

Gainscha Android Bluetooth/Ethernet/USB

Function Library Instructions

1. GTSPL_openPort()

★ Bluetooth

GTSPL_openPort(MacAddress)

■ Description: Start the printer spool.

■ Parameter:

➔ address: String Type, Specifies the Bluetooth MacAddress(BR/EDR MacAddress).

example: " DC:1D:30:00:1D:87"

★ Ethernet

GTSPL_openPort(IP, Port,time)

■ Description: Start the printer spool.

■ Parameter:

➔ IP: String Type, IP Address, example: "192.168.1.109"

➔ Port: int Type, Port Number, example: 8899

➔ time: int Type, Delay Time,1000 = 1 second, example: GTSPL_closePort(2000)

★ USB

GTSPL_openPort(manager, device)

■ Description: Start the printer spool.

■ Parameter:

➔ **manager:** USBManager Class, pass USBManager variables.

➔ **device:** USBDevice Class, pass USBDevice variables.

2. GTSPL_closePort()

- Description: Close printer spool.
- Parameter: None

3. GTSPL_closePort(time)

- Description: Close printer spool.
- Parameter:
 - ➔ time: int Type, delay time, 1000=1 second, example: GTSPL_closePort(2000)

4. GTSPL_setCmdSendMode(mode)

- Description: Sent the built-in commands to the printer or file.
- Parameter:
 - ➔ mode: String Type

F: Send the built-in commands to file.

(File will be saved in InternalStorage/android/data/packageName/files)

P: Send the built-in commands to the printer.

5. GTSPL_setup(width, height, speed, density, sensor, sensorDistance, sensorOffset, context)

- Description: Set up label width, label height, print speed, print density, sensor type, gap/black mark vertical distance and gap/black mark offset distance.
- Parameter:

Parameter	Type	Description
width	int	Set up label width; unit: mm.
height	int	Set up label height; unit: mm.
speed	int	Set up print speed (1~15: print speed at 1"~15"/sec). Selectable print speeds depend on different printer models, and maximum speed is 15"/sec.
density	int	Set up print density(0~15);If the number is bigger, the printout will be darker.

sensor	int	Set up the sensor type. 0: Gap sensor 1: Black mark sensor
sensorDistance	int	Set up vertical gap height of the gap/black mark; unit: mm
sensorOffset	int	Set up offset distance of the gap/black mark, unit: mm, this parameter is set to 0 when the general label is used.
context	Context	Pass current context.

6. GTSP_setup(width, height, speed, density, sensor, sensorDistance, sensorOffset, context)

- Description: Set up label width, label height, print speed, print density, sensor type, gap/black mark vertical distance and gap/black mark offset distance.

- Parameter:

Parameter	Type	Description
width	Double	Set up label width; unit: mm.
height	Double	Set up label height; unit: mm.
speed	int	Set up print speed (1~15: print speed at 1"~15"/sec). Selectable print speeds depend on different printer models, and maximum speed is 15"/sec.
density	int	Set up print density(0~15);If the number is bigger, the printout will be darker.
sensor	int	Set up the sensor type. 0: Gap sensor 1: Black mark sensor
sensorDistance	Double	Set up vertical gap height of the gap/black mark; unit: mm
sensorOffset	Double	Set up offset distance of the gap/black mark, unit: mm, this parameter is set to 0 when the general label is used.
context	Context	Pass current context.

7. GTSP_setDirectionAndMirror(direction,mirror, context)

- Description: Set direction and mirror.

- Parameter:

Parameter	Type	Description
direction	int	Set direction, default:0

		0 : Top out 1 : Bottom out
mirror	int	Set mirror 0 : No 1 : Yes
context	Context	Pass current context.

8. GTSPL_setShift (shiftY, context)

- Description: Set the vertical displacement distance, when value is positive, it will shift in the printing direction, otherwise, it will shift in opposite direction.
- Parameter:
 - ➔ shiftY : int type, vertical displacement distance, the unit is dot.
 - ➔ context : Context Class, Pass current context.

9. GTSPL_printReverse(x_start, y_start, x_width, y_height, context)

- Description: Reverse the designated area.
- Parameter:

Parameter	Type	Description
x_start	int	The x-coordinate of the area(in dots)
y_start	int	The y-coordinate of the area (in dots)
x_width	int	The width of area in dots
y_height	int	The height of area in dots
context	Context	Pass current context.

10.GTSPL_setOffset(offset, context)

- Description: Set feed offset(Usually use with peel mode and cut mode)
- Parameter:
 - ➔ **offset** : double type, extra feed offset, the unit is mm
 - ➔ context : Context Class, Pass current context.

11.GTSPL_setCutMode(mode, piece, context)

- Description: Set cut mode and cut number
- Parameter:

Parameter	Type	Description
mode	int	Set cut mode, default:1 0 : Backward 1 : Forward
piece	int	Set cut number
context	Context	Pass current context.

12.GTSPL_setAfterPrintAction(mode, context)

- Description: Set action after print
- Parameter:

Parameter	Type	Description
mode	int	Set action after print, default:1 0 : Normal 1 : Tear Mode 2 : Peel Mode 3 : Cut Mode
context	Context	Pass current context.

13.GTSPL_genericDefault (context)

- Description: Initialize the general setting value
- Parameter:
 - ➔ context : Context Class, Pass current context.

14.GTSPL_sensorDefault (context)

- Description: Initialize the sensor setting value
- Parameter:
 - ➔ context : Context Class, Pass current context.

15. GTSPL_clearBuffer(context)

- Description: Clear the image buffer.
- Parameter:
 - ➔ context: Context Class, Pass current context.

16. GTSPL_barcode(x, y, type, height, readable, rotation, narrow, wide, content, context)

- Description: Use built-in barcode formats to print barcodes.
- Parameter:

Parameter	Type	Description
x	int	Specify the x-coordinate bar code on the label, Unit: dot
y	int	Specify the y-coordinate bar code on the label, Unit: dot
type	String	Set up Code Type · refer to Appendix
height	int	Set up bar code height (in dots)
readable	int	Set up whether to print human recognizable interpretation (text) or not. 0: Do not print 1: Print barcode document left 2: Print barcode code document in the center 3: Print barcode document right
rotation	int	Set up barcode rotation 0 : No rotation

		90 : Rotate 90 degrees clockwise 180 : Rotate 180 degrees clockwise 270 : Rotate 270 degrees clockwise
narrow	int	Set up narrow bar ratio (in dots), refer to Appendix
wide	int	Set up wide bar ratio (in dots), refer to Appendix
content	String	Content of barcode. Please note that the maximum number of digits of bar code content. refer to Appendix
context	Context	Pass current context.

