

CITIZEN

iOS POS Print SDK (Swift)

Programing Manual

For Ver. 2.12

CITIZEN SYSTEMS JAPAN CO., LTD.

Revision Record

Date	Version	Description
Aug 18, 2015	1.12	New issue
Dec 10, 2015	1.13	Added the description of the development environment to the supported terminals. Added CT-S310II to the support models.
Jan 29, 2016	1.14	Modified the description in the connect method, the disconnect method and the printerCheck method.
May 23, 2016	1.15	Added the description of "Add the SDK to Xcode" to the section "1.5 Definition method". Added the description of "Setting at the time of the archive file making" to the section "1.5 Definition method".
Dec 22, 2016	1.16	Added the section "2.38 MapMode property".
Jul 12, 2017	1.17	Added CT-D150/151 and CT-E351/651 to the support models. Added the setting contents of MSW13-1 and MSW13-5 to "CT-S601/651/801/851/601II/651II/801II/851II Series Memory Switch Setting" in the section "1.4 Supported models".
Nov 21, 2017		Added the description of "SDK type" to in the section "1.5 Definition method".
Apr 20, 2018		Added "Note" to the PrintBarCode method. Added the description of Swift 4.1 to "SDK type" of in the section "1.5 Definition method".
Jul 12, 2018	2.00	Added "1.5 Supported models (peripheral device)". Added "3. Linedisplay Control". Added "4. Barcode Scanner Control".
Oct 24, 2018	2.01	Adder CT-S751 to the support models. Added BC-NL3000U to the barcode scanner support models. Added the section "3.3.2 About SDK version when using line display". Added a section on "4.3.2 About SDK version when using barcode scanner".
Jan 16, 2019	2.02	Added the description about Bluetooth interface to the section "1.5 Supported models (peripheral device)". Added explanation of Bluetooth connection to "3.2.3 connect method". Added explanation of Bluetooth connection to "4.2.3 connect method".
Feb 25, 2019	2.03	Added CT-S4500 to the printer support models. Added the printPaddingText method. Added "2.3.2 About printing UTF-8 encode characters".
Apr 2, 2019		Added Swift5.0 (iOS12.2) to the SDK type.
Oct 7, 2019		Added Swift5.1 (iOS13.1/iPadOS13.1) to the SDK type.
Nov 12, 2019		Added Swift5.1.2 (iOS13.2/iPadOS13.2) to the SDK type.
Dec 12, 2019	2.04	Added CT-S2000 and CT-S4000 as searchable printers when CMP_PORT_WiFi is specified in the searchCitizenPrinter / searchESCPOSPrinter method.
Mar 16, 2020	2.06	Added the framework for the Swift 5.1 or later to the SDK type. Added section "Program Structure" to each device.
Oct 5, 2020	2.07	Added explanation of Lightning I/F. Added USB to interface of supported models (printers). Added USB to connect type of the connect method. Added the printTextLocalFont method. Added the setPrintCompletedTimeout method.
Jan 5, 2021		Modified the example of the connect method when connecting via USB.
Feb 9, 2021	2.08	Added CT-D101, CT-E301 and CT-E601 to the support models. Added the description of "1.6. Definition method" to XCFramework library. Added CMP_MODE_CMD_GRAY16DOWNLOAD to mode argument of the printBitmap method.
Nov 1, 2021	2.09	Added CMP Series to the support models. Added the Settings for searching LAN models (Bonjour). Added the explanation about italic specification of the printTextLocalFont method.

Apr 8, 2022	2.10	Added XCFramework for Swift5.3 (iOS14/iPadOS14) or later to SDK type. Modified the description of the printData method. Added description of log function.
Jun 30, 2023	2.11	Modified the description of the definition method. Added Near Empty and Drawer Status to the status method of the printer control. Added the printerCheckEx and openDrawerEx method to the printer control. Added ErrorCodeExtended property to the printer and the display and the scanner control.
Sep 13, 2023		Added CT-S801III and CT-S851III to the printer support models. Added DSP01-LT2 and DSP02-LS2 to the display support models.
Jan 10, 2024		Supported privacy manifest.
Jul 18, 2024	2.12	Updated the version number.

Notes

1. Unauthorized use of all or any part of this document is prohibited.
2. The information in this document is subject to change without prior notice.
3. This document has been created with full attention. If, however, you find an error or question, please contact us.
4. We shall not be liable for any effect resulting from operation regardless of the above item 3.
5. If you do not agree with the above terms, you are not permitted to use this driver.

Trademark

iOS is registered trademarks of Cisco in the United States and/or other countries.

iPod touch, Swift, and Xcode is registered trademarks of Apple Inc. in the United States and/or other countries.

Company names and product names appearing on this document are trademarks and/or registered trademarks of respective companies.

CITIZEN is a registered trademark of Citizen Watch Co., Ltd.

Table of Contents

1. Introduction	8
1.1. Document target range	8
1.2. System summary	8
1.3. Supported terminals	8
1.4. Supported models (Printers)	9
1.5. Supported models (Peripheral Devices)	19
1.6. Definition method	23
2. Printer Control	26
2.1. Program structure	26
2.2. Functions list	27
2.3. SDK interfaces	29
2.3.1. <i>Return value</i>	29
2.3.2. <i>Instance</i>	30
2.3.3. <i>connect method</i>	31
2.3.4. <i>disconnect method</i>	33
2.3.5. <i>setEncoding method</i>	34
2.3.6. <i>printerCheck method</i>	35
2.3.7. <i>status method</i>	36
2.3.8. <i>printText method</i>	38
2.3.9. <i>printPaddingText method</i>	39
2.3.10. <i>printTextLocalFont method</i>	41
2.3.11. <i>printBitmap/printBitmapData method</i>	42
2.3.12. <i>setNVBitmap method</i>	44
2.3.13. <i>printNVBitmap method</i>	46
2.3.14. <i>printBarCode method</i>	47
2.3.15. <i>printPDF417 method</i>	49
2.3.16. <i>printQRCode method</i>	50
2.3.17. <i>printGS1DataBarStacked method</i>	51
2.3.18. <i>cutPaper method</i>	52
2.3.19. <i>unitFeed method</i>	53
2.3.20. <i>markFeed method</i>	54
2.3.21. <i>openDrawer method</i>	55
2.3.22. <i>transactionPrint method</i>	56
2.3.23. <i>rotatePrint method</i>	57
2.3.24. <i>pageModePrint method</i>	58
2.3.25. <i>clearPrintArea method</i>	60
2.3.26. <i>clearOutput method</i>	61
2.3.27. <i>printData method</i>	62
2.3.28. <i>printNormal method</i>	63
2.3.29. <i>watermarkPrint method</i>	64
2.3.30. <i>searchCitizenPrinter method</i>	65
2.3.31. <i>searchESCPOSPrinter method</i>	67
2.3.32. <i>printerCheckEx method</i>	69
2.3.33. <i>openDrawerEx method</i>	71
2.3.34. <i>setPrintCompletedTimeout method</i>	72
2.3.35. <i>setLog method</i>	73
2.3.36. <i>getVersionCode method</i>	74

2.3.37. <i>getVersionName</i> method	75
2.3.38. <i>PageModeArea</i> property.....	76
2.3.39. <i>PageModePrintArea</i> property.....	77
2.3.40. <i>PageModePrintDirection</i> property	78
2.3.41. <i>PageModeHorizontalPosition</i> property	79
2.3.42. <i>PageModeVerticalPosition</i> property.....	80
2.3.43. <i>RecLineSpacing</i> property.....	81
2.3.44. <i>MapMode</i> property	82
2.3.45. <i>ErrorCodeExtended</i> property	83
2.4. Notes	84
2.4.1. <i>Function to detect the completion of printing</i>	84
2.4.2. <i>About printing UTF-8 encode characters</i>	84
2.4.3. <i>Logging function</i>	86
2.4.4. <i>Predefined Constants List</i>	88
3. Line Display Control	91
3.1. Program structure	91
3.2. Functions list	92
3.3. Library interfaces.....	93
3.3.1. <i>Return value</i>	93
3.3.2. <i>Instance</i>	94
3.3.3. <i>connect</i> method.....	95
3.3.4. <i>disconnect</i> method.....	97
3.3.5. <i>displayText</i> method	98
3.3.6. <i>clearDisplay</i> method.....	99
3.3.7. <i>blinkDisplay</i> method	100
3.3.8. <i>setDisplayMode</i> method.....	101
3.3.9. <i>setDisplayConfig</i> method.....	102
3.3.10. <i>setCursorPosition</i> method.....	103
3.3.11. <i>moveCursor</i> method.....	104
3.3.12. <i>setCursorType</i> method.....	105
3.3.13. <i>initializeDisplay</i> method.....	106
3.3.14. <i>displayData</i> method	107
3.3.15. <i>setEncoding</i> method	108
3.3.16. <i>setCodePage</i> method	109
3.3.17. <i>setInternationalCharacterSet</i> method.....	110
3.3.18. <i>displayCheck</i> method	111
3.3.19. <i>setLog</i> method.....	112
3.3.20. <i>getVersionCode</i> method.....	113
3.3.21. <i>getVersionName</i> method.....	114
3.3.22. <i>ErrorCodeExtended</i> property	115
3.4. Notes	116
3.4.1. <i>Logging function</i>	116
3.4.2. <i>Predefined Constants List</i>	118
3.4.3. <i>About SDK version when using Line Display</i>	118
4. Barcode Scanner Control.....	119
4.1. Program Structure	119
4.2. Functions list	120
4.3. Library interfaces.....	121
4.3.1. <i>Return value</i>	121

4.3.2. <i>Instance</i>	122
4.3.3. <i>connect method</i>	123
4.3.4. <i>disconnect method</i>	125
4.3.5. <i>setLog method</i>	126
4.3.6. <i>getVersionCode method</i>	127
4.3.7. <i>getVersionName method</i>	128
4.3.8. <i>ErrorCodeExtended property</i>	129
4.3.9. <i>DataEvent event</i>	130
4.3.10. <i>StatusUpdateEvent event</i>	131
4.4. Notes	132
4.4.1. <i>Logging function</i>	132
4.4.2. <i>Predefined Constants List</i>	134
4.4.3. <i>About SDK version when using Barcode Scanner</i>	134

