

Android ZPL Program Manual

v3.4.1

(Note: Please use the PDF left navigation bar when browsing)

1. Instruction

This manual describes how to print labels with ZPL instructions. Constant variables are defined in ZPLConst class.

2. ZPLPrinter

2.1. ZPLPrinter

Constructor to create print objects.

```
ZPLPrinter(IDeviceConnection connection)
```

[Parameter]

➤ connection

Connected object, available via POSConnect.createDevice(deviceType).

2.2. addStart

This method is used at the beginning of the label

```
ZPLPrinter addStart()
```

[Return]

ZPLPrinter Instance

2.3. addEnd

End of label format. After calling this method, the label is printed.

```
ZPLPrinter addEnd()
```

[Return]

ZPLPrinter Instance

2.4. addText

text printing

```
ZPLPrinter addText(int x, int y, String fontName, String rotation, int sizeW, int sizeH, String content)
```

```
ZPLPrinter addText(int x, int y, char fontName, int sizeW, int sizeH, String content)
```

```
ZPLPrinter addText(int x, int y, char fontName, String content)
```

```
ZPLPrinter addText(int x, int y, String content)
```

[Parameter]

➤ x

the starting x value of the text

➤ y

the starting y value of the text

➤ font

The font type of the text, the default is FNT_F.

Variable	Description
FNT_A	9 x 5
FNT_B	11x7
FNT_C、FNT_D	18x10
FNT_E	28x15
FNT_F	26x13
FNT_G	60x40
FNT_O	15*12

For other fonts, please use custom names.

➤ sizeW

The effective width of the text, the default is the base size. Please use an integer multiple of the base size.

➤ sizeH

The effective height of the text. Default is base size. Please use an integer multiple of the base size.

➤ rotation

Clockwise rotation angle, default ROTATION_0

Variable	Description
ROTATION_0	No rotation
ROTATION_90	Rotate 90 degrees clockwise
ROTATION_180	Rotate 180 degrees clockwise
ROTATION_270	Rotate 270 degrees clockwise

➤ content

text content

[Return]

ZPLPrinter Instance

2.5. addTextBlock

The method is used to prints a text block with defined width and height. The text block has an automatic wordwrap function. If the text exceeds the block height, the text is truncated.

ZPLPrinter addTextBlock(int x, int y, char fontName, String rotation, int sizeW, int sizeH, int textblockWidth, int textblockHeight, String content)

ZPLPrinter addTextBlock(int x, int y, char fontName, int sizeW, int sizeH, int textblockWidth, int textblockHeight, String content)

ZPLPrinter addTextBlock(int x, int y, char fontName, int textblockWidth, int textblockHeight, String content)

ZPLPrinter addTextBlock(int x, int y, int textblockWidth, int textblockHeight, String content)

[Parameter]

➤ x

the starting x value of the text

➤ y

the starting y value of the text

➤ font

The font type of the text, the default is FNT_F.

Variable	Description
FNT_A	9 x 5
FNT_B	11x7
FNT_C、FNT_D	18x10
FNT_E	28x15
FNT_F	26x13
FNT_G	60x40
FNT_O	15*12

For other fonts, please use custom names.

➤ sizeW

The effective width of the text, the default is the base size. Please use an integer multiple of the base size.

➤ sizeH

The effective height of the text. Default is base size. Please use an integer multiple of the base size.

➤ rotation

Clockwise rotation angle, default ROTATION_0

Variable	Description
----------	-------------

ROTATION_0	No rotation
ROTATION_90	Rotate 90 degrees clockwise
ROTATION_180	Rotate 180 degrees clockwise
ROTATION_270	Rotate 270 degrees clockwise

- textblockWidth
block width in dots
- textblockHeight
block height in dots
- content
text content

[Return]

ZPLPrinter Instance

2.6. setCustomFont

Set custom font. After the machine is powered off, the settings will be invalid.