17. GTSPL_formFeed(context)

- Description: Feed label to the top of next label.
- Parameter:
 - ➔ context: Context Class, Pass current context.

18. GTSPL_noBackFeed(context)

- Description: Set the paper not to back feed.
- Parameter:
 - ➔ context: Context Class, Pass current context.

19. GTSPL_sendCommand (context, command)

- Description: Sends built-in commands to the printer.
- Parameter:
 - ➔ command: String Type, refer to TSPL programming manual commands for details.
 - ➔ context: Context Class, Pass current context.

20. GTSPL_printerFont(x, y, size, rotation, x_scale, y_scale, content, context)

- Description: Use printer built-in fonts to print.
- Parameter:

Parameter	Type	Description
x	int	The x-coordinate of the text
y	int	The y-coordinate of the text
size	String	Built-in font type 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots TST24.BF2: Traditional Chinese 24*24 TST16.BF2: Traditional Chinese 16*16 TSS24.BF2: Simplified Chinese 24*24 TSS16.BF2: Simplified Chinese 16*16
rotation	int	The rotation angle of text 0 : No rotation 90: degrees, in clockwise direction 180 : degrees, in clockwise direction 270 : degrees, in clockwise direction

x_scale	int	Horizontal multiplication, Available factors: 1~10
y_scale	int	Vertical multiplication, Available factors: 1~10
content	String	Content of text string
context	Context	Pass current context.

21.GTSPL_qrcode(x, y, size, ECCLevel, cellWidth, mode, rotation, content)

- Description : Use built-in QR code formats to print QR code
- Parameter:

parameter	Type	illustrate
x	int	The upper left corner x-coordinate of the QR code
y	int	The upper left corner y-coordinate of the QR code
ECCLevel	String	Error correction recovery level L : 7% 、 M : 15% 、 Q : 25% 、 H : 30%
cellWidth	int	QR code shape width 1~10
mode	int	Set QR code mode : A : Auto 、 M : Manual
rotation	int	Set up QR code rotation : 0 : 0 degree 、 90 : 90 degree 、 180 : 180 degree 、 270 : 270 degree
content	String	The encodable character set is described as below, *Encodable character set: 1) Numeric data: (digits 0~9) 2) Alphanumeric data Digits 0-9

		<p>Upper case letters A-Z</p> <p>Nine other characters: space, \$ % * + - . / :)</p> <p>3) 8-bit byte data</p> <p>JIS 8-bit character set (Latin and Kana) in accordance with JIS X 0201</p> <p>4) Kanji characters</p> <p>Shift JIS values 8140HEX –9FFCHEX and E040HEX –EAA4 HEX. These are values shifted from those of JIS X 0208. Refer to JIS X 0208 Annex 1</p> <p>Shift Coded Representation for detail.</p>
context	Context	Pass current context.

22. GTSPL_printLabel(set, copy, context)

- Description: Print the label format currently stored in the image buffer.
- Parameter:
 - ➔ set: int Type, Specifies how many sets of labels will be printed.
 $1 \leq \text{set} \leq 999999999$
 - ➔ copy: int Type, Specifies how many copies should be printed for each particular label set.
 $1 \leq \text{copy} \leq 999999999$
 - ➔ context: Context Class, Pass current context.

23. GTSPL_downloadPCX(filename, context)

- Description: Download the monochrome PCX file to main board flash memory

- Parameter:

- ➔ filename: String Type, PCX file name

- (File has to be saved in InternalStorage/android/data/packageName/files)

- ➔ context: Context Class, Pass current context.

24. GTSPL_downloadBMP(filename, context)

- Description: Download the monochrome BMP file to main board flash memory

- Parameter:

- ➔ filename: String Type, BMP file name.

- (File has to be saved in InternalStorage/android/data/packageName/files)

- ➔ context: Context Class, Pass current context.

25. GTSPL_download_Not1BitDepthBMP (filename, context)

- Description: Download the non-monochrome file to main board flash memory.

- Parameter:

- ➔ filename: String Type, File name

- (File has to be saved in InternalStorage/android/data/packageName/files)

- ➔ context : Context Class, Pass current context.

26. GTSPL_downloadTTF(filename, context)

- Description: Download the True Type Font file to main board flash memory.

- Parameter:

- ➔ filename: String Type, File name

- (File has to be saved in InternalStorage/android/data/packageName/files)

- ➔ context: Context Class, Pass current context.

27. GTSPL_printerStatus(delaytime)

- Description: Obatin the printer status.

■ Parameter:

➔ delaytime: int Type, Set up delay time.

■ Return Value(String Type):

Return Value	Printer Status
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

28. GTSPL_getSDKVersion (returnWay, context)

■ Description: Response the SDK version.

■ Parameter:

➔ retrunWay: int Type, pass entering 0 will bounce out the SDK version message in addition to returning the SDK.

➔ context: Context Class, Pass current context.

29. GTSPL_writeUHF (dataFormat,startBlockNo,byteSize,Gen2MemoryBank,datastring, context)

■ Description: Write data to UHF tag memory.

■ Parameter:

Parameter	Type	Description
dataFormat	string	Define data format · default is "H" A : ASCII H : Hexadecimal
startBlockNo	int	Secify the 16-bit starting block number · Default : 2
byteSize	int	Values: 1 to n, where n is the maximum number of bytes for the tag. Default: 1
Gen2Memory Bank	string	Select Gen2 memory bank R : Reserved E : EPC (Default) T : TID(Tag ID) U : User
datastring	string	Data string.
context	Context	Pass current context.

30. GTSPL_EPCPWD_Action(action, password, context)

- Description: Lock or unlock EPC memory with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
action	string	Action type U : unlock EPC memory bank L : lock EPC memory bank

		O : permanently unlock EPC memory bank P : permanently lock EPC memory bank
password	string	password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

31. GTSPL_TIDPWD_Action(action, password, context)

- Description: Lock or unlock TID memory with password for UHF GEN2 tag.

- Parameter:

Parameter	Type	Description
action	string	Action type U : unlock TID memory bank L : lock TID memory bank O : permanently unlock TID memory bank P : permanently lock TID memory bank
password	string	password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

32. GTSPL_USERPWD_Action(action, password, context)

- Description: Lock or unlock USER memory with password for UHF GEN2 tag.