1. Introduction

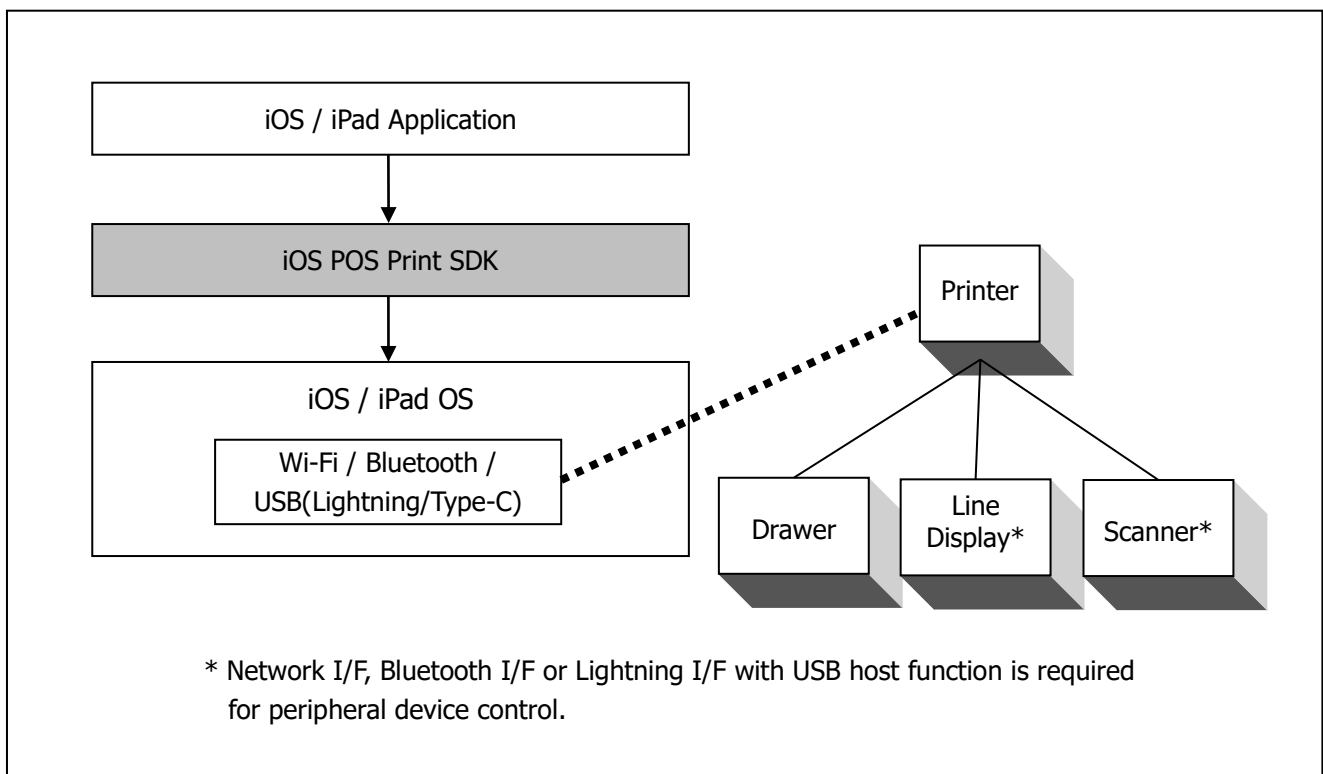
This document is a iOS POS Print SDK Programing Manual.

1.1. Document target range

This document is intended to be iOS or iPad OS application (Swift) developers to take advantage of CITIZEN POS printers.

1.2. System summary

This SDK assumes that it is referred from iOS application using CITIZEN POS printers.



System diagram of the SDK

1.3. Supported terminals

The specification of terminals supported by this SDK are as follows.

iOS version:	Supported interfaces:	Supported privacy manifest:
iOS 8.0 or newer	Wi-Fi, Bluetooth	Unsupported features
iOS 10.0.2 or newer	USB(Lightning/Type-C)	Unsupported features
iOS 12.0 or newer	Wi-Fi, Bluetooth, USB(Lightning/Type-C)	Supported features

1.4. Supported models (Printers)

The models supported by this SDK and the corresponding interfaces are as listed below.
Refer to the user's manual of the printer for the detailed functions of each model.

Series of Model	Object Model	Interface	Printer Functions
CT-D101 Series	CT-D101	Wired LAN	Standard
CT-D150 Series	CT-D150	Wired LAN	Standard
CT-D151 Series	CT-D151	Wired/Wireless LAN Bluetooth	Standard
	CT-D151-L	USB(Lightning/Type-C)	Label/Blackmark paper is supported.
CT-E301 Series	CT-E301	Wired LAN	Standard
CT-E351 Series	CT-E351	Wired LAN	Standard
CT-E601 Series	CT-E601	Wired/Wireless LAN Bluetooth USB(Lightning/Type-C)	Standard
CT-E651 Series	CT-E651	Wired/Wireless LAN Bluetooth	Standard
	CT-E651-L	USB(Lightning/Type-C)	Label/Blackmark paper is supported.
CT-S251 Series	CT-S251	Wired/Wireless LAN Bluetooth USB(Lightning/Type-C)	Standard
CT-S281 Series	CT-S281BD	Bluetooth	Standard
	CT-S281BD-M		Blackmark paper is supported.
	CT-S281BD-L		Label paper is supported.
CT-S310II Series	CT-S310II	Wired LAN	Standard
CT-S601/651/801/851 Series	CT-S601/651/801/851	Wired/Wireless LAN	Standard
	CT-S801/851-M		Blackmark paper is supported.
	CT-S801-L		Label paper is supported.
CT-S601II/651II/801II/851II Series	CT-S601II/651II/801II/851II	Wired/Wireless LAN Bluetooth	Standard
	CT-S801II/851II-M		Blackmark paper is supported.
	CT-S801II-L		Label paper is supported.
CT-S801III/851III Series	CT-S801III/851III	Wired/Wireless LAN Bluetooth	Standard
CT-S751 Series	CT-S751	Wired/Wireless LAN Bluetooth USB(Lightning/Type-C)	Standard
CT-S2000 Series	CT-S2000	Wired LAN	Standard
	CT-S2000-M		Blackmark paper is supported.
	CT-S2000-L		Label paper is supported.
CT-S4000 Series	CT-S4000	Wired LAN	Standard (Paper with blackmark on front side is supported.)
	CT-S4000-M		Paper with blackmark on back side is supported.
	CT-S4000-L		Label paper is supported.
CT-S4500 Series	CT-S4500	Wired/Wireless LAN Bluetooth USB(Lightning/Type-C)	Standard (Label/Blackmark paper is supported.)
CMP-20/30/20II/30II/40 Series	CMP-20/30/20II/30II/40 (ESC/POS)	Wireless LAN Bluetooth	Standard

It is the prerequisite for the use of this SDK that the memory switch of the printers are set as listed below.

CT-D101 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid

MSW No.	Function	Setting
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receive Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5

CT-D150 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receive Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5

CT-D151 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receive Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid

MSW No.	Function	Setting
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code Page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5
13-6	Auto Reconnect (When Bluetooth I/F is used)	Invalid

CT-E301 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5

CT-E351 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid

MSW No.	Function	Setting
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5

CT-E601 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code Page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5
13-6	Auto Reconnect (When Bluetooth I/F is used)	Invalid

CT-E651 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000 Mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid

MSW No.	Function	Setting
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code Page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5
13-6	Auto Reconnect (When Bluetooth I/F is used)	Invalid

CT-S251 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-3	Kanji	ON (*1)
9-4	JIS/Shift-JIS	Shift-JIS (*1)
13-6	Auto Reconnect (When Bluetooth I/F is used)	Valid

CT-S281 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	L/F enabled
3-7	CBM-270-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-3	USB Mode	Printer Class

MSW No.	Function	Setting
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-3	Kanji	ON (*1)
9-4	JIS/Shift-JIS	Shift-JIS (*1)
13-6	Auto Reconnect (When Bluetooth I/F is used)	Valid

CT-S310II Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-3	Kanji	ON (*1)
9-4	JIS/Shift-JIS	Shift-JIS (*1)

CT-S601/651/801/851 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-3	Parallel 31Pin	Reset
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-3	Kanji	ON (*1)
9-4	JIS/Shift-JIS	Shift-JIS (*1)

MSW No.	Function	Setting
10-3	ACK output timing	Before BUSY

CT-S601II/651II/801II/851II Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-3	Parallel 31Pin	Reset
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5
10-3	ACK Timing	Before BUSY
13-6	Auto Reconnect (When Bluetooth I/F is used)	Valid

CT-S801III/851III Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-3	Parallel 31Pin	Reset
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)

MSW No.	Function	Setting
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5
10-3	ACK Timing	Before BUSY
13-6	Auto Reconnect (When Bluetooth I/F is used)	Valid

CT-S751 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5-HKSCS
13-6	Auto Reconnect (When Bluetooth I/F is used)	Invalid

CT-S2000 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-3	Parallel 31Pin	Reset
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Disabled
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)

MSW No.	Function	Setting
9-2	Int'Char Set	Japan (*1)
9-3	Kanji	ON (*1)
9-4	JIS/Shift-JIS	Shift-JIS (*1)
10-3	ACK Timing	Before BUSY

CT-S4000 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-3	Parallel 31Pin	Reset
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-3	Kanji	ON (*1)
9-4	JIS/Shift-JIS	Shift-JIS (*1)
10-3	ACK Timing	Before BUSY

CT-S4500 Series Memory Switch Setting

MSW No.	Function	Setting
1-1	Power ON Info	Valid
1-2	Buffer Size	4K bytes
1-3	Busy condition	Full
1-4	Receiving Error	Print ?
1-5	CR Mode	Ignored
2-2	Auto cutter	Valid
2-4	Full Col Print	Wait Data
3-1	Resume Ctrr Err	Valid
3-7	CBM1000-compatible mode	Valid
3-8	Resume Open Err	Close
4-8	Partial Only	Invalid
5-2	Line Pitch	1/360
5-3	USB Mode	Printer Class
6-1	Act. For Driver	Valid
7-6	DMA control	Valid
9-1	Code page	Katakana (*1)
9-2	Int'Char Set	Japan (*1)
9-4	Multi-byte Char (*2)	SJIS(CP932) GB18030 EUC Hangul BIG5-HKSCS

MSW No.	Function	Setting
13-6	Auto Reconnect (When Bluetooth I/F is used)	Invalid

*1 MSW No.9-1~4 is the setting when using Japanese. Please change it by use environment.

*2 The CT-D101/150/151, CT-E301/351/601/651, CT-S601II/651II/801II/851II/801III/851III/751/4500 series can change the Multi-byte character to Shift_JIS, GB18030, EUC-KR and Big5. Please change it by use environment.

Firmware

The firmware version of the printer has to be the following condition to use of this SDK in CT-S601/651/801/851 Series.

With the older printer than following, it is necessary to update the firmware.

Model	Firmware Version
CT-S601	DL00-2000 or newer
CT-S651	DM00-2000 or newer
CT-S801	DH00-2000 or newer
CT-S851	DK00-2000 or newer

1.5. Supported models (Peripheral Devices)

The models of peripheral devices applicable for control with this service are as follows.

For details on the functions of each model, refer to the instruction manual of each peripheral device.

Network I/F, Bluetooth I/F or Lightning I/F with USB host function is required for peripheral device control.

[Line Display]

Applicable Display	I/F	Product Specification Overview
DSP01-LT / DSP01-LT2	USB	TFT line display
DSP02-LS / DSP02-LS2	USB	STN line display

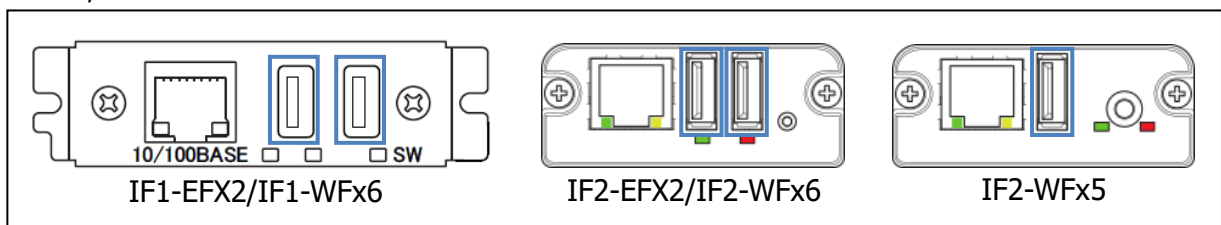
[Barcode Scanner]

Applicable Display	I/F	Product Specification Overview
SCN01-Z1D	USB	1D barcode scanner
SCN02-Z2D	USB	2D barcode scanner
BC-NL3000U	USB	2D barcode scanner

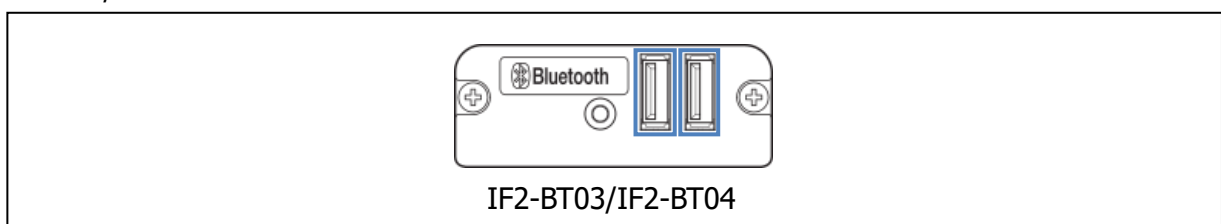
About connection to printer

For the connection to the applicable peripheral device, first turn off the printer power and then connect to a USB port of the corresponding interface shown in the figure below. Next, turn on the printer power, wait about 30 seconds until the applicable peripheral device becomes ready to use so as to ensure stable operation, and then execute the control start process of the peripheral device.