ZPLPrinter setCustomFont(String font, char alias, int codePage)

[Parameter]

- font
Font name and suffix of font library, for example: LZHONGHEI.TTF
- alias
Font alias, corresponding to fontName in addText. Range: A to Z and 0 to 9.
- codePage

Character Encoding

Variable	Description
CODE_PAGE_UTF8	Unicode (UTF-8 encoding) - Unicode Character Set
CODE_PAGE_UTF16	Unicode (UTF-16 Big-Endian encoding) - Unicode Character Set
CODE_PAGE_UTF16_2	Unicode (UTF-16 Little-Endian encoding) - Unicode Character Set
CODE_PAGE_USA1	Single Byte Encoding - U.S.A. 1 Character Set
CODE_PAGE_USA2	Single Byte Encoding - U.S.A. 2 Character Set
CODE_PAGE_UK	Single Byte Encoding - U.K. Character Set
CODE_PAGE_NL	Single Byte Encoding - Holland Character Set
CODE_PAGE_DK	Single Byte Encoding - Denmark/Norway Character Set
CODE_PAGE_SWEDE	Single Byte Encoding - Sweden/Finland Character Set
CODE_PAGE_GER	Single Byte Encoding - Germany Character Set
CODE_PAGE_FR1	Single Byte Encoding - France 1 Character Set
CODE_PAGE_FR2	Single Byte Encoding - France 2 Character Set
CODE_PAGE_ITA	Single Byte Encoding - Italy Character Set

CODE_PAGE_ES	Single Byte Encoding - Spain Character Set
CODE_PAGE_JA	Single Byte Encoding - Japan (ASCII with Yen symbol) Character Set

[Return]

ZPLPrinter Instance

2.7. setPrinterWidth

Set Printer Width

ZPLPrinter setPrinterWidth(int width)

[Parameter]

➤ width
label width (in dots)

[Return]

ZPLPrinter Instance

2.8. setLabelSize

The method defines the length of the label.

ZPLPrinter setLabelLength(int length)

[Parameter]

➤ length
label length(in dots)

[Return]

ZPLPrinter Instance

2.9. addReverse

Area Reverse

ZPLPrinter addReverse(int x, int y, int width, int height)

ZPLPrinter addReverse(int x, int y, int width, int height, int radius)

[Parameter]

➤ x

Start x value of the area

➤ y

Start y value of the area

➤ width

Area width

➤ height

Area height

➤ radius

degree of cornerrounding. Range: 0(no rounding) to 8 (heaviest rounding). Default is 0.

[Return]

ZPLPrinter Instance

2.10. addBox

The method is used to draw boxes and lines as part of a label format.

ZPLPrinter addBox(int x, int y, int width, int height, int thickness)

ZPLPrinter addBox(int x, int y, int width, int height, int thickness, int radius)

[Parameter]

➤ x

Start x value of the box

➤ y

Start y value of the box

➤ width

box width (in dots)

➤ height

box height (in dots)

➤ thickness

border thickness (in dots)

➤ radius

degree of cornerrounding, Range: 0(no rounding) to 8 (heaviest rounding), Default is 0.

[Return]

ZPLPrinter Instance

2.11. addGraphicDiagonalLine

The method is used to draw diagonals.

ZPLPrinter addGraphicDiagonalLine(int x, int y, char orientation, int width, int height, int thickness)

[Parameter]

➤ x
Horizontal starting position

➤ y
Vertical starting position

➤ orientation
The direction of the diagonal.

Variable	Description
R (or /)	right slanted diagonal
L (or \)	left slanted diagonal

➤ width
The width of the box (range: 1-32000, unit: dot).
➤ height
The height of the box (range: 1-32000, unit: dot).
➤ thickness
Boundary thickness (range: 1-32000, unit: dot).

[Return]

ZPLPrinter Instance

2.12. addGraphicEllipse

The method is used to draw a graphical ellipse.

ZPLPrinter addGraphicEllipse(int x, int y, int width, int height, int thickness)

[Parameter]

➤ x
Horizontal starting position

➤ y
Vertical starting position

➤ width
Ellipse width (range: 3-4095, unit: dot).
➤ height
Ellipse height (range: 3-4095, unit: dot).

➤ thickness

Boundary thickness (range: 2-4095, unit: dot).

[Return]

ZPLPrinter Instance

2.13. addGraphicCircle

The method is used to command produces a circle on the printed label.

ZPLPrinter addGraphicCircle(int x, int y, int diameter, int thickness)

[Parameter]

➤ x

Horizontal starting position

➤ y

Vertical starting position

➤ diameter

Round diameter(range:3-4095,unit:dot).

➤ thickness

Boundary thickness(range:1-4095,unit:dot).