- Parameter:

Parameter	Type	Description
action	String	Action type

		U : unlock USER memory bank L : lock USER memory bank O : permanently unlock USER memory bank P : permanently lock USER memory bank
password	String	password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

33. GTSPL_AccessPWD_Action (action, password, context)

- Description: Lock or unlock access password with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
action	string	Action type U : unlock the access password* L : lock the access password* O : permanently unlock the access password P : permanently lock the access password S : Set Password
password	string	password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

34. GTSPL_KillPWD_Action (action, password, context)

- Description: Lock or unlock kill password with password for UHF GEN2 tag.

- Parameter:

Parameter	Type	Description
action	string	Action type U : unlock the kill password* L : lock the kill password* O : permanently unlock the kill password P : permanently lock the kill password S : Set Password
password	string	password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

35. GTSPL_Set_RFIDPorcedure (tagType, rw_position, void_printout, tryEncodie_times,error_handle, speed, retry_times, context)

- Description: Set RFID procedure

- Parameter:

Parameter	Type	Description
tagType	int	Set Tag type · accepted value:1~10 · For UHF: 1 = ISO 18000 6C/Class 1 Gen2 (Q command) 8 = ISO 18000 6C/Class 1 Gen 2 (default)

		<p>For HF</p> <p>10 = UHF-J</p>
rw_position	int	<p>Move the media to the specified position on the label, measured in dot rows from the label top, before encoding</p> <p>Accept value: 0~9999(mm) · default is 0</p>
void_printout	int	<p>Set the length of the void printout in vertical (Y axis) dot rows.</p> <p>Accepted values: 0 to label length</p> <p>Default: label length</p>
tryEncodie_times	string	<p>The number of labels that will be attempted in case of read/encode failure. Accepted values: 1 to 10</p> <p>Default: 3</p>
error_handle	string	<p>If an error persists after the specified number of labels are tried, perform this error handling action.</p> <p>N : No action (Default)</p> <p>P : Pause mode</p> <p>E : Error mode</p>
speed	int	<p>If a label is voided, the speed at which "VOID" will be printed across the label.</p> <p>Accepted value: 2~10(IPS),</p>

		Default is 2.
retry_times	int	The retry times of a tag that will be attempted in case of read/encode failure. Accepted value:0~10 · Default is 6
context	Context	Pass current context.

36. GTSPL_Set_RFIDPorcedure (tagType, rw_position, void_printout, tryEncodie_times,error_handle, speed, retry_times, dpi, context)

- Description: Set RFID procedure
- Parameter:

Parameter	Type	Description
tagType	int	Set Tag type · accepted value:1~10 · For UHF: 1 = ISO 18000 6C/Class 1 Gen2 (Q command) 8 = ISO 18000 6C/Class 1 Gen 2 (default) For HF 10 = UHF-J
rw_position	int	Move the media to the specified position on the label, measured in dot rows from the label top, before encoding Accept value: 203dpi:0 ~ 1251 (mm) 、 300dpi:0 ~ 846 (mm) 、 600dpi:0 ~ 423 (mm) · default is 0

void_printout	int	<p>Set the length of the void printout in vertical (Y axis) dot rows.</p> <p>Accepted values: 0 to label length</p> <p>Default: label length</p>
tryEncode_times	string	<p>The number of labels that will be attempted in case of read/encode failure. Accepted values: 1 to 10</p> <p>Default: 3</p>
error_handle	string	<p>If an error persists after the specified number of labels are tried, perform this error handling action.</p> <p>N : No action (Default)</p> <p>P : Pause mode</p> <p>E : Error mode</p>
speed	int	<p>If a label is voided, the speed at which "VOID" will be printed across the label.</p> <p>Accepted value: 2~10(IPS),</p> <p>Default is 2.</p>
retry_times	int	<p>The retry times of a tag that will be attempted in case of read/encode failure.</p> <p>Accepted value:0~10 · Default is 6</p>
dpi	String	<p>Set DPI of the printer</p> <p>203: 203 dpi</p>

		300: 300 dpi 600: 600 dpi
context	Context	Pass current context.

37. GTSPL_writeHF (dataFormat,startBlockNo,byteSize,datastring, context)

- Description: Write data to HF tag memory.

- Parameter:

Parameter	Type	Description
dataFormat	string	Define data format · default is "H" A : ASCII H : Hexadecimal
startBlockNo	int	Secify the 16-bit starting block number · Default : 2
byteSize	int	Values: 1 to n, where n is the maximum number of bytes for the tag. Default: 1
datastring	string	Data string
context	Context	Pass current context.

38. GTSPL_printerFontBlock (x, y, width, height, fontname, rotation, x_scale, y_scale, space, align,

content, context

- Description: Use printer built-in fonts to print paragraph.

- Parameter:

Parameter	Type	Description
x	string	The x-coordinate of the text (in dots)
y	string	The y-coordinate of the text (in dots)

width	string	The width of block for the paragraph in dots
height	string	The height of block for the paragraph in dots
fontname	string	<p>Built-in font type</p> <p>1: 8*/12 dots</p> <p>2: 12*20 dots</p> <p>3: 16*24 dots</p> <p>4: 24*32 dots</p> <p>5: 32*48 dots</p> <p>TST24.BF2: Traditional Chinese 24*24</p> <p>TST16.BF2: Traditional Chinese 16*16</p> <p>TSS24.BF2: Simplified Chinese 24*24</p> <p>TSS16.BF2: Simplified Chinese 16*16</p>
rotation	string	<p>The rotation angle of text</p> <p>0 : No rotation</p> <p>90 : degrees, in clockwise direction</p> <p>180 : degrees, in clockwise direction</p> <p>270 : degrees, in clockwise direction</p>
x_scale	string	Horizontal multiplication, Available factors: 1~10
y_scale	string	Vertical multiplication, Available factors: 1~10
space	string	Add or delete the space between lines (in dots)

align	string	Text alignment 0 : default (Left) 1 : Left 2 : Center 3 : Right
content	string	Data in block. The maximum data length is 4092 bytes.
context	Context	Pass current context.

39. **GTSP_readUHF(dataFormat,startBlockNo,byteSize,Gen2MemoryBank,context)**

- Description: Read data from UHF tag memory (R command)
- Parameter:

Parameter	Type	Description
dataFormat	string	Setting callback returned data format · A : ASCII H : Hexadecimal (Default)
startBlockNo	int	Secify the 16-bit starting block number to read · Default is 0
byteSize	int	Secify the data lengths to read · default is 1
Gen2Memory Bank	string	Gen2 memory bank · R = Reserved E = EPC T = TID

		U = UESR Default : E
context	Context	Pass current context.