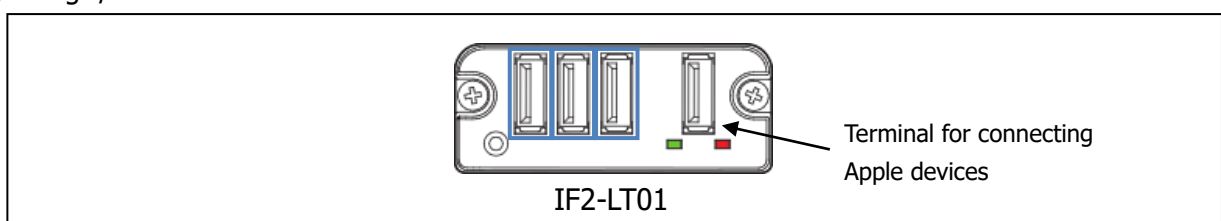
Network I/F



Bluetooth I/F



Lightning I/F



The following lists prohibited actions that must not be performed with regard to a peripheral device connection.

Prohibited Actions

- Connecting other than a supported peripheral device (USB hub, smartphone, etc.) to a USB port of the interface.
- Inserting and removing the cable connector of the peripheral device into/from a USB port of the interface while the printer power is on.
- Connecting multiple peripheral devices of the same type to a USB port of the interface (e.g. connecting two displays).
- Connecting iPad/iPod/iPhone to the USB terminal for peripheral devices of Lightning I/F.

If any of the above actions is performed, it may lead to the misoperation and, in the worst case, cause a failure of the printer or connected peripheral device.

About the Network I/F setting

When using the line display and the barcode scanner with the Network I/F, it is necessary to change the setting related to the service. For the basic operation, refer to the instruction manual of the interface board of the printer.

Please connect to each printer from web browser and display the following Service screen. You can set the services provided by the printer.

LAN board CITIZEN SYSTEMS

HOME | STATUS | CONFIG Logout

General Service User Account Maintenance

Media Converter

VCOM Convert ☒ Enable ☐ Disable ☐ Show configuration

HID Scanner Convert ☒ Enable ☐ Disable ☐ Show configuration

XML Print

Port Number 8080

Timeout for connect 10 5-60[Seconds]

Timeout for print 60 10-600[Seconds]

XML Device Control

Port Number 8085

Timeout for connect 10 5-180[Seconds]

Maxconnection 2

XML Device Control /Line Display

Baud rate 9600

Data 8 bit

Copyright © 2012 CITIZEN SYSTEMS JAPAN CO.,LTD. All rights reserved.

Select "Enable" of "VCOM Converter" and "HID Scanner Convert" with reference to the inside of the red frame. Then scroll to the bottom and press the "Submit" button.

Finally, press the "Save & Reboot" button on the "Maintenance" tab, select "Yes", and when the buzzer beeps from the printer, the setting is completed

When checking "Show configuration" in the above red frame, the setting screen of "Media Converter Configuration / VCOM Convert" is displayed, but since it already has an appropriate value for the corresponding display, it is not changed by normal use. Please do.

Each setting value holds the value even when the power is turned off. When factory default setting (Factory Default) processing is done, set each setting value to the initial value.

About the Barcode Scanner setting







When using barcode scanner, barcode scanner must be set as follows.

Item	Value		Description
	Network or Bluetooth I/F	Lightning I/F	
Interface	USB HID Class	USB virtual COM	Communication protocol
Keyboard	US Keyboard	US Keyboard	Keyboard language
Terminator	Enter	<CR><LF>	Data suffix

When using a barcode scanner with this SDK, it is necessary to change the communication protocol setting of the barcode scanner by the interface of the printer. The setting method is as follows.


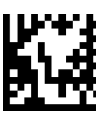
SCN01-Z1D

Scan all the bar codes below from the top and change the settings.

Network or Bluetooth I/F	Lightning I/F
 Start setting	 Start setting
 Change to USB HID	 Change to USB virtual COM
 End setting	 End setting







SCN01-Z1D

Scan all the bar code and change the settings.

Network or Bluetooth I/F	Lightning I/F
 Change to USB HID	 Change to USB virtual COM

BC-NL3000U

Scan all the bar codes below from the top and change the settings.

Network or Bluetooth I/F	Lightning I/F
 0006010 Start setting	 0006010 Start setting
 1100080 Change to USB HID	 1100060 Change to USB virtual COM
 0006000 End setting	 0006000 End setting

1.6. Definition method

SDK type

Select according to the development environment of the SDK (Framework).

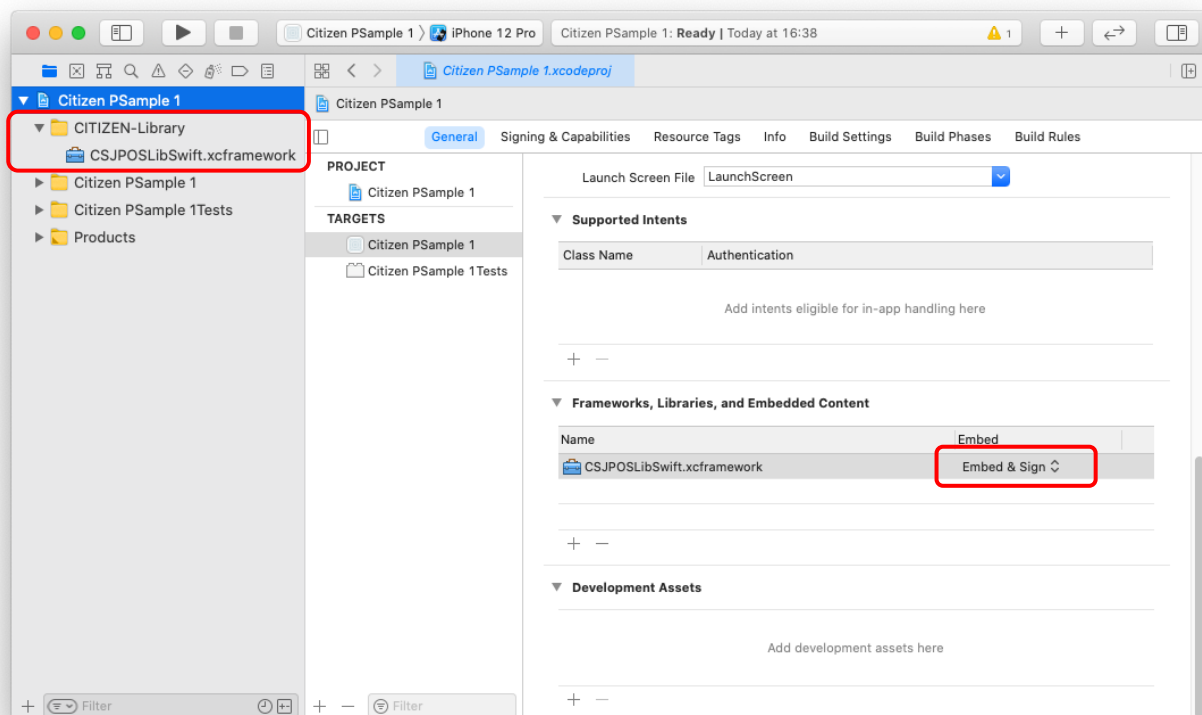
SDK (Framework)	development environment	Target OS
Swift 5.1 or later (XCFramework)	Xcode 11.0 or later	Depends on Xcode version
Swift 5.3 or later (XCFramework)	Xcode 12.0 or later	Depends on Xcode version
Swift 5.9 or later Privacy manifest support version (XCFramework)	Xcode 15.0 or later	iOS 12.0 or later

* Stopped providing frameworks for Swift 5.0 or earlier from version 2.11.

Add the SDK to Xcode

Add the framework file (CSJPOSLibSwift.xcframework) to the project to develop.

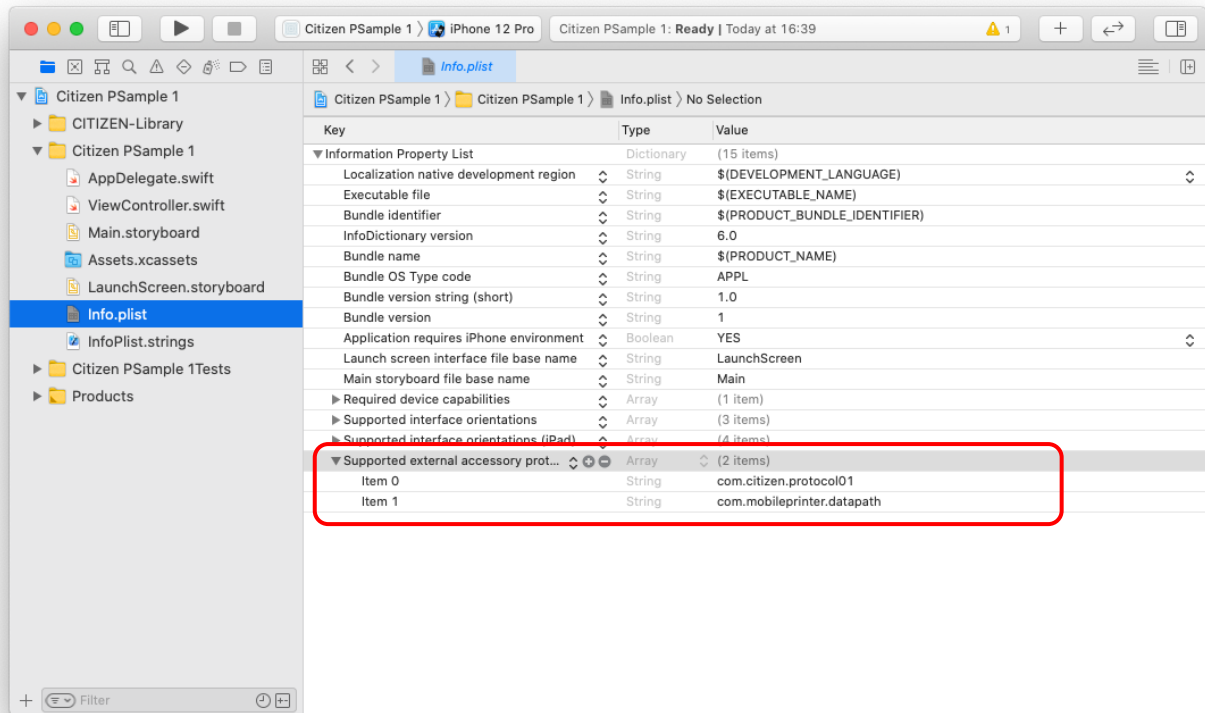
Drag the framework file from the 'Finder' to any place in the "Project Navigator" of Xcode.