[Return]

ZPLPrinter Instance

2.14. addBarcode

The method is used to prints 1D barcodes.

ZPLPrinter addBarcode(int x, int y, String codeType, String ratio, byte textPosition, String data, int width, int height)

ZPLPrinter addBarcode(int x, int y, String codeType, String data, int height)

ZPLPrinter addBarcode(int x, int y, String codeType, String data)

[Parameter]

➤ x

Start x value of the barcode

➤ y

Start y value of the barcode

➤ codeType

Code type

Variable	Description
BCS_CODE11	Code 11 barcode
BCS_INTERLEAVED2OF5	Interleaved 2 of 5 Bar Code
BCS_CODE39	Code 39 Barcode
BCS_EAN8	EAN-8 Barcode
BCS_UPCE	UPC-E Barcode
BCS_CODE93	Code 93 Barcode
BCS_CODE128	Code 128 Barcode
BCS_EAN13	EAN-13 Barcode
BCS_CODABAR	ANSI Codabar Bar Code
BCS_MSI	MSI Bar Code
BCS_PLESSEY	Plessey Bar Code
BCS_UPC_EAN	UPC/EAN Extensions
BCS_UPCA	UPC-A Bar Code

➤ ratio

Barcode direction, Default is ROTATION_0

➤ textPosition

Interpretation line position, Default is HRI_TEXT_BELOW.

Variable	Description
HRI_TEXT_NONE	No Interpretation
HRI_TEXT_ABOVE	print interpretation line above code
HRI_TEXT_BELOW	print interpretation line below code

➤ data

Barcode content

➤ width

module width (in dots), Default is 2.

➤ height

bar code height (in dots), Default is 50.

[Return]

ZPLPrinter Instance

2.15. addQRCode

Add 2D barcode

ZPLPrinter addQRCode(int x, int y, String data)

ZPLPrinter addQRCode(int x, int y, int size, String data)

[Parameter]

➤ x

Start x value of the qrcode

➤ y

Start y value of the qrcode

➤ data

QRCode content

➤ size

magnification factor. Values:1 to 10, default is 3.

[Return]

ZPLPrinter Instance

2.16. printBitmap

Print pictures

ZPLPrinter printBitmap(int x, int y, Bitmap bmp, int width, AlgorithmType algorithmType)

ZPLPrinter printBitmap(int x, int y, Bitmap bmp, int width)

By transmitting images through compression, you can save transmission time.

ZPLPrinter printBmpCompress(int x, int y, Bitmap bmp, int width, AlgorithmType algorithmType)

ZPLPrinter printBmpCompress(int x, int y, Bitmap bmp, int width)

Print base64 string to image file, only supports printers that can print compressed images.

ZPLPrinter printBase64Img(int x, int y, String base64String, int pageWidth)

ZPLPrinter printBase64Img(int x, int y, String base64String, int pageWidth, AlgorithmType algorithmType)

[Parameter]

➤ x

Horizontal starting position

➤ y

Vertical starting position

➤ bmp

Print width of picture

➤ base64String

The base64 string of the image file

➤ width

Print width of picture

➤ algorithmType

Algorithm type. Default is AlgorithmType.Threshold.

AlgorithmType.Dithering

AlgorithmType.Threshold

[Return]

ZPLPrinter Instance

2.17. printPdf

Print PDF files

ZPLPrinter printPdf(int x, int y, String path, int pageWidth, AlgorithmType algorithmType) throws IOException

ZPLPrinter printPdf(int x, int y, String path, int pageWidth) throws IOException

Print PDF files in base64 format

ZPLPrinter printBase64Pdf(int x, int y, String base64String, int pageWidth, AlgorithmType algorithmType) throws IOException

ZPLPrinter printBase64Pdf(int x, int y, String base64String, int pageWidth) throws IOException

[Parameter]

➤ x

Horizontal starting position

➤ y

Vertical starting position

➤ path

pdf document path

➤ base64String

The base64 string of the PDF file

➤ pageWidth

Print width

➤ algorithmType

Algorithm type. Default is AlgorithmType.Threshold.

AlgorithmType.Dithering

AlgorithmType.Threshold

[Return]

ZPLPrinter Instance

2.18. downloadBitmap

The method is used to download a graphic image.