■ Return Value(String Type):

dataFormat	Return string(example)
A	Label data is displayed in ASCII (ex: 24051324000103456400)
H	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)
* An error occurs and an error code is returned , For error code description, refer to Appendix 2	

40. GTSPL_readUHF(dataFormat,startBlockNo,byteSize,Gen2MemoryBank, delaytime,context)

■ Description: Read data from UHF tag memory (R command)(only supports USB)

■ Parameter:

Parameter	Type	Description
dataFormat	string	Setting callback returned data format . A : ASCII H : Hexadecimal (Default)
startBlockNo	int	Secify the 16-bit starting block number to read . Default is 0
byteSize	int	Secify the data lengths to read . default is 1
Gen2Memory Bank	string	Gen2 memory bank . R = Reserved E = EPC

		T = TID U = UESR Default : E
delaytime	int	Setting read delaytime
context	Context	Pass current context.

■ Return Value(String Type):

dataFormat	Return string(example)
A	Label data is displayed in ASCII (ex: 24051324000103456400)
H	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)
* An error occurs and an error code is returned , For error code description, refer to Appendix 2	

41. GTSPL_readUHFQ(dataFormat, PCReturnStatus, CRCReturnStatus,context)

■ Description: Read data from UHF tag memory (Q command)

■ Parameter:

Parameter	Type	Description
dataFormat	string	Set up callback return data format . A : ASCII H : Hexadecimal (Default)
PCReturnStatus	int	enable/disable PC value returned 0 : read epc data not include PC value 1 : read epc data include PC value
CRCReturnStatus	int	enable/disable CRC-16 value returned 0 : read epc data not include CRC-16 value

		1 : read epc data include CRC-16 value
context	Context	Pass current context.

- Return Value(String Type):

dataFormat	Return string(example)
A	Label data is displayed in ASCII (ex: 24051324000103456400)
H	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)
* An error occurs and an error code is returned · For error code description, refer to Appendix 2	

42. GTSPL_LabelCalibration (Context context)

- Description: Auto calibration for RFID label
- Parameter:
 - context: Context Class, Pass current context.

43. GTSPL_rfidSetupDefault (Context context)

- Description: Initialize the RFID setting value
- Parameter:
 - context: Context Class, Pass current context.

44. GTSPL_writeGJB(String dataFormat, int startBlockNo, int byteSize, String GJBMemoryBank, String datastring, String writePWD, Context context)

- Description: Write data to UHF GJB tag memory
- Parameter:

Parameter	Type	Description
dataFormat	string	Define data format · default is "H" A : ASCII

		H : Hexadecimal
startBlockNo	int	Secify the 16-bit starting block number · Default : 1
byteSize	int	Values: 1 to n, where n is the maximum number of bytes for the tag. Default: 1
Gen2MemoryBank	string	Select GJB memory bank to write R = Reserved E = EPC T = TID U = User 2 = User 2 3 = User 3 Default : E
datastring	string	Data string
writePWD	string	Writing password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

45. GTSPL_readGJB(String dataFormat, int startBlockNo, int byteSize, String GJBMemoryBank, String ReadPWD, Context context)

- Description: Read data from UHF GJB tag.
- Parameter:

Parameter	Type	Description
dataFormat	string	Set up callback returned data format · A : ASCII H : Hexadecimal (Default)
startBlockNo	int	Secify the 16-bit starting block number to read · Default is 0
byteSize	int	Secify the data lengths to read · default is 1
Gen2MemoryBank	string	Select GJB memory bank to read · E = EPC T = TID U = User 2 = User 2 3 = User 3 Default : E
ReadPWD	string	Reading password · 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

■ Return Value(String Type):

dataFormat	Return string(example)
A	Label data is displayed in ASCII (ex: 24051324000103456400)
H	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)

46. GTSPL_Set_GJB_Pwd(String pwdArea, String action, String pwdSet, String writePWD, Context context)

- Description: Set Write/Read/Status/Kill Password to UHF GJB tag.
- Parameter:

Parameter	Type	Description
passwordArea	string	Set password area K=Kill, W=Write (Default), R=Read, S=Status
action	string	S=Set Password
pwdSet	string	New password for password area setting above . 8 HEX characters. (0~9, A,B,C,D,E,F)
writePWD	string	Writing password . 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

47. GTSPL_Set_GJB_Status(String GJBMemoryBank, String action, String statusPWD, Context context)

- Description: Set memory status with password for UHF GJB tag.
- Parameter:

Parameter	Type	Description
Gen2MemoryBank	string	Select GJB memory bank for set, E = EPC (default) T = TID

		U = User 2 = User 2 3 = User 3
action	string	Memory Status type A : Lock0(readable and writable) B : Lock1(read only) C : Lock2(write only) D : Lock3(non-readable and non-writable) *Each Memory support status : <ul style="list-style-type: none"> EPC area : A:read and write B: read only USER area(include User 、User2 、User3) : A:read and write B:read only C: write only D: not allow access TID area : B:read only D: not allow access SAFE area : C: write only D: not allow access
statusPWD	string	Status password , 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

48. GTSPL_Kill_GJB_Tag(String kill_PWD, Context context)

■ Description: Kill UHF GJB tag.

■ Parameter:

Parameter	Type	Description
kill_PWD	string	Killing password , 8 HEX characters. (0~9, A,B,C,D,E,F)
context	Context	Pass current context.

49. GTSPL_WifiFrequency (String Frequency, Context context)

- Description: When using a compatible 5G frequency band WiFi module, it can be used to switch between frequency bands.
- Parameter:

Parameter	Type	Description
Frequency	string	Module Frequency Setting: 2.4G: Use the 2.4GHz frequency band. 5G: Use the 5GHz frequency band. BOTH: Use dual-band frequency
context	Context	Pass current context.