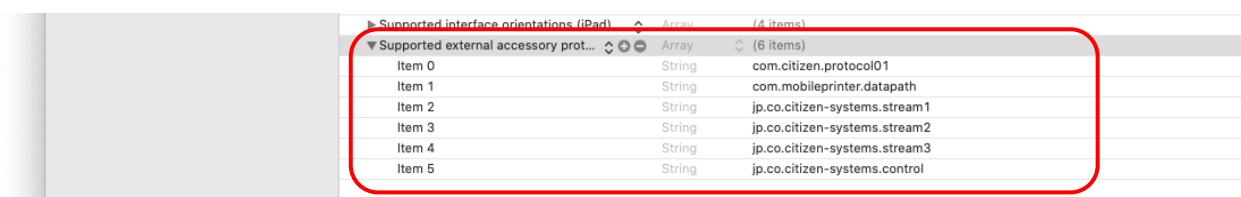
Select the project file from "Project Navigator". Then, change the "Embed" setting of the framework you just added displayed in "Frameworks, Libraries, and Embedded Content" of "TARGET"-General to "Embed & Sign".

Set the development of Bluetooth or USB(Lightning/Type-C)

Protocol name is added. "Info.plist" file is opened from "Project Navigator". "Add Row" is chosen from the pop-up menu (Open when Control + click), "Supported external accessory protocols" is selected and added. After that, "Supported external accessory protocols" is opened, and "com.citizen.protocol01" and "com.mobileprinter.datapath" are input to the column.



In addition, when using peripheral devices with Lightning I/F, add "jp.co.citizen-systems.stream1", "Item 1", "jp.co.citizen-systems.stream2", "jp.co.citizen-systems.stream3" and "jp.co.citizen-systems.control" as needed.

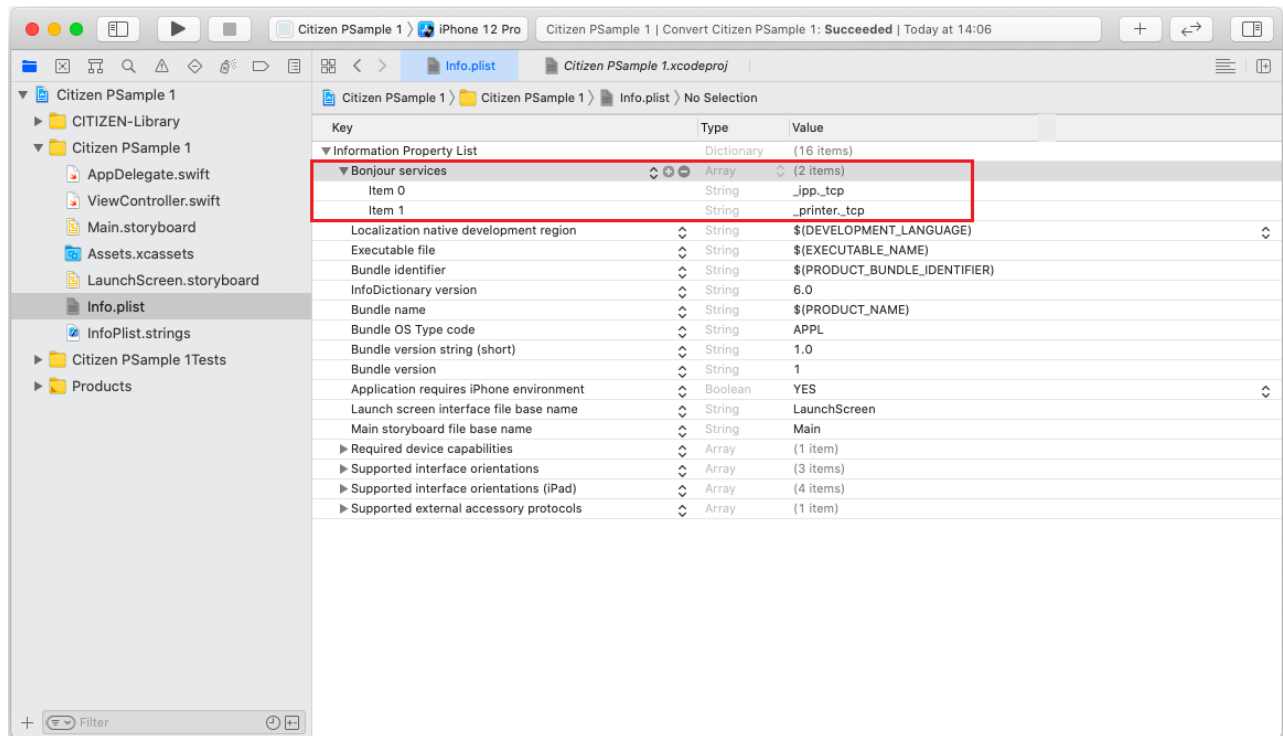


It is completion in the above.

Settings for searching LAN models (Bonjour)

When using the methods (searchESCPOSPrinter, searchCitizenPrinter) that search for LAN devices using the SDK of V209 or higher, please set your App to use the Bonjour service.

Open the "Info.plist" file of the App from "Project Navigator". Click "Information Property List" and add "Bonjour service". Add the character string "_ipp._tcp" to the "Item 0" field of "Bonjour service" and the character string "_printer._tcp" to the "Item 1" field.



It is completion in the above.

2. Printer Control

2.1. Program structure

Here is an example program which uses the SDK.

<pre>// Create an instance. var printer: ESCPOSPrinter? = CSJPOSLibSwift.ESCPOSPrinter() var result: Int32 = 0 // Connect printer result = printer!.connect(ESCPOSConst.CMP_PORT_WiFi, withAddress: "192.168.0.10") if ESCPOSConst.CMP_SUCCESS == result { // Set encoding _ = printer!.setEncoding(String.Encoding.shiftJIS) // Start Transaction (Batch) _ = printer!.transactionPrint(ESCPOSConst.CMP_TP_TRANSACTION) // Print Text _ = printer!.printText("- Sample Print 1 -\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_CENTER, withAttribute: ESCPOSConst.CMP_FNT_DEFAULT, withTextSize: ESCPOSConst.CMP_TXT_1WIDTH ESCPOSConst.CMP_TXT_2HEIGHT) _ = printer!.printText("1234567890123456789012345678901234567890\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_RIGHT, withAttribute: ESCPOSConst.CMP_FNT_DEFAULT, withTextSize: ESCPOSConst.CMP_TXT_1WIDTH ESCPOSConst.CMP_TXT_1HEIGHT) // Print QRcode _ = printer!.printQRCode("http://www.citizen-systems.co.jp", withModuleSize:6, withECLevel: ESCPOSConst.CMP_QRCODE_EC_LEVEL_L, withAlignment: ESCPOSConst.CMP_ALIGNMENT_RIGHT) // Partial Cut with Pre-Feed _ = printer!.cutPaper(ESCPOSConst.CMP_CUT_PARTIAL_PREFEED) // End Transaction (Batch) result = printer!.transactionPrint(ESCPOSConst.CMP_TP_NORMAL) // Disconnect _ = printer!.disconnect() if ESCPOSConst.CMP_SUCCESS != result { messageBox("DisplayText Error : \(result)", withTitle: "Error", withAutoDismiss: true) } }else{ // Connect Error messageBox("Connect Error : \(result)", withTitle: "Error", withAutoDismiss: true) }</pre>	<div style="font-size: 2em;">}</div> Class definition <div style="font-size: 2em;">}</div> Connect <div style="font-size: 2em;">}</div> Print processes <div style="font-size: 2em;">}</div> Disconnect
---	---

2.2. Functions list

This SDK provides the following functions.

Methods list

No	Function	Detail
1	Create class (instance)	This is instance method.
2	Connect printer (connect method)	Connect to the printer.
3	Disconnect printer (disconnect method)	Disconnect the printer connection.
4	Set encoding (setEncoding method)	Set the encoding of character.
5	Check printer status (printerCheck method)	Sends command for status check of the printer.
6	Get printer status (status method)	Get the status of the printer.
7	Print text (printText method)	Prints text data.
8	Print space padding text (printPaddingText method)	Prints text data with space padding.
9	Print local font text (printTextLocalFont method)	Prints text data using a font installed in the terminal.
10	Print bitmap (printBitmap/printBitmapData method)	Prints a bitmap file. (BMP/JPG/PNG/GIF format)
11	Store NV bitmap (setNVBitmap method)	Stores a bitmap image in the flash memory.
12	Print NV bitmap (printNVBitmap method)	Prints a bitmap image that is stored in the flash memory.
13	Print BarCode (printBarCode method)	Prints a one-dimensional barcode.
14	Print PDF-417 (printPDF417 method)	Prints a PDF417 barcode.
15	Print QRcode (printQRCode method)	Prints a QRCode barcode.
16	Print 2D GS1DataBar (printGS1DataBarStacked method)	Prints a 2-dimensional GS1DataBar barcode.
17	Cut paper (cutPaper method)	Cuts the paper.
18	Feed dot units (unitFeed method)	Feeds the paper forward by dot units.
19	Feed mark (markFeed method)	Support for label / black mark paper.
20	Open drawer (openDrawer method)	Opens the drawer.
21	Transaction print (transactionPrint method)	Enters or exits transaction mode.
22	Rotate print (rotatePrint method)	Enters or exits rotated print mode. (180°)
23	PageMode print (pageModePrint method)	Enters or exits page mode.
24	PageMode clear print area (clearPrintArea method)	Clear the area of the page mode print area.
25	Clear output data (clearOutput method)	Clears all buffered output data. (data and printer buffer)

26	Output data (printData method)	Sends to the printer without changing the data.
27	Print OPOS format (printNormal method)	Prints text using OPOS escape sequences.
28	Watermark print (watermarkPrint method)	Enters or exits watermark print mode.
29	Search printer (searchCitizenPrinter method)	Search the printer and get the list of the printer information.
30	Search printer (searchESCPOSPrinter method)	Search the printer and get the list of addresses.
31	Connect and Check printer status (printerCheckEx method)	Connect and get the status of the printer.
32	Connect and Open Drawer (openDrawerEx method)	Connect and open the drawer.
33	Set print completed timeout (setPrintCompletedTimeout method)	Set the timeout to check the print completion notification.
34	Log function (setLog method)	Set the log function.
35	Get version code (getVersionCode method)	Get a numerical value for the version number of this SDK.
36	Get version name (getVersionName method)	Get a string for the version number of this SDK.

Properties List

No	Function	Attribute	Detail
1	PageMode area (PageModeArea property)	R	Shows the page area of page mode.
2	PageMode print area (PageModePrintArea property)	R/W	Shows the print area of page mode.
3	PageMode print direction (PageModePrintDirection property)	R/W	Shows the print direction of page mode.
4	PageMode horizontal position (PageModeHorizontalPosition property)	R/W	Shows the horizontal start position offset within the print area of page mode.
5	PageMode vertical position (PageModeVerticalPosition property)	R/W	Shows the vertical start position offset within the print area of page mode.
6	Line spacing (RecLineSpacing property)	R/W	Shows the spacing of each single-high print line.
7	Mapping mode (MapMode property)	R/W	Show the mapping mode (the unit of measure) of the printer.
8	Extended error code (ErrorCodeExtended property)	R	Show the extended error code when connecting.

2.3. SDK interfaces

The following are the interfaces of this SDK.

2.3.1. Return value

Methods to be described later return the value in the list below.