ZPLPrinter downloadBitmap(int width, String bmpName, Bitmap bmp)

ZPLPrinter downloadBitmap(int width, String bmpName, Bitmap bmp, AlgorithmType algorithmType)

[Parameter]

➤ width

Print width of picture

➤ bmpName

image name and extension, The number or character whose name is 1 to 8.

➤ bmp

Bitmap object

➤ algorithmType

Algorithm type. Default is AlgorithmType.Threshold.

AlgorithmType.Dithering

AlgorithmType.Threshold

[Return]

ZPLPrinter Instance

2.19. addBitmap

The method is used to print bitmap

ZPLPrinter addBitmap(int x, int y, String bmpName, int mx, int my)

ZPLPrinter addBitmap(int x, int y, String bmpName)

[Parameter]

➤ x

Start x value of the bitmap

➤ y

Start y value of the bitmap

➤ bmpName

Bitmap name and extension name

➤ mx

magnification factor on the x-axis,The default value is 1, and the range is 1~10.

➤ my

magnification factor on the y-axis,The default value is 1, and the range is 1~10.

[Return]

ZPLPrinter Instance

2.20. addPrintCount

The method controls the number of labels to print

ZPLPrinter addPrintCount(int count)

[Parameter]

- count
total quantity of labels to print

[Return]

ZPLPrinter Instance

2.21. setPrintSpeed

This method is used to set the printing speed.

ZPLPrinter setPrintSpeed(int speed)

[Parameter]

- speed
print speed. Unit is inches/sec

[Return]

ZPLPrinter Instance

2.22. setPrintOrientation

The method inverts the label format 180 degrees. The label appears to be printed upside down.

ZPLPrinter setPrintOrientation(String orientation)

[Parameter]

- orientation

Print Orientation

Variable	Description
ROTATION_0	normal
ROTATION_180	invert

[Return]

ZPLPrinter Instance

2.23. setPrintDensity

The method is used to set the darkness of printing.

ZPLPrinter setPrintDensity(int density)

[Parameter]

➤ density

desired darkness(range: 0-30)

[Return]

ZPLPrinter Instance

2.24. setCloseHeadModel

Set to turn off print head mode

ZPLPrinter setCloseHeadModel(char model)

[Parameter]

➤ model

Print head off mode

Variable	Description
CLOSE_HEAD_FEED	Feed
CLOSE_HEAD_CALIBRATE	Calibration
CLOSE_HEAD_DETECTING	Detecting tag length
CLOSE_HEAD_NONE	No Motion
CLOSE_HEAD_C_FORM	Intelligent calibration
CLOSE_HEAD_BACK	Adjust label position
CLOSE_HEAD_SMART	Smart Feed

[Return]

ZPLPrinter Instance

2.25. rfidCalibration

This function is to initiate tag calibration for RFID media.

ZPLPrinter rfidCalibration()

[Return]

ZPLPrinter Instance

2.26. rfidWrite

This function is to Write RFID Format.

ZPLPrinter rfidWrite(char format, int begin, int size, char memoryBlock, String text)

[Parameter]

➤ format

format: A = ASCII, H = Hexadecimal, E = EPC (We need to first use rfidDefineEPC() to define the EPC data structure)

➤ begin

starting block number

➤ size

number of bytes to read or write

➤ memoryBlock

Specifies the Gen 2 memory bank.: 0 = Reserved, 1 = EPC, 2 = TID (Tag ID), 3 = User

➤ text

Text data

[Return]

ZPLPrinter Instance

2.27. rfidDefineFont

This function is to define the text format read by RFID for printing, and to be used in conjunction with rfidRead.

ZPLPrinter rfidDefineFont(int x, int y, char fontName, String rotation, int sizeW, int sizeH)

[Parameter]

➤ x

the starting x value of the text

➤ y

the starting y value of the text

➤ font

The font type of the text, the default is FNT_F.

Variable	Description
FNT_A	9 x 5
FNT_B	11x7
FNT_C、FNT_D	18x10
FNT_E	28x15
FNT_F	26x13
FNT_G	60x40
FNT_O	15*12

For other fonts, please use custom names.

➤ sizeW

The effective width of the text, the default is the base size. Please use an integer multiple of the base size.

➤ sizeH

The effective height of the text. Default is base size. Please use an integer multiple of the base size.