50. GTSPL_printBMP (String x, String y, int width, int height, int mode, String filename, Context context)

- Description: Convert the image into a monochrome bitmap and directly print using the printer.
- Parameter:

Parameter	Type	Description
x	string	The x-coordinate of the text (in dots)
y	string	The y-coordinate of the text (in dots)
width	int	The width of the image is represented in bytes.
height	int	The height of the image is represented in dots (pixels).
mode	int	Image format 0: OVERWRITE 1: OR 2: XOR
filename	string	The file name . (File has to be saved in InternalStorage/android/data/packageName/files) The supported image formats are as follows: 1. BMP (Bitmap): A bitmap format. 2. JPG (JPEG): Compressed image format. 3. PNG (Portable Network Graphics): Lossless compressed image format. 4. GIF (Graphics Interchange Format): Format that supports multiple images, commonly used for animations. 5. TIFF (Tagged Image File Format): High-quality lossless compressed image format.

		6. ICO (Icon): Icon format used for displaying icons of files, programs, or folders. 7. WMF (Windows Metafile): Windows graphics file format. EMF (Enhanced Metafile): Enhanced Windows graphics file format.
context	Context	Pass current context.

51. GTSPL_printBMP_Compression (String x, String y, int width, int height, String filename, Context context)

- Description: Convert the image into a monochrome bitmap, compress it, and then print using the printer.

- Parameter:

Parameter	Type	Description
x	string	The x-coordinate of the text (in dots)
y	string	The y-coordinate of the text (in dots)
width	int	The width of the image is represented in bytes.
height	int	The height of the image is represented in dots (pixels).
filename	string	The file name. (File has to be saved in InternalStorage/android/data/packageName/files) The supported image formats are as follows: 1. BMP (Bitmap): A bitmap format. 2. JPG (JPEG): Compressed image format. 3. PNG (Portable Network Graphics): Lossless compressed image format. 4. GIF (Graphics Interchange Format): Format that supports multiple images, commonly used for animations. 5. TIFF (Tagged Image File Format): High-quality lossless compressed image format. 6. ICO (Icon): Icon format used for displaying icons of files, programs, or folders. 7. WMF (Windows Metafile): Windows graphics file format. EMF (Enhanced Metafile): Enhanced Windows graphics file format.
context	Context	Pass current context.

Android Bluetooth Examples

1.Add the following permissions in AndroidManifest.xml first:

```
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

<uses-permission android:name="android.permission.BLUETOOTH" />
```

2. Import GTSPS_SDK SDK:

```
import com.gainscha.gtspl_sdk.GTSPLActivity;
```

3.Example:

```
public class MainActivity extends AppCompatActivity {

    GTSPLActivity mGtsplCmdTest = new GTSPLActivity();

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        mGtsplCmdTest.GTSPL_setCmdSendMode ("P");

        mGtsplCmdTest.GTSPL_openPort ("DC:1D:30:00:1D:87");

        mGtsplCmdTest.GTSPL_setup(62, 45, 2, 6, 0, 3, 0, this);

        mGtsplCmdTest.GTSPL_setup(62.0, 45.0, 2, 6, 0, 3.0, 0.0, this);

        mGtsplCmdTest.GTSPL_sendCommand(this, "DIRECTION 1\n\n");

        mGtsplCmdTest.GTSPL_clearBuffer(this);

        mGtsplCmdTest.GTSPL_printerFont(100, 100, "3", 0, 1, 1, "Print Font 123456", this);

        mGtsplCmdTest.GTSPL_barcode(30, 30, "128", 100, 1, 0, 2, 2, "barcode1234567", this);

        mGtsplCmdTest.GTSPL_qrcode(300, 100, "H", 4, "A", 0, "ABCabc123", this);

        mGtsplCmdTest.GTSPL_downloadBMP("CIRCLE.BMP", this);

        mGtsplCmdTest.GTSPL_sendCommand(this, "PUTBMP 150, 30, \"CIRCLE.BMP\"\n\n");

        mGtsplCmdTest.GTSPL_download_Not1BitDepthBMP("printTest4.BMP", MainActivity.this);

        mGtsplCmdTest.GTSPL_sendCommand(MainActivity.this,

            "PUTBMP10,10,\"PrintTest4.BMP\"\n\n");
```

```
String sBlock = "We stand behind our products with one of the most
```



```

comprehensive support programs in the Auto-ID industry.";

mGtsplCmdTest.GTSPL_printFontBlock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2",sBlock,this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

// initialization

mGtsplCmdTest.GTSPL_genericDefault(); // Initialize the general setting value
mGtsplCmdTest.GTSPL_sensorDefault(); // Initialize the sensor setting value
mGtsplCmdTest.GTSPL_rfidSetupDefault(); // Initialize the RFID setting value

// Modify Wifi frequency band

mGtsplCmdTest.GTSPL_WifiFrequency("5G");

//Set print function(It needs to use with print)

mGtsplCmdTest.GTSPL_setDirectionAndMirror(1, 1); // Set direction and mirror.

mGtsplCmdTest.GTSPL_setShift(50); // Set the vertical displacement distance

mGtsplCmdTest.GTSPL_printReverse(10, 10, 160, 160); //Set reverse

mGtsplCmdTest.GTSPL_setOffset(20); // Set feed offset

mGtsplCmdTest.GTSPL_setCutMode(0, 2); // Set cut mode and cut number

mGtsplCmdTest.GTSPL_setAfterPrintAction(2); // Set action after print

mGtsplCmdTest.GTSPL_printLabel(1, 1);

//Bitmap Print

mGtsplCmdTest.GTSPL_printBMP(-500,30,400,300,1,"CIRCLE.BMP ",this);

mGtsplCmdTest.GTSPL_printBMP_Compression(-500,30,400,300," CIRCLE.BMP ",this);

mGtsplCmdTest.GTSPL_printLabel(1, 1);

//GEN2 RFID

mGtsplCmdTest.GTSPL_writeUHF("H", 2, 12, "E", "414142424343444445454646", this);

mGtsplCmdTest.GTSPL_GTSPL_EPCPWD_Action("U", "12345678", this);

mGtsplCmdTest.GTSPL_GTSPL_TIDPWD_Action("L", "12345678", this);

mGtsplCmdTest.GTSPL_USERPWD_Action("L", "12345678", this);