Return value	Description
CMP_SUCCESS (0)	The operation is success.
CMP_E_CONNECTED (1001)	The printer is already connected.
CMP_E_DISCONNECT (1002)	The printer is not connected.
CMP_E_NOTCONNECT (1003)	Failed connection to the printer.
CMP_E_CONNECT_NOTFOUND (1004)	Failed to check the support model after connecting to the device.
CMP_E_CONNECT_OFFLINE (1005)	Failed to check the printer status after connecting to the device.
CMP_E_ILLEGAL (1101)	Unsupported operation with the Device, or an invalid parameter value was used.
CMP_E_OFFLINE (1102)	The printer is off-line.
CMP_E_NOEXIST (1103)	The file name does not exist.
CMP_E_FAILURE (1104)	The Service cannot perform the requested procedure.
CMP_E_TIMEOUT (1105)	The Service timed out waiting for a response from the printer.
CMP_E_NO_LIST (1106)	The printer cannot be found in the printer search.
CMP_EPTR_COVER_OPEN (1201)	The cover of the printer opens.
CMP_EPTR_REC_EMPTY (1202)	The printer is out of paper.
CMP_EPTR_BADFORMAT (1203)	The specified file is in an unsupported format.
CMP_EPTR_TOOBIG (1204)	The specified bitmap is either too big.

2.3.2. Instance

Syntax

ESCPOSPrinter

Description

To generate the class and to initialize it.

Example

```
var escp: ESCPOSPrinter? = CSJPOSLibSwift.ESCPOSPrinter()
```

2.3.3. connect method

Syntax

- 1) func connect(connectType: Int32, withAddress addr: String?) -> Int32
- 2) func connect(connectType: Int32, withAddress addr: String?, withPort port: Int32) -> Int32
- 3) func connect(connectType: Int32, withAddress addr: String?, withPort port: Int32, withTimeout timeout: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
connectType	[IN]	Connect type	CMP_PORT_WiFi: Wi-Fi connection CMP_PORT_BLUETOOTH: Bluetooth connection CMP_PORT_USB: USB connection (Lightning/Type-C)
addr	[IN]	IP address to connect or Bluetooth device address or Bluetooth device name or serial number	WiFi: 0.0.0.0 - 255.255.255.255 Bluetooth: 00:00:00:00:00:00 – FF:FF:FF:FF:FF:FF Device name USB: Blank or serial number
port	[IN]	Connection port number	
timeout	[IN]	Timeout (msec)	

Description

This method is used to connect the printer. Please specify the type and address of the printer connection.

When Bluetooth is used, it is possible to connect with only the printer with which pairing and connecting by the BLUETOOTH setting of the iOS device (Please refer to "1.6 Use of Bluetooth"). And, it should be iOS6 or more to be connected by using the Bluetooth device address. Please connect by using the Bluetooth device name besides.

When USB is used, if you specify a space for the address, the connection will be established automatically. It is also to connect to a specific printer by specifying the printer serial number.

Connection port number is valid only if Wi-Fi is specified for the connection type. If it is omitted, it connects with number 9100.

Timeout is gives the maximum number of milliseconds to connect printer. If it is omitted, it connects with 4000 milliseconds in the case of Wi-Fi/USB and 8000 milliseconds in the case of Bluetooth.

When connecting to the printer, this SDK also checks the status of the printer and the supporting models.

When communication with the printer is not necessary, must execute the [disconnect method](#) to disconnect the printer connection. When not disconnect, the next connection will be an error.

Return value

Return CMP_SUCCESS (0) in success. Please check the description of the error codes below in the case of failure. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Error codes	Description
CMP_E_NOTCONNECT (1003)	Failed connection to the printer. (1) The printer is under none-connection status. (2) The printer is not turned ON. (3) Cannot obtain handle of interface board.

CMP_E_CONNECT_NOTFOUND (1004)	Failed to check the support model after connecting to the printer. (1) The model is not supported.
CMP_E_CONNECT_OFFLINE (1005)	Failed to check the printer status after connecting to the printer. The printer is connected but the following errors occurred. (1) The cover of the printer opens. (2) The printer is out of paper. (3) Auto Cutter Error occurred due to paper jam, etc. (4) Unrecoverable error occurred due to circuit failure, etc.

Example

```

escp!.connect(ESCPOSConst.CMP_PORT_WiFi, withAddress: "192.168.182.100",
              withPort: 9100)

escp!.connect(ESCPOSConst.CMP_PORT_BLUETOOTH,
              withAddress: "00:01:90:F0:81:AB", withPort: 9100, withTimeout:8000)

escp!.connect(ESCPOSConst.CMP_PORT_BLUETOOTH,
              withAddress: "CITIZEN SYSTEMS", withPort: 9100, withTimeout:8000)

escp!.connect(ESCPOSConst.CMP_PORT_USB, withAddress: "")

```


2.3.4. disconnect method

Syntax

```
func disconnect() -> Int32
```

Parameter

Not exist.

Description

This method is used to disconnect the printer connection.

When the end of the print or some kind of error occurs, please disconnect the connection by the execution of this method.

Return value

Return CMP_SUCCESS(0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.disconnect()
```

2.3.5. setEncoding method

Syntax

```
func setEncoding(charset: NSStringEncoding) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
Charset	[IN]	Character set name	

Description

This method is used to set the encoding of the send data to the printer.

When you create an instance, it is initialized to the default character set of the OS.

Please set the encoding by the setting of the memory switch of the printer. (Please refer to "[1.4 Supported models \(Printers\)](#)")

This SDK supports printing UTF-8 encoded characters. Please refer to "[2.4.2 About printing UTF-8 encode characters](#)" for the detail.

Please refer to the following URL for the details of the setting range.

http://developer.apple.com/library/mac/#documentation/Cocoa/Reference/Foundation/Classes/NSString_Class/Reference/NSString.html#//apple_ref/doc/uid/20000154-BAJJAICE

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.setEncoding(String.Encoding.shiftJIS)
```

```
let encode = CFStringConvertEncodingToNSStringEncoding( 0x0632 )
escp!.setEncoding(String.Encoding(rawValue:encode))
```

```
let encode = CFStringConvertEncodingToNSStringEncoding( 0x0A03 )
escp!.setEncoding(String.Encoding(rawValue:encode))
```

```
let encode = CFStringConvertEncodingToNSStringEncoding( 0x0940 )
escp!.setEncoding(String.Encoding(rawValue:encode))
```

```
escp!.setEncoding(String.Encoding.utf8)
```

2.3.6. printerCheck method

Syntax

```
func printerCheck() -> Int32
```

Parameter

Not exist.

Description

This method is used to send the command to get the status of the printer.

If the result of this method is successful, you can get the status of the printer by [status method](#).

If the result of this method is failure, there is a possibility that the connection or the printer abnormality has occurred. In this case, please reconnect using the [disconnect method](#) and the [connect method](#).

If you want to print after the connected and some time passed, please check the status of the printer by the execution of this method and the [status method](#) beforehand.

In the case of network connection, it is automatically disconnected when passed a long time. If you want to keep a connection, please execute this method regularly.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
if ESCPOSCONST.CMP_SUCCESS == escp!.printerCheck() {  
    // Success  
} else {  
    // Fail  
}
```

2.3.7. status method

Syntax

- 1) func status() -> Int32
- 2) func status(musk: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
musk	[IN]	Status type mask	CMP_STS_PRINTEROFF CMP_STS_MSR_READ CMP_STS_PAPER_EMPTY CMP_STS_COVER_OPEN CMP_STS_BATTERY_LOW CMP_STS_PAPER_NEAREMPTY CMP_STS_DRAWER_LEVEL_H

Description

This method is used to get the status of the printer obtained by the [printerCheck method](#).

Before the execution of this method, you must run the [printerCheck method](#).

When there is not a parameter, return the logical sum of the status (CMP_STS_COVER_OPEN, CMP_STS_PAPER_EMPTY, CMP_STS_PRINTEROFF) indicating the error of the printer.

When the status type is specified, return the status that matches. Status type can be specified in combination. If you want to combine, please specify the logical sum.

Return value

Return the following status codes.

status codes	Description
CMP_STS_NORMAL (0)	The printer is normal.
CMP_STS_PRINTEROFF (128)	The printer is off-line.
CMP_STS_MSR_READ (64)	Currently MSR in read mode. (CMP-20/30/20II/30II/40 only)
CMP_STS_PAPER_EMPTY (32)	The printer is out of paper.
CMP_STS_COVER_OPEN (16)	The cover of the printer opens.
CMP_STS_BATTERY_LOW (8)	Printer battery capacity is low. (CMP-20/30/20II/30II/40 only)
CMP_STS_PAPER_NEAREMPTY (4)	Paper near empty. (when the type parameter is set)
CMP_STS_DRAWER_LEVEL_H (2)	Status of pin 3 of drawer kick-out connector = H (when the type parameter is set)

Example

```
var status = escp!.status()
if ESCPOSConst.CMP_STS_NORMAL == status {
    // No Error
    var status2 = escp!.status(ESCPOSConst.CMP_STS_PAPER_NEAREMPTY)
    if (ESCPOSConst.CMP_STS_PAPER_NEAREMPTY & status2) > 0 {
        // Paper Near Empty
    }
} else {
    if (ESCPOSConst.CMP_STS_COVER_OPEN & status) > 0 {
        // Cover Open
    }
}
```

```
    if (ESCPOSConst.CMP_STS_PAPER_EMPTY & status) > 0 {  
        // Paper Empty  
    }  
    if (ESCPOSConst.CMP_STS_PRINTEROFF & status) > 0 {  
        // Printer Offline  
    }  
}  
  
var status3 = escp!.status(ESCPOSConst.CMP_STS_DRAWER_LEVEL_H)  
if (ESCPOSConst.CMP_STS_DRAWER_LEVEL_H & status3) > 0 {  
    // Status of pin 3 of drawer kick-out connector = H  
}
```

2.3.8. printText method

Syntax

```
func printText(data: String?, withAlignment alignment: Int32, withAttribute attribute: Int32,
               withTextSize textSize: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Text data	
alignment	[IN]	Text alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment
attribute	[IN]	Text attribute	CMP_FNT_DEFAULT: Default font CMP_FNT_FONTB: Font B CMP_FNT_FONTC: Font C CMP_FNT_BOLD: Bold CMP_FNT_REVERSE: Reverse CMP_FNT_UNDERLINE: Underline
textSize	[IN]	Text size	CMP_TXT_1WIDTH: 1 times width CMP_TXT_2WIDTH: 2 times width CMP_TXT_3WIDTH: 3 times width CMP_TXT_4WIDTH: 4 times width CMP_TXT_5WIDTH: 5 times width CMP_TXT_6WIDTH: 6 times width CMP_TXT_7WIDTH: 7 times width CMP_TXT_8WIDTH: 8 times width CMP_TXT_1HEIGHT: 1 times height CMP_TXT_2HEIGHT: 2 times height CMP_TXT_3HEIGHT: 3 times height CMP_TXT_4HEIGHT: 4 times height CMP_TXT_5HEIGHT: 5 times height CMP_TXT_6HEIGHT: 6 times height CMP_TXT_7HEIGHT: 7 times height CMP_TXT_8HEIGHT: 8 times height

Description

This method is used to print text which specifies alignment and attribute and size.

Text attribute can be specified in combination font B, font C, bold, reverse, and underline. If you want to combine, please specify the logical sum.

Text size can be specified in combination with the width and height. If you want to combine, please specify the logical sum.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.printText("Print text data.\n",
               withAlignment: ESCPOSConst.CMP_ALIGNMENT_CENTER,
               withAttribute: ESCPOSConst.CMP_FNT_BOLD,
               withTextSize:
               ESCPOSConst.CMP_TXT_2WIDTH|ESCPOSConst.CMP_TXT_2HEIGHT)
```

