return value

如果指令写入成功，返回 true。否则，返回 false

### Remarks

## Page\_SetPageArea

页模式下设置页区域

### Syntax

```
bool Page_SetPageArea(unsigned short x, unsigned short y, unsigned short w, unsigned short h);
```

**Parameters**

- x  
    横向起始位置
- y  
    纵向起始位置
- w  
    打印区域宽度
- h  
    打印区域高度

**Return value**

如果指令写入成功，返回 true。否则，返回 false

**Remarks**

**Page\_Text**

页模式下打印文本

**Syntax**

bool Page\_Text(const wchar\_t \*prnText, int nLan, int nOrgx, int nWidthTimes, int nHeightTimes, int FontType, int nFontStyle);

**Parameters**

- prnText  
    需要打印的文本
- nLan  
    打印的文本编码类型，各值定义如下：  
    值    定义  
    0    GBK  
    1    UTF-8  
    3    BIG-5  
    4    SHIFT-JIS  
    5    EUC-KR
- nOrgx  
    打印的文本位置，各值定义如下：  
    值    定义  
    -1    左对齐  
    -2    居中对齐  
    -3    右对齐  
    >=0    在第 n 点位置开始打印
- nWidthTimes

字符放大的倍数，范围[0,7]

nHeightTimes

字符放大的倍数，范围[0,7]

FontType

打印的字体类型，各值定义如下：

值	定义
0	12*24
1	9*17

nFontStyle

打印的字体类型，各值定义如下：

值	定义
0x00	正常
0x08	加粗
0x80	1 点下划线
0x100	2 点下划线
0x200	倒置打印
0x400	反显、黑底白字
0x1000	每个字符顺时针旋转 90 度

**Return value**

如果指令写入成功，返回 true。否则，返回 false

**Remark**

nFontStyle 数值是可以通过&操作来实现多种风格同时出现的。

## Page\_Barcode

页模式打印条码

**Syntax**

```
bool Page_Barcode(const char * BarcodeData, int nBarcodeType, int nOrgx, int nUnitWidth, int nUnitHeight, int nFontStyle, int FontPosition);
```

**Parameters**

BarcodeData

打印的条码内容

nBarcodeType

打印的条码类型，各值定义如下：

值	类型
0x41	UPC-A
0x42	UPC-E
0x43	EAN13
0x44	EAN8
0x45	CODE39



0x46	ITF
0x47	CODABAR
0x48	CODE93

#### nOrgx

打印的条码位置，各值定义如下：

值	定义
-1	左对齐
-2	居中对齐
-3	右对齐
>=0	在第 n 点位置开始打印

#### nUnitWidth

打印的条码宽度，值范围[1,6]

#### nUnitHeight

打印的条码高度，值范围[1,255]

#### nFontStyle

可读字符（HRI）的字体类型，各值定义如下：

值	定义
0	12*24
1	9*17

#### FontPosition

可读字符（HRI）的打印位置，各值定义如下：

值	定义
0	不打印
1	条码上方
2	条码下方
3	条码上方和下方

#### Return value

如果指令写入成功，返回 true。否则，返回 false

#### Remarks

nUnitWidth 如果超出打印边界，则不打印。

## Page\_Qrcode

页模式下打印二维码

#### Syntax

```
bool Page_Qrcode(const wchar_t *QrcodeData, int nWidth = 4, int nErrlevel = 4);
```

#### Parameters

QrcodeData

二维码内容。

nWidth

二维码的宽度,取值范围[1,16]  
二维码单元宽度越大,QR 码越大。

nErrlevel

二维码纠错等级,取值[1,4]。

return

返回值仅指示指令是否写入成功。返回 true 表示写入成功，返回 false 表示写入失败。

#### Return value

如果指令写入成功，返回 true。否则，返回 false

#### Remarks

请确认二维码大小,因为该 API 超出也会打印,会导致无法正常扫描。

## Page\_ImagePrint

页模式下打印图片

#### Syntax

```
bool Page_ImagePrint(const wchar_t *FileName, int nWidth = 384, int nBinaryAlgorithm = 0);
```

#### Parameters

FileName

图片路径。

nWidth

指定打印机打印该图片的宽度(像素)

2 寸打印机最大值为 384

3 寸打印机最大值为 576

nBinaryAlgorithm

图片计算的模式，各值定义如下：

值	定义
0	抖动算法
1	平均阈值算法

该算法对彩色图片打印效果较好

该算法对纯文字图片打印效果较好

#### Return value

如果指令写入成功，返回 true。否则，返回 false

#### Remarks