```

```

mGtsplCmdTest.GTSPL_AccessPWD_Action("S", "12345678", this);

mGtsplCmdTest.GTSPL_KillPWD_Action("S", "12345678", this);

mGtsplCmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, this);

mGtsplCmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, "203", this);

mGtsplCmdTest.GTSPL_writeHF("H", 2, 12, "414142424343444445454646", this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

String uhfData = mGtsplCmdTest.GTSPL_readUHF("H",2,12,"E", this);

String uhfData = mGtsplCmdTest.GTSPL_readUHFQ("H",0,0, this);

//GJB RFID set password

// Use write password, set a new read password

mGtsplCmdTest.GTSPL_Set_GJB_Pwd("R","S","87654321","12345678",this);

// Use write password, set a new write password

mGtsplCmdTest.GTSPL_Set_GJB_Pwd("W","S","87654321","12345678",this);

// Use write password, set a new kill password

mGtsplCmdTest.GTSPL_Set_GJB_Pwd("K","S","87654321","12345678",this);

// Use write password, set a new status password

mGtsplCmdTest.GTSPL_Set_GJB_Pwd("S","S","87654321","12345678",this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID set status

mGtsplCmdTest.GTSPL_Set_GJB_Status("E","C","11112222",this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID write EPC data

mGtsplCmdTest.GTSPL_writeGJB("H",2,12,"E","404041414242434344444545","12345678",t
his);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID read EPC data

String GJBData=mGtsplCmdTest.GTSPL_readGJB("H",2,12,"E","33334444",MainActivity.this);

```

```

//GJB RFID kill tag

mGtsplCmdTest.GTSPL_Kill_GJB_Tag("11224455",this);

mGtsplCmdTest.GTSPL_printLabel(1,1,this);

// Print Simplified Chinese

String stString="默认简体中文测试";

mGtsplCmdTest.GTSPL_clearBuffer(this);

mGtsplCmdTest.GTSPL_printerFont(100, 10, "TSS24.BF2", 0, 1, 1, stString, this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

// Print traditional Chinese

String ttString="默認繁體中文測試";

mGtsplCmdTest.GTSPL_clearBuffer(this);

mGtsplCmdTest.GTSPL_printerFont(100, 10, "TST24.BF2", 0, 1, 1, ttString, this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

String status = mGtsplCmdTest.GTSPL_printersStatus(1000);

mGtsplCmdTest.GTSPL_closePort(1000);

String version= mGtsplCmdTest.GTSPL_getSDKVersion(0,this);

//RFID auto calibration

mGtsplCmdTest.GTSPL_LabelCalibration(this);

}

}

```

Android Ethernet Examples

1.Add the following permissions in AndroidManifest.xml first:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

2.Import GTSPL_SDK SDK:

```
import com.gainscha.gtspl_sdk.GTSPLWiFiActivity;
```

3.Example:

```

public class MainActivity extends AppCompatActivity {

    GTSPWIFIActivity mGtsplWIFICmdTest = new GTSPWIFIActivity();

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        mGtsplWIFICmdTest.GTSPL_setCmdSendMode("P");

        mGtsplWIFICmdTest.GTSPL_openPort("192.168.1.109", 8899);

        mGtsplWIFICmdTest.GTSPL_setup(62, 45, 2, 6, 0, 3, 0, this);

        mGtsplWIFICmdTest.GTSPL_setup(62.0, 45.0, 2, 6, 0, 3.0, 0.0, this);

        mGtsplWIFICmdTest.GTSPL_sendCommand(this, "DIRECTION 1\n\n");

        mGtsplWIFICmdTest.GTSPL_clearBuffer(this);

        mGtsplWIFICmdTest.GTSPL_printerFont(100,10,"5",0,1,1,"Print Font 123456",this);

        mGtsplWIFICmdTest.GTSPL_barcode(30, 30, "128", 100, 1, 0, 2, 2, "barcode1234567", this);

        mGtsplWIFICmdTest.GTSPL_qrcode(300, 100, "H", 4, "A", 0, "ABCabc123", this);

        mGtsplWIFICmdTest.GTSPL_downloadBMP("CIRCLE.BMP", this);

        mGtsplWIFICmdTest.GTSPL_sendCommand(this, "PUTBMP 150,30,\"CIRCLE.BMP\"\n\n");

        mGtsplWIFICmdTest.GTSPL_download_Not1BitDepthBMP("printTest4.BMP",
MainActivity.this);

        mGtsplWIFICmdTest.GTSPL_sendCommand(MainActivity.this, "PUTBMP
10,10,\"PrintTest4.BMP\"\n\n");

        String sBlock = "We stand behind our products with one of the most comprehensive support
programs in the Auto-ID industry.";

        mGtsplWIFICmdTest.GTSPL_printFontBlock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2",sBlock,this);

        mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

        // initialization

        mGtsplWIFICmdTest.GTSPL_genericDefault(); // Initialize the general setting value

```

```

mGtsplWIFICmdTest.GTSPL_sensorDefault(); // Initialize the sensor setting value

mGtsplWIFICmdTest.GTSPL_rfidSetupDefault(); // Initialize the RFID setting value

//Modify Wifi frequency band

mGtsplWIFICmdTest.GTSPL_WifiFrequency("5G");

//Set print function(It needs to use with print)

mGtsplWIFICmdTest.GTSPL_setDirectionAndMirror(1, 1); // Set direction and mirror

mGtsplWIFICmdTest.GTSPL_setShift(50); // Set the vertical displacement distance

mGtsplWIFICmdTest.GTSPL_printReverse(10, 10, 160, 160); //Set reverse

mGtsplWIFICmdTest.GTSPL_setOffset(20); // Set feed offset

mGtsplWIFICmdTest.GTSPL_setCutMode(0, 2); // Set cut mode and cut number

mGtsplWIFICmdTest.GTSPL_setAfterPrintAction(2); // Set action after print

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1);

//Bitmap Print

mGtsplWIFICmdTest.GTSPL_printBMP(-500,30,400,300,1,"CIRCLE.BMP ",this);

mGtsplWIFICmdTest.GTSPL_printBMP_Compression(-500,30,400,300," CIRCLE.BMP ",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1);

//GEN2 RFID

mGtsplWIFICmdTest.GTSPL_writeUHF("H", 2, 12, "E", "414142424343444445454646", this);

mGtsplWIFICmdTest.GTSPL_EPCPWD_Action("L", "12345678", this);

mGtsplWIFICmdTest.GTSPL_TIDPWD_Action("L", "12345678", this);

mGtsplWIFICmdTest.GTSPL_USERPWD_Action("L", "12345678", this);

mGtsplWIFICmdTest.GTSPL_AccessPWD_Action("S", "12345678", this);

mGtsplWIFICmdTest.GTSPL_KillPWD_Action("S", "12345678", this);

mGtsplWIFICmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, this);

mGtsplWIFICmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, "203", this);

mGtsplWIFICmdTest.GTSPL_writeHF("H", 2, 12, "414142424343444445454646", this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