2.3.9. printPaddingText method

Syntax

```
func printPaddingText(data: String?, withAttribute attribute: Int32, withTextSize textSize: Int32,
    withLength length: Int32, withSide side: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Text data	
attribute	[IN]	Text attribute	CMP_FNT_DEFAULT: Default font CMP_FNT_FONTB: Font B CMP_FNT_FONTC: Font C CMP_FNT_BOLD: Bold CMP_FNT_REVERSE: Reverse CMP_FNT_UNDERLINE: Underline
textSize	[IN]	Text size	CMP_TXT_1WIDTH: 1 times width CMP_TXT_2WIDTH: 2 times width CMP_TXT_3WIDTH: 3 times width CMP_TXT_4WIDTH: 4 times width CMP_TXT_5WIDTH: 5 times width CMP_TXT_6WIDTH: 6 times width CMP_TXT_7WIDTH: 7 times width CMP_TXT_8WIDTH: 8 times width CMP_TXT_1HEIGHT: 1 times height CMP_TXT_2HEIGHT: 2 times height CMP_TXT_3HEIGHT: 3 times height CMP_TXT_4HEIGHT: 4 times height CMP_TXT_5HEIGHT: 5 times height CMP_TXT_6HEIGHT: 6 times height CMP_TXT_7HEIGHT: 7 times height CMP_TXT_8HEIGHT: 8 times height
length	[IN]	Length	1 -
side	[IN]	Side	CMP_SIDE_RIGHT: Right side of text data CMP_SIDE_LEFT: Left side of text data

Description

This method is used to print text with space padding which specifies attribute and size and length of the single-byte character equivalent and side where space is added.

Cannot use the combining characters in the text data.

Text attribute can be specified in combination font B, font C, bold, reverse, and underline. If you want to combine, please specify the logical sum.

Text size can be specified in combination with the width and height. If you want to combine, please specify the logical sum.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
let nameSize = 24;    // Order name size
let priceSize = 7;    // Price size

// Line 1
escp!.printPaddingText("Sandwich",
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH, withLength: nameSize,
    withSide: ESCPOSConst.CMP_SIDE_RIGHT)
escp!.printPaddingText("5.00", withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: CMP_TXT_1WIDTH, withLength: priceSize,
    withSide: ESCPOSConst.CMP_SIDE_LEFT)
escp!.printNormal("\n")

// Line 2
escp!.printPaddingText("Hamburg steak",
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH, withLength: nameSize,
    withSide: ESCPOSConst.CMP_SIDE_RIGHT)
escp!.printPaddingText("12.00", withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH, withLength: priceSize,
    withSide: ESCPOSConst.CMP_SIDE_LEFT)
escp!.printNormal("\n")

// Line 3
escp!.printPaddingText("Coffee", withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH, withLength: nameSize,
    withSide: ESCPOSConst.CMP_SIDE_RIGHT)
escp!.printPaddingText("2.00", withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH, withLength: priceSize,
    withSide: ESCPOSConst.CMP_SIDE_LEFT)
escp!.printNormal("\n")
```


2.3.10. printTextLocalFont method

Syntax

```
func printTextLocalFont(data: String?, withAlignment alignment: Int32,
    withFontName fontName: String?, withPoint point: Int32, withStyle style: Int32,
    withHRatio hRatio: Int32, withVRatio vRatio: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Text data	
alignment	[IN]	Text alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment
fontName	[IN]	Font Name	The supported fonts depend on the OS implementation.
point	[IN]	Font size [Points]	1 -
style	[IN]	Font style	CMP_FNT_DEFAULT: Default font CMP_FNT_BOLD: Bold CMP_FNT_REVERSE: Reverse CMP_FNT_UNDERLINE: Underline CMP_FNT_ITALIC: Italic CMP_FNT_STRIKEOUT: Strikethrough
hRatio	[IN]	Horizontal ratio [%]	1 - 1000
vRatio	[IN]	Vertical ratio [%]	1 - 1000

Description

This method is used to print text by using a font installed in the computer, which specifies alignment, font, size, style, and ratio.

What this method does internally is to generate a graphic image at the terminal based on the given parameters, to print the graphic image. Therefore, please note that it is affected by the terminal environment.

Font style can be specified in combination bold, reverse, underline, italic and strikeout. If you want to combine, please specify the logical sum. In iOS 13 or later, italic specifications are ignored.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.printTextLocalFont("Print local font text data.\n",
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_CENTER,
    withFontName: "Arial Hebrew",
    withPoint: 12,
    withStyle: ESCPOSConst.CMP_FNT_BOLD|ESCPOSConst.CMP_FNT_UNDERLINE,
    withHRatio: 100,
    withVRatio: 100)
```

2.3.11. printBitmap/printBitmapData method

Syntax

- 1) func printBitmap(fileName: String?, withAlignment alignment: Int32) -> Int32
- 2) func printBitmap(fileName: String?, withWidth width: Int32, withAlignment alignment: Int32) -> Int32
- 3) func printBitmap(fileName: String?, withWidth width: Int32, withAlignment alignment: Int32, withMode mode: Int32) -> Int32
- 4) func printBitmapData(imageData: UIImage?, withAlignment alignment: Int32) -> Int32
- 5) func printBitmapData(imageData: UIImage?, withWidth width: Int32, withAlignment alignment: Int32) -> Int32
- 6) func printBitmapData(imageData: UIImage?, withWidth width: Int32, withAlignment alignment: Int32, withMode mode: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
fileName	[IN]	Bitmap file name	
imageData	[IN]	Bitmap data	
width	[IN]	Bitmap width	CMP_BM_ASIS: Print the bitmap with one bitmap pixel per printer dot. Other Values: Bitmap width expressed in the unit of dot.
alignment	[IN]	Bitmap alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment Other Values: Distance from the left-most print column to the start of the bitmap.
mode	[IN]	Bitmap mode	CMP_BM_MODE_HT_THRESHOLD: Halftone threshold CMP_BM_MODE_HT_DITHER: Halftone dither CMP_BM_MODE_CMD_RASTER: Monochrome raster command output CMP_BM_MODE_CMD_BITIMAGE: Monochrome bitimage command output CMP_BM_MODE_CMD_GRAY16 Grayscale (4bpp) output CMP_BM_MODE_CMD_GRAY16DOWNLOAD: Grayscale (4bpp) download graphics command output

Description

This method is used to print bitmap which specifies file name and width and alignment.
Printable bitmap formats are BMP / JPG / PNG / GIF.

If the bitmap width is omitted, printing in CMP_BM_ASIS.

Mode can be specified in combination with the halftone and output method. If you want to combine, please specify the logical sum. If mode is omitted, printed at CMP_BM_MODE_HT_THRESHOLD | CMP_BM_MODE_CMD_RASTER.

For more information on mode is as follows.

Halftone Specify the halftone treatment method.

Value	Description
CMP_BM_MODE_HT_THRESHOLD	Threshold Suitable for characters printing.
CMP_BM_MODE_HT_DITHER	Dither Suitable for graphics printing.

Output Specify the output method.

Value	Description
CMP_BM_MODE_CMD_RASTER	Monochrome raster command output Suitable for small data printing. In order to output the data collectively, there is a height limit (2,304 dots 28cm approximately).
CMP_BM_MODE_CMD_BITIMAGE	Monochrome bitimage command output Suitable for large data printing. In order to output the split data, there is no height limit.
CMP_BM_MODE_CMD_GRAY16	Grayscale(4bpp) output Available in CT-D151, CT-E601/651, CT-S251/601II/651II/801II/851II/801III/851III/751. Graphic can be printed more beautifully.
CMP_BM_MODE_CMD_GRAY16DOWNLOAD	Grayscale(4bpp) download graphics command output Available in CT-D151, CT-E601/651, CT-S251/601III/651II/801III/851II/801III/851III/751. Graphic can be printed more beautifully. In order to output the data collectively, there is a 384KB limit on the size of 4bpp.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

- In the case of file designation in the resource

```
let FileName = NSBundle.mainBundle().pathForResource("sample_1.jpg",
    ofType: nil)
var errCode = escp!.printBitmap(FileName, withWidth: ESCPOSConst.CMP_BM_ASIS,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT)
```

- In the case of file designation in the images folder

```
// Get the documents folder path
let docDir = NSSearchPathForDirectoriesInDomains(.DocumentDirectory,
    .UserDomainMask, true)[0] as! String
// Get the images folder path
let image_path = docDir + "/Images"
let FileName = image_path + "/sample_1.jpg"
var errCode = escp!.printBitmap(FileName, withWidth: ESCPOSConst.CMP_BM_ASIS,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT)
```

2.3.12. setNVBitmap method

Syntax

- 1) func setNVBitmap(number: Int32, withFileName 44ilename: String?, withWidth width: Int32) -> Int32
- 2) func setNVBitmap(number: Int32, withFileName 44ilename: String?, withWidth width: Int32, withMode mode: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
number	[IN]	Number of bitmap to store in the flash memory of the printer	1 – 20
fileName	[IN]	File name of bitmap to store	
width	[IN]	Width of bitmap to store	CMP_BM_ASIS: Store the bitmap with one bitmap pixel per printer dot. Other Values: Bitmap width expressed in the unit of dot.
Mode	[IN]	Bitmap mode	CMP_BM_MODE_HT_THRESHOLD: Halftone threshold CMP_BM_MODE_HT_DITHER: Halftone dither CMP_BM_MODE_CMD_MONO Monochrome storing CMP_BM_MODE_CMD_GRAY16 Grayscale (4bpp) storing

Description

This method is used to store bitmap which specifies number, file name, width, mode as parameters. The stored bitmap can print using [printNVBitmap method](#) or [watermarkPrint method](#).

The fileName parameter sets the full path of the bitmap file to store.

The bitmap formats that can be stored are BMP / JPG / PNG / GIF.

If the width parameter is omitted, it is in CMP_BM_ASIS to store.

The mode parameter can be specified in combination with the halftone and store method. To use of the combination, please specify the logical sum. If the mode parameter is omitted, it is in CMP_BM_MODE_HT_THRESHOLD | CMP_BM_MODE_CMD_MONO to store.

For more information on the mode parameter is as follows.

Halftone Specify the halftone treatment method.

Value	Description
CMP_BM_MODE_HT_THRESHOLD	Threshold Suitable for characters printing.
CMP_BM_MODE_HT_DITHER	Dither Suitable for graphics printing.

Storing Specify the storing method.

Value	Description
CMP_BM_MODE_CMD_GRAY16	Monochrome storing
CMP_BM_MODE_CMD_GRAY16	Grayscale storing Available in CT-D151, CT-E601/651, CT-S251/601II/651II/

	801II/851II/801III/851III/751. Graphic can be stored more beautifully.
--	---

[CT-S281, CMP-20/30/20II/30II/40 Series]

It is necessary that the bitmap numbers are contiguous from number 1. If you register a new bitmap after the connection, the bitmap that was previously registered will be erased.

[CT-D101/150/151, CT-E301/351/601/651, CT-S251/310II/601/651/801/851/601II/651II/801II/851II/801III/851III/751/2000/4000/4500 Series]

It is not necessary that the bitmap numbers are contiguous. And it is possible to remove a registered image by assigning the fileName parameter as an empty string.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
let FileName = NSBundle.mainBundle().pathForResource("sample_1.jpg",
    ofType: nil)
var errCode = escp!.setNVBitmap(1, withFileName: FileName,
    withWidth: ESCPOSConst.CMP_BM_ASIS,
    withMode:
    ESCPOSConst.CMP_BM_MODE_HT_DITHER|ESCPOSConst.CMP_BM_MODE_CMD_GRAY16
)
```

2.3.13. printNVBitmap method

Syntax

```
func printNVBitmap(nvImageNumber: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
nvImageNumber	[IN]	Bitmap image number that is stored in the flash memory of the printer	1 - 20

Description

This method is used to print bitmap image (Logo) that is stored in the flash memory of the printer. To use this method, you need to register of the logo in advance. Logo registration, please store it using [setNVBitmap method](#) or use the "POS Printer utility" of utility software for the printer. Registration mode varies among the model of the printer. Please register as follows.

[CT-S281, CMP-20/30/20II/30II/40 Series]

Please register the logo with "Unused key code mode".

To the image number to use, it is necessary to register the logo sequentially.