➤ rotation

Clockwise rotation angle, default ROTATION_0

Variable	Description
ROTATION_0	No rotation
ROTATION_90	Rotate 90 degrees clockwise
ROTATION_180	Rotate 180 degrees clockwise
ROTATION_270	Rotate 270 degrees clockwise

[Return]

ZPLPrinter Instance

2.28. rfidRead

This function is to Read RFID Format, Please use it in conjunction with the readData function.

ZPLPrinter rfidRead(char format, int begin, int size, char memoryBlock, boolean isPrint)

[Parameter]

➤ format

format: A = ASCII, H = Hexadecimal, E = EPC (We need to first use rfidDefineEPC() to define the EPC data structure)

➤ begin

starting block number

➤ size

number of bytes to read or write

➤ memoryBlock

Specifies the Gen 2 memory bank.:0 = Reserved, 1 =EPC, 2= TID (Tag ID), 3 = User

➤ isPrint

Is the content read printed out. true is used in conjunction with rfidDefineFont for printing, false is not printed.

[Return]

ZPLPrinter Instance

2.29. rfidSetPower

This function is to Set RF Power Levels for Read and Write

ZPLPrinter rfidSetPower(String read, String write)

[Parameter]

➤ read

read power, Values: 0 to 30

➤ write

write power, Values: 0 to 30

[Return]

ZPLPrinter Instance

2.30. rfidDefineEPC

This function is to Define EPC Data Structure

ZPLPrinter rfidDefineEPC(byte... bit)

[Parameter]

➤ bit

Partition size set. The maximum single partition is 64 bits.

[Return]

ZPLPrinter Instance

2.31. rfidSetParam

This function is to set up RFID parameters including tag type.

ZPLPrinter rfidSetParam(char labelType, int pos, int len, int number, char err)

[Parameter]

➤ labelType

Tag type.Values: 8 = EPC Class 1, Generation 2 (Gen 2)

➤ pos

read/write position of the tag (programming position)

➤ len

length of void printout

➤ number

number of labels to try encoding,The number of labels that will be attempted in case of read/encode failure.Values: 1 to 10

➤ err

[in] error handling,Values:

N = No action (printer drops the label format causing the error and moves to the next queued label)

P = Place printer in Pause mode (label format stays in the queue until the user cancels)

E = Place printer in Error mode (label format stays in the queue until the user cancels)

[Return]

ZPLPrinter Instance

2.32. printerStatus

Get printer status

void printerStatus(IStatusCallback callback)

void printerStatus(int timeout, IStatusCallback callback)

[Parameter]

➤ callback

The callback content is the corresponding printer state

```
public interface IStatusCallback {  
    void receive(int status);  
}
```

status(HEX)	Description
00	Normal
01	Head opened
02	Paper Jam
03	Paper Jam and head opened
04	Out of paper
05	Out of paper and head opened

08	Out of ribbon
09	Out of ribbon and head opened
0A	Out of ribbon and paper jam
0B	Out of ribbon, paper jam and head opened
0C	Out of ribbon and out of paper
0D	Out of ribbon, out of paper and head opened
10	Pause
20	Printing
80	Other error
-1	Receive timeout

➤ timeout

Receive timeout, Unit is ms, Default is 5000ms

[Return]

ZPLPrinter Instance

2.33. setCharSet

Set character encoding, Default is “UTF-8”

void setCharSet(String charSet)

[Parameter]

➤ charSet

Character set name.

2.34. sendData

The method is used to send data to the printer.

ZPLPrinter sendData(byte[] data);

ZPLPrinter sendData(List<byte[]> datas);

[Parameter]

➤ data

Byte array to be sent

➤ datas

Byte array collection to be sent

[Return]

ZPLPrinter Instance

2.35. waitSendResultSync

Check if the data has been sent successfully.

```
void waitSendResultSync(int timeout)
```

[Parameter]

➤ timeout

Maximum detection time, Unit in milliseconds

[Return]

void

3. ImageUtils

3.1. handleImageEffect

This method is used to adjust the contrast and brightness of the image.

```
static Bitmap handleImageEffect(Bitmap bmp, float contrast, float brightness)
```

[Parameter]

➤ bmp

Original image

➤ contrast

Contrast, The range is 0~2

➤ brightness

Brightness, The range is -255~255

[Return]

processed image object