```

```

String uhfData = mGtsplWIFICmdTest.GTSPL_readUHF("H",2,12,"E", this);

String uhfData = mGtsplWIFICmdTest.GTSPL_readUHFQ("H",0,0,this);

//GJB RFID set password

// Use write password, set a new read password

mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("R","S","87654321","12345678",this);

// Use write password, set a new write password

mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("W","S","87654321","12345678",this);

// Use write password, set a new kill password

mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("K","S","87654321","12345678",this);

// Use write password, set a new status password

mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("S","S","87654321","12345678",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID set status

mGtsplWIFICmdTest.GTSPL_Set_GJB_Status("E","C","11112222",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID write EPC data

mGtsplWIFICmdTest.GTSPL_writeGJB("H",2,12,"E","404041414242434344444545","123456
78",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID read EPC data

String GJBData =

mGtsplWIFICmdTest.GTSPL_readGJB("H",2,12,"E","33334444",MainActivity.this);

//GJB RFID kill tag

mGtsplWIFICmdTest.GTSPL_Kill_GJB_Tag("11224455",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1,1,this);

// Print Simplified Chinese

String stString="默认简体中文测试";

```

```

        mGtsplWIFICmdTest.GTSPL_clearBuffer(this);

        mGtsplWIFICmdTest.GTSPL_printerFont(100, 10, "TSS24.BF2", 0, 1, 1, stString, this);

        mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

        // Print traditional Chinese
        String ttString="默認繁體中文測試";

        mGtsplWIFICmdTest.GTSPL_clearBuffer(this);

        mGtsplWIFICmdTest.GTSPL_printerFont(100, 10, " TST24.BF2", 0, 1, 1, ttString, this);

        mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);

        String status = mGtsplWIFICmdTest.GTSPL_printersStatus(1000);

        mGtsplWIFICmdTest.GTSPL_closePort();

        String version=mGtsplWIFICmdTest.GTSPL_getSDKVersion(0,this);

        //RFID auto calibration

        mGtsplWIFICmdTest.GTSPL_LabelCalibration(this);

    }

}

```

Android USB Examples

1.Import GTSPL_SDK:

```
import com.gainscha.gtspl_sdk.GTSPLUsbActivity;
```

2.Example:

```

public class MainActivity extends AppCompatActivity {

    GTSPLUsbActivity mUSB = new GTSPLUsbActivity();

    private static final String ACTION_USB_PERMISSION ="com.android.example.USB_PERMISSION";

    private static UsbManager mUsbManager;

    private static PendingIntent mPermissionIntent;

    private static boolean hasPermissionToCommunicate = false;

    private static UsbDevice mDevice;

```

```

private final BroadcastReceiver mUsbReceiver = new BroadcastReceiver() {

    public void onReceive(Context context, Intent intent) {

        String action = intent.getAction();

        if (ACTION_USB_PERMISSION.equals(action)) {

            synchronized (this) {

                UsbDevice device = intent.getParcelableExtra(UsbManager.EXTRA_DEVICE);

                if (intent.getBooleanExtra(UsbManager.EXTRA_PERMISSION_GRANTED, false)) {

                    if (device != null) {hasPermissionToCommunicate = true;}}

            }

        }

    }

};

```

@Override

```

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main);

    mUsbManager = (UsbManager) getSystemService(Context.USB_SERVICE);

    mPermissionIntent = PendingIntent.getBroadcast(this, 0, new
    Intent(ACTION_USB_PERMISSION), 0);

    IntentFilter filter = new IntentFilter(ACTION_USB_PERMISSION);

    registerReceiver(mUsbReceiver, filter);

    HashMap<String, UsbDevice> deviceList = mUsbManager.getDeviceList();

    Iterator<UsbDevice> deviceIterator = deviceList.values().iterator();

    while (deviceIterator.hasNext()) {

        mDevice = deviceIterator.next();

        if (mDevice.getVendorId() == 1137) {break;}

    }

}

```



```

mPermissionIntent = PendingIntent.getBroadcast(MainActivity.this, 0, new
Intent(ACTION_USB_PERMISSION), PendingIntent.FLAG_ONE_SHOT);

mUsbManager.requestPermission(mDevice, mPermissionIntent);

mUSB.GTSPL_setCmdSendMode("P");

mUSB.GTSPL_openPort(mUsbManager, mDevice);

mUSB.GTSPL_setup(62, 45, 2, 3, 0, 3, 0, this);

mUSB.GTSPL_setup(62.0, 45.0, 2, 3, 0, 3.0, 0.0, this);

mUSB.GTSPL_sendCommand(this, "DIRECTION 1\r\n");

mUSB.GTSPL_clearBuffer(this);

mUSB.GTSPL_barcode(30, 30, "128", 100, 1, 0, 2, 2, "barcode9463521", this);

mUSB.GTSPL_qrcode(300, 100, "H", 4, "A", 0, "ABCabc123", this);

mUSB.GTSPL_printerFont(100, 50, "2", 0, 1, 1, "PrintFontTest123", this);

mUSB.GTSPL_downloadBMP("LOGO.BMP", this);

mUSB.GTSPL_sendCommand(this, "PUTBMP 100,80,\"LOGO.BMP\"\\r\\n");

mUSB.GTSPL_download_Not1BitDepthBMP("printTest4.BMP", MainActivity.this);

mUSB.GTSPL_sendCommand(MainActivity.this, "PUTBMP10,10,\"PrintTest4.BMP\"\\r\\n");


String sBlock = "We stand behind our products with one of the most comprehensive
support programs in the Auto-ID industry.";

mUSB.GTSPL_printFontBlock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2",sBlock,this);

mUSB.GTSPL_printLabel(1, 1, this);

// initialization

mUSB.GTSPL_genericDefault(); // Initialize the general setting value

mUSB.GTSPL_sensorDefault(); // Initialize the sensor setting value

mUSB.GTSPL_rfidSetupDefault(); // Initialize the RFID setting value

//Modify Wifi frequency band