[CT-D101/150/151, CT-E301/351/601/651, CT-S251/310II/601/651/801/851/601II/651II/801II/851II/801III/851III/751/2000/4000/4500 Series]

Please register the logo with "Key code mode".

To the image number to use, it is necessary to register the logo that specifies the key code.

The key code corresponding to the image number is as follows.

Image number	Key code (Characters)
1	"01"
2	"02"
3	"03"
.	.
.	.
.	.
19	"19"
20	"20"

Return value

Return CMP_SUCCESS (0) in success. Please refer to ["2.3.1 Return value"](#) for the error code except it.

Example

```
escp!.printNVBitmap(1)
```

2.3.14. printBarcode method

Syntax

```
func printBarcode(data: String?, withSymbology symbology: Int32, withHeight height: Int32, withWidth width: Int32, withAlignment alignment: Int32, withTextPosition textPosition: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Barcode data	
symbology	[IN]	Barcode symbol type	CMP_BCS_UPCA: UPC-A CMP_BCS_UPCE: UPC-E CMP_BCS_EAN8: EAN8 (=JAN8) CMP_BCS_JAN8: JAN8 (=EAN8) CMP_BCS_EAN13: EAN13 (=JAN13) CMP_BCS_JAN13: JAN13 (=EAN13) CMP_BCS_ITF: Interleaved 2 of 5 CMP_BCS_Codabar: Codabar CMP_BCS_Code39: Code 39 CMP_BCS_Code93: Code 93 CMP_BCS_Code128: Code 128 CMP_BCS_GS1DATABAR: GS1 DataBar Omnidirectional CMP_BCS_GS1DATABAR_E: GS1 DataBar Expanded CMP_BCS_GS1DATABAR_T: GS1 DataBar Truncated CMP_BCS_GS1DATABAR_L: GS1 DataBar Limited
height	[IN]	Barcode height (dot)	
width	[IN]	Barcode horizontal size (magnification)	2 - 6
alignment	[IN]	Barcode alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment Other Values: Distance from the left-most print column to the start of the barcode.
textPosition	[IN]	HRI characters position	CMP_HRI_TEXT_NONE: No printing CMP_HRI_TEXT_ABOVE: Above the barcode CMP_HRI_TEXT_BELOW: Below the barcode

Description

This method is used to print one-dimensional barcode.

GS1 DataBar (CMP_BCS_GS1DATABAR, CMP_BCS_GS1DATABAR_E, CMP_BCS_GS1DATABAR_T, CMP_BCS_GS1DATABAR_L) can use only the printers of CT-D101/150/151, CT-E301/351/601/651, CT-S251/310II/601/651/801/851/601II/651II/801II/851II/801III/851III/751/4500 series.

The designation of CMP_ALIGNMENT_CENTER and CMP_ALIGNMENT_RIGHT of the Barcode alignment on the page mode is ignored.

Note: The data has restriction such as characters type, the number of digits and the addition of Code Set characters. For details, please refer to the printer command reference."

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.printBarcode("123456789012",  
    withSymbology: ESCPOSConst.CMP_BCS_UPCA, withHeight: 50, withWidth: 2,  
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,  
    withTextPosition: ESCPOSConst.CMP_HRI_TEXT_ABOVE)
```


2.3.15. printPDF417 method

Syntax

```
func printPDF417(data: String?, withDigits digits: Int32, withSteps steps: Int32, withModuleWidth moduleWidth: Int32, withStepHeight stepHeight: Int32, withECLevel ECLevel: Int32, withAlignment alignment: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Barcode data	
digits	[IN]	Digits number	0: automatic 1 - 30
steps	[IN]	Steps number	0: automatic 3 - 90
moduleWidth	[IN]	Module width (dot)	2 - 8
stepHeight	[IN]	Height of step	2 - 8
ECLevel	[IN]	Error correction level	CMP_PDF417_EC_LEVEL_0: Level 0 CMP_PDF417_EC_LEVEL_1: Level 2 CMP_PDF417_EC_LEVEL_2: Level 2 CMP_PDF417_EC_LEVEL_3: Level 3 CMP_PDF417_EC_LEVEL_4: Level 4 CMP_PDF417_EC_LEVEL_5: Level 5 CMP_PDF417_EC_LEVEL_6: Level 6 CMP_PDF417_EC_LEVEL_7: Level 7 CMP_PDF417_EC_LEVEL_8: Level 8
alignment	[IN]	Barcode alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment Other Values: Distance from the left-most print column to the start of the barcode.

Description

This method is used to print PDF-417 barcode.

Please refer to the Command Reference of the printer for details on each parameter.

The designation of CMP_ALIGNMENT_CENTER and CMP_ALIGNMENT_RIGHT of the Barcode alignment on the page mode is ignored.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.printPDF417(
    "http://www.citizen-systems.co.jp/printer/tps/index.html",
    withDigits: 0,
    withSteps: 0,
    withModuleWidth: 3,
    withStepHeight: 3,
    withECLevel: ESCPOSConst.CMP_PDF417_EC_LEVEL_0,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT)
```

2.3.16. printQRCode method

Syntax

```
func printQRCode(data: String?, withModuleSize moduleSize: Int32, withECLevel ECLevel: Int32,
    withAlignment alignment: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Barcode data	
moduleSize	[IN]	Module width (dot)	1 - 16
ECLevel	[IN]	Error correction level	CMP_QRCODE_EC_LEVEL_L: Level L (7%) CMP_QRCODE_EC_LEVEL_M: Level M (15%) CMP_QRCODE_EC_LEVEL_Q: Level Q (25%) CMP_QRCODE_EC_LEVEL_H: Level H (30%)
alignment	[IN]	Barcode alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment Other Values: Distance from the left-most print column to the start of the barcode.

Description

This method is used to print QRCode barcode.

Please refer to the Command Reference of the printer for details on each parameter.

The designation of CMP_ALIGNMENT_CENTER and CMP_ALIGNMENT_RIGHT of the Barcode alignment on the page mode is ignored.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.printQRCode(
    "http://www.citizen-systems.co.jp/printer/tps/index.html",
    withModuleSize: 4,
    withECLevel: ESCPOSConst.CMP_QRCODE_EC_LEVEL_L,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT)
```

2.3.17. printGS1DataBarStacked method

Syntax

```
func printGS1DataBarStacked(data: String?, withSymbology symbology: Int32, withModuleSize
    moduleSize: Int32, withMaxWidth maxWidth: Int32, withAlignment alignment: Int32) ->
    Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Barcode data	
symbology	[IN]	Barcode symbol type	CMP_BCS_GS1DATABAR_S : GS1 DataBar Stacked CMP_BCS_GS1DATABAR_E_S : GS1 DataBar Expanded Stacked CMP_BCS_GS1DATABAR_S_O : GS1 DataBar Stacked Omnidirectional
moduleSize	[IN]	Module width (dot)	1 - 16
maxWidth	[IN]	Max width (dot)	106 - 39528 Max width of GS1 DataBar Expanded Stacked.
alignment	[IN]	Barcode alignment	CMP_ALIGNMENT_LEFT: Left alignment CMP_ALIGNMENT_CENTER: Center alignment CMP_ALIGNMENT_RIGHT: Right alignment Other Values: Distance from the left-most print column to the start of the barcode.

Description

This method is used to print 2-dimensional GS1 DataBar barcode.

This method can use only the printers of CT-D101/150/151, CT-E301/351/601/651, CT-S251/310II/601/651/801/851/601II/651II/801II/851II/801III/851III/751/4500 series.

Please refer to the Command Reference of the printer for details on each parameter.

The designation of CMP_ALIGNMENT_CENTER and CMP_ALIGNMENT_RIGHT of the Barcode alignment on the page mode is ignored.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.printGS1DataBarStacked(
    "0123456789012",
    withSymbology: ESCPOSConst.CMP_BCS_GS1DATABAR_S,
    withModuleSize: 4,
    withMaxWidth: 400,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT)
```

2.3.18. cutPaper method

Syntax

```
func cutPaper(percentage: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
type	[IN]	Cut type	CMP_CUT_FULL: Full cut CMP_CUT_PARTIAL: Partial cut CMP_CUT_FULL_PREFEED : After feed the paper to the cutting position, full cut. CMP_CUT_PARTIAL_PREFEED : After feed the paper to the cutting position, partial cut.

Description

This method is used to cut the paper.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.cutPaper(ESCPOSConst.CMP_CUT_PARTIAL_PREFEED)
```

2.3.19. unitFeed method

Syntax

```
func unitFeed(IfConunt: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
ufCount	[IN]	Number of paper feed (Dot units)	

Description

This method is used to feed the paper in dot units.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.unitFeed(200)
```

2.3.20. markFeed method

Syntax

```
func markFeed(type: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
type	[IN]	Handling type of label paper or black mark paper	CMP_MF_TO_CUTTER : After feed the paper to the auto cutter cutting position, cut further. CMP_MF_TO_NEXT_TOF : Feed the paper to the next paper's top of form.

Description

This method is used to utilize label paper and black mark paper.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.markFeed(ESCPOSConst.CMP_MF_TO_CUTTER)
```

2.3.21. openDrawer method

Syntax

```
func openDrawer(drawer: Int32, withPulseLength pulsLen: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
drawer	[IN]	Cash drawer number	CMP_DRAWER_1: Drawer 1 CMP_DRAWER_2: Drawer 2
pulseLen	[IN]	Signal length	1 - 8 Set theONtime/OFF time to the value x100ms, respectively.

Description

This method is used to open the cash drawer is connected to the printer.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp?.openDrawer(ESCPOSConst.CMP_DRAWER_1, withPulseLength: 1)
```

2.3.22. transactionPrint method

Syntax

```
func transactionPrint(control: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
control	[IN]	Transaction control	CMP_TP_TRANSACTION : Begin a transaction. CMP_TP_NORMAL : End a transaction by printing the buffered data.

Description

This method is used to start or end a transaction mode.

If control is CMP_TP_TRANSACTION, then transaction mode is entered. Subsequent methods calls will buffer the print data. The methods applied to a transaction mode are as follows.

printText, printBitmap, printNVBitmap, printBarCode, printPDF417, printQRCode, cutPaper, unitFeed, markFeed, openDrawer, rotatePrint, pageModePrint, clearePrintArea, printData, printNormal

If control is CMP_TP_NORMAL, then transaction mode is exited. If some data was buffered, then the buffered data is printed. The entire transaction is treated as one message.

Calling the clearOutput method cancels transaction mode. Any buffered print lines are also cleared.

Return value

Return CMP_SUCCESS (0) in success. Please refer to ["2.3.1 Return value"](#) for the error code except it.

Example

```
escp!.transactionPrint(ESCPOSConst.CMP_TP_TRANSACTION)
escp!.printNVBitmap(1)
escp!.printBarCode("123456789012",
    withSymbology: ESCPOSConst.CMP_BCS_UPCA,
    withHeight: 50, withWidth: 2,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withTextPosition: ESCPOSConst.CMP_HRI_TEXT_ABOVE)
escp!.printText("Line 1\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH)
escp!.printText("Line 2\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH)
escp!.printText("Line 3\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH )
escp!.printBarCode("123456789012", withSymbology: ESCPOSConst.CMP_BCS_UPCA,
    withHeight: 50, withWidth: 2,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withTextPosition: ESCPOSConst.CMP_HRI_TEXT_ABOVE)
escp!.printNVBitmap(1)
escp!.cutPaper(ESCPOSConst.CMP_CUT_PARTIAL_PREFEED)
escp!.transactionPrint(ESCPOSConst.CMP_TP_NORMAL)
```


2.3.23. rotatePrint method

Syntax

```
func rotatePrint(rotation: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
rotation	[IN]	Direction of rotation	CMP_RP_ROTATE180: Start rotated printing 180°, that is, print upside-down CMP_RP_BARCODE : Start rotated bar code printing. This value is ORed with the above start rotated print values. CMP_RP_BITMAP : Start rotated bitmap printing. This value is ORed with the above start rotated print values. CMP_RP_NORMAL : End rotated printing

Description

This method is used to start or end a rotation print mode.