```

```

mUSB.GTSPL _ WifiFrequency("5G");

//Set print function (It needs to use with print)

mUSB.GTSPL_setDirectionAndMirror(1, 1); // Set direction and mirror.

mUSB.GTSPL_setShift(50); // Set the vertical displacement distance

mUSB.GTSPL_printReverse(10, 10, 160, 160); //Set reverse

mUSB.GTSPL_setOffset(20); // Set feed offset

mUSB.GTSPL_setCutMode(0, 2); // Set cut mode and cut number

mUSB.GTSPL_setAfterPrintAction(2); // Set action after print

    mUSB.GTSPL_printLabel(1, 1);

//Bitmap Print

mUSB.GTSPL_printBMP(-500,30,400,300,1,"CIRCLE.BMP ",this);

mUSB.GTSPL_printBMP_Compression(-500,30,400,300," CIRCLE.BMP ",this);

    mUSB.GTSPL_printLabel(1, 1);

//GEN2 RFID

mUSB.GTSPL_writeUHF("H", 2, 12, "E", "414142424343444445454646", this);

mUSB.GTSPL_EPCPWD_Action("L", "12345678", this);

mUSB.GTSPL_TIDPWD_Action("L", "12345678", this);

mUSB.GTSPL_USERPWD_Action("L", "12345678", this);

mUSB.GTSPL_AccessPWD_Action("S", "12345678", this);

mUSB. GTSPL_KillPWD_Action ("S", "12345678", this);

mUSB.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, this);

mUSB.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, "203", this);

mUSB.GTSPL_writeHF("H", 2, 12, "414142424343444445454646", this);

mUSB.GTSPL_printLabel(1, 1, this);

String uhfData = mUSB.GTSPL_readUHF("H",2,12,"E", this);

String uhfData = mUSB.GTSPL_readUHF("H",2,12,"E", 4000, this);

String uhfData = mUSB.GTSPL_readUHFQ("H",0,0,this);

```

```

//GJB RFID set password

// Use write password, set a new read password
mUSB.GTSPL_Set_GJB_Pwd("R","S","87654321","12345678",this);

// Use write password, set a new read password
mUSB.GTSPL_Set_GJB_Pwd("W","S","87654321","12345678",this);

// Use write password, set a new kill password
mUSB.GTSPL_Set_GJB_Pwd("K","S","87654321","12345678",this);

// Use write password, set a new status password
mUSB.GTSPL_Set_GJB_Pwd("S","S","87654321","12345678",this);

mUSB.GTSPL_printLabel(1, 1, this);

//GJB RFID Set status
mUSB.GTSPL_Set_GJB_Status("E","C","11112222",this);

mUSB.GTSPL_printLabel(1, 1, this);

//GJB RFID write EPC data
mUSB.GTSPL_writeGJB("H",2,12,"E","404041414242434344444545","12345678",this);

mUSB.GTSPL_printLabel(1, 1, this);

//GJB RFID read EPC data
String GJBData = mUSB.GTSPL_readGJB("H",2,12,"E","33334444",MainActivity.this);

//GJB RFID kill tag
mUSB.GTSPL_Kill_GJB_Tag("11224455",this);

mUSB.GTSPL_printLabel(1,1,this);

// Print Simplified Chinese
String stString="默认简体中文测试";

mUSB.GTSPL_clearBuffer(this);

mUSB.GTSPL_printerFont(100, 10, "TSS24.BF2", 0, 1, 1, stString, this);

mUSB.GTSPL_printLabel(1, 1, this);

// Print traditional Chinese

```

```
String ttString="默認繁體中文測試";

mUSB.GTSPL_clearBuffer(this);

mUSB.GTSPL_printerFont(100, 10, " TST24.BF2", 0, 1, 1, ttString, this);

mUSB.GTSPL_printLabel(1, 1, this);

String status = mUSB.GTSPL_printersStatus(1000);

mUSB.GTSPL_closePort();

String version= mUSB.GTSPL_getSDKVersion(0,this);

//RFID auto calibration

mUSB.GTSPL_LabelCalibration(this);

}
```

Appendix 1

Code Type	Description	Narrow : Width					Max. data length
		1:1	1:2	1:3	2:5	3:7	
128	Code 128, switching code subset automatically.	V					
128M	Code 128, switching code subset manually.	V					
EAN128	EAN128, switching code subset automatically.	V					
EAN128M	EAN128M, switching code subset manually.	V					
25	Interleaved 2 of 5.		V	V	V		Length is even
25C	Interleaved 2 of 5 with check digit.		V	V	V		Length is odd
25S	Standard 2 of 5.		V	V	V		
25I	Industrial 2 of 5.		V	V	V		
39	Code 39, switching standard and full ASCII mode automatically.		V	V	V		
39C	Code 39 with check digit.		V	V	V		
93	Code 93.			V			
EAN13	EAN 13.	V					12
EAN13+2	EAN 13 with 2 digits add-on.	V					14
EAN13+5	EAN 13 with 5 digits add-on.	V					17
EANB	EAN 8.	V					7
EANB+2	EAN 8 with 2 digits add-on.	V					96
EANB+5	EAN 8 with 5 digits add-on.	V					12
CODA	Codabar.		V	V	V		
POST	Postnet.	V					5,9,11
UPCA	UPC-A.	V					11
UPCA+2	UPC-A with 2 digits add-on.	V					13
UPA+5	UPC-A with 5 digits add-on.	V					16
UPCE	UPC-E.	V					6
UPCE+2	UPC-E with 2 digits add-on.	V					8
UPE+5	UPC-E with 5 digits add-on.	V					11
MSI	MSI.		V	V	V		
MSIC	MSI with check digit.		V	V	V		
PLESSEY	PLESSEY.		V	V	V		

CPOST	China post.					V	
ITF14	ITF14.		V	V	V		13
EAN14	EAN14.	V					13
11	Code 11.		V	V	V		
TELEPEN	Telepen. *Since V6.89EZ.		V	V	V		
TELEPENN	Telepen number. *Since V6.89EZ.		V	V	V		
PLANET	Planet. *Since V6.89EZ.	V					
CODE49	Code 49. *Since V6.89EZ.	V					
DPI	Deutsche Post Identcode. *Since V6.91EZ.		V	V	V		11
DPL	Deutsche Post Leitcode. *Since V6.91EZ.		V	V	V		13
LOGMARS	A special use of Code 39. *Since V6.88EZ.		V	V	V		

Appendix 2

Code Type	Description
100	other errors
101	memory space exceeded
102	memory is locked
103	Insufficient read power
104	non-specific error
105	CRC error
106	If an error occurs during writing , Reply how many words have been written
107	If the Tag tag replies with an error during writing , Error code plus how many words have been written
108	no tag exists
109	command format error
110	Failed to set power strength
111	Failed to set regulations