If rotation includes PTR_RP_ROTATE180, then upside-down print mode is entered. The methods applied to a rotation print mode are as follows.

printText、printNormal

If rotation includes PTR_RP_BARCODE and/or PTR_RP_BITMAP, the following methods are printed also rotated.

printBarcod, printPDF417, printQRCode and/or printBitmap

If rotation is CMP_RP_NORMAL, then rotation mode is exited.

Return value

Return CMP_SUCCESS (0) in success. Please refer to ["2.3.1 Return value"](#) for the error code except it.

Example

```
escp!.rotatePrint(ESCPOSConst.CMP_RP_ROTATE180|ESCPOSConst.CMP_RP_BARCODE|
    ESCPOSConst.CMP_RP_BITMAP)
escp!.printBitmap("samplebitmap.bmp", withWidth: ESCPOSConst.CMP_BM_ASIS,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_CENTER)
escp!.printBarCode("123456789012", withSymbology: ESCPOSConst.CMP_BCS_UPCA,
    withHeight: 50, withWidth: 2,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withTextPosition: ESCPOSConst.CMP_HRI_TEXT_ABOVE)
escp!.printText("Line 1\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH)
escp!.printText("Line 2\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH)
escp!.printText("Line 3\n", withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize: ESCPOSConst.CMP_TXT_1WIDTH)
escp!.cutPaper(ESCPOSConst.CMP_CUT_PARTIAL_PREFEED)
escp!.rotatePrint(ESCPOSConst.CMP_RP_NORMAL)
```

2.3.24. pageModePrint method

Syntax

```
func pageModePrint(control: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
control	[IN]	Page Mode control	CMP_PM_PAGE_MODE: Enter Page Mode CMP_PM_PRINT_SAVE: Print PageModePrintArea and save the canvas CMP_PM_NORMAL: Print the print area and destroy the canvas and exit Page Mode. CMP_PM_CANCEL: Clear the page and exit the Page Mode without any printing of any print area

Description

This method is used to start or end a Page Mode.

If control is PTR_PM_PAGE_MODE, then Page Mode is entered. Subsequent methods calls will buffer the print data. The methods applied to a Page Mode are as follows.

PrintText, PrintBitmap, PrintBarCode, printPDF417, printQRCode, PrintNormal

If control is PTR_PM_PRINT_SAVE, then Page Mode is not exited. If some data is buffered, then the buffered data is saved and printed. This control is used to print the same page layout with additional print items inside of the page.

If control is PTR_PM_NORMAL, then Page Mode is exited. If some data is buffered, then the buffered data is printed. The buffered data will not be saved.

If control is PTR_PM_CANCEL, then Page Mode is exited. If some data is buffered, then the buffered data is not printed and is not saved.

Note that when the pageModePrint method is called, all of the data that is to be printed in the PageModePrintArea will be printed and the paper is fed to the end of the PageModePrintArea. If more than one PageModePrintArea is defined, then after the pageModePrint method is called, all of the data that is to be printed in the respective PageModePrintArea(s) will be printed and the paper will be fed to the end of the PageModePrintArea located the farthest "down" the sheet of paper.

The entire Page Mode transaction is treated as one message.

Calling the clearOutput method cancels Page Mode. Any buffered print lines are also cleared.

Return value

Return CMP_SUCCESS (0) in success. Please refer to ["2.3.1 Return value"](#) for the error code except it.

Example

```
// Standard print
var sample2 = String(format: "%c|2vCSample 2 - Print\n", 27)
escp!.printNormal(sample2)
escp!.printText("1234567890123456789012345678901234567890123456789012345678901234567890123456789012345 67890123456789012345678901234567890\n",
```

```

        withAlignment: ESCPOSConst.CMP_ALIGNMENT_RIGHT,
        withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
        withTextSize:
            ESCPOSConst.CMP_TXT_1WIDTH|ESCPOSConst.CMP_TXT_1HEIGHT)
// Start of Page Mode
escp!.pageModePrint(ESCPOSConst.CMP_PM_PAGE_MODE)
// Set offset of Page Mode
escp!.setPageModeVerticalPosition(0)
escp!.setPageModeHorizontalPosition(0)
// Set direction of Page Mode
escp!.setPageModePrintDirection(ESCPOSConst.CMP_PD_TOP_TO_BOTTOM)
// Set print area of Page Mode
escp!.setPageModePrintArea("308,0,76,800")
var receipt = String(format: "%c|4C- Receipt -\n", 27)
escp!.printNormal(receipt)
// Set print area of Page Mode
escp!.setPageModePrintArea("184,0,120,800")
escp!.printText(" $ 299.99- \n",
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_CENTER,
    withAttribute:
        ESCPOSConst.CMP_FNT_UNDERLINE|ESCPOSConst.CMP_FNT_BOLD,
    withTextSize:
        ESCPOSConst.CMP_TXT_4WIDTH|ESCPOSConst.CMP_TXT_4HEIGHT)
// Set print area of Page Mode
escp!.setPageModePrintArea("88,0,88,560")
escp!.printText("CITIZEN SYSTEMS\n",
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_RIGHT,
    withAttribute: ESCPOSConst.CMP_FNT_DEFAULT,
    withTextSize:
        ESCPOSConst.CMP_TXT_2WIDTH|ESCPOSConst.CMP_TXT_3HEIGHT)
// Set print area of Page Mode
escp!.setPageModePrintArea("0,0,98,480")
escp!.printBarCode("123456789012", withSymbology: ESCPOSConst.CMP_BCS_UPCA,
    withHeight: 64, withWidth: 4,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT,
    withTextPosition: ESCPOSConst.CMP_HRI_TEXT_BELOW)
// Set print area of Page Mode
escp!.setPageModePrintArea("0,600,192,192")
escp!.printQRCode("http://www.citizen-systems.co.jp/", withModuleSize: 5,
    withECLevel: ESCPOSConst.CMP_QRCODE_EC_LEVEL_L,
    withAlignment: ESCPOSConst.CMP_ALIGNMENT_LEFT)
// End of Page Mode
escp!.pageModePrint(ESCPOSConst.CMP_PM_NORMAL)

```

Print image



2.3.25. clearPrintArea method

Syntax

```
func clearPrintArea() -> Int32
```

Parameter

Not exist.

Description

This method is used to clear the area defined by the PageModePrintArea property.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.clearPrintArea()
```

2.3.26. clearOutput method

Syntax

```
func clearOutput() -> Int32
```

Parameter

Not exist.

Description

This method is used to clear all buffered output data by tranzactionPrint and pageModePrint method. Also, when possible, halts outputs that are in progress. At the same time, the command to clear print data on the printer is sent.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.clearOutput()
```

2.3.27. printData method

Syntax

```
func printData(data: UnsafeMutablePointer<Int8>, withLength length: UInt) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
data	[IN]	Send data	
length	[IN]	Send data size	

Description

This method is used to send data bytes to the printer directly.

It is usually not necessary, please use if you want to send ESC commands directly to the printer.

If you want to use, please be careful so as not to affect the other methods.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
// Sound the buzzer (The printer must support buzzer.)
var data: [Int8] = [0x1b, 0x1e]
res = escp!.printData(&data, withLength: 2)
```

2.3.28. printNormal method

Syntax

```
func printNormal(data: String?) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
Data	[IN]	Print data (Support OPOS escape sequence)	

Description

This method is used to print using the escape sequences that are defined in the OPOS.

Please use this if you are familiar with the OPOS.

The supporting escape sequences in this SDK are as follows.

Please refer to specifications of OPOS for the details.

Escape Sequence		Notes
Paper cut	ESC #P	Partial cut (1-99), Full cut (0,100)
Feed and paper cut	ESC #fP	Partial cut (1-99), Full cut (0,100)
Bitmap print	ESC #B	1-20 (Bitmap image number that is stored in the flash memory of the printer) After Bitmap printing, print position returns to the initial state (left-justified).
Multi-line feed	ESC #IF	
Unit feed	ESC #uF	
Barcode print	ESC #R	
Font type specification	ESC #fT	
Bold	ESC bC	
Underline	ESC #uC	
Custom color	ESC #rC	Effective only when dedicated 2-color paper is used.
Red	ESC rC	Effective only when dedicated 2-color paper is used.
Reverse character	ESC rvC	
Standard	ESC 1C	
Double width	ESC 2C	
Double height	ESC 3C	
Quadruple	ESC 4C	
Horizontal magnification	ESC #hC	1-8
Vertical magnification	ESC #vC	1-8
Centering	ESC cA	
Right adjustment	ESC rA	
Normal	ESC N	

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
var receipt = String(format: "%c|2vC Receipt -\n", 27)
escp!.printNormal(receipt)
```

2.3.29. watermarkPrint method

Syntax

```
func watermarkPrint(start: Int32, withNVImageNumber nvImageNumber: Int32, withPass pass: Int32,
    withFeed feed: Int32, withRepeat repeat: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
start	[IN]	The start / Stop of the watermark print	CMP_WM_START: The start of the watermark print CMP_WM_STOP: The stop of the watermark print
nvImageNumber	[IN]	The NV image number that is stored in the flash memory of the printer	1 - 20
pass	[IN]	The first start position (vertical direction) of the watermark	0 - 65,535 [dot]
feed	[IN]	The blank length each watermark	0 - 65,535 [dot]
repeat	[IN]	The print number of times of the watermark	0: Infinite repetition 1 - 65,535: The repetition number of times

Description

This method is used to print watermark.

This is available with a printer of the CT-D151, CT-E601/651, CT-S251/601II/651II/801II/851II/751 series.

The bitmap image stored in the flash memory of the printer is printed out as watermark.

It is necessary to store the image beforehand to use this method.

About logo registration method, please store it using [setNVBitmap method](#) or use the "POS Printer utility" of utility software for the printer.

When the printing of watermark was stopped in CMP_WM_STOP, all other arguments are ignored

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
escp!.watermarkPrint(ESCPOSConst.CMP_WM_START, withNVImageNumber: 1,
    withPass: 0, withFeed: 0, withRepeat: 0)
```


2.3.30. searchCitizenPrinter method

Syntax

```
class func searchCitizenPrinter(ifType: Int32, withSearchTime searchTime: Int32, result:
    UnsafeMutablePointer<Int32>) -> [AnyObject]?
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
ifType	[IN]	Connect type	CMP_PORT_WiFi: Wi-Fi connection CMP_PORT_BLUETOOTH: Bluetooth connection CMP_PORT_USB: USB connection
searchTime	[IN]	Search time (sec)	1 - 30
result	[OUT]	Error code	---

Return CMP_SUCCESS (0) to result in success. Please check the description of the error codes below in the case of failure. Please refer to ["2.3.1 Return value"](#) for the error code except it.

Error codes	Description
CMP_E_ILLEGAL (1101)	Invalid parameter. (1) The connect type is unsupported. (2) The search time is out of range.
CMP_E_FAILURE (1104)	Error occurs during the search, the search failed.
CMP_E_NO_LIST (1106)	As a result of search, the printer cannot be found.

Description

This method is used to search the printer and to obtain the list of the printer information. Please specify the type of the printer connection and the search time. Only WiFi connection is available at the time of the simulator use. After search time passed, set a result to the result parameter and return the information of the found printers as NSArray type.

In the case of CMP_PORT_WiFi for the connection type, you can search only the printers of CT-D101/150/151, CT-E301/351/601/651, CT-S251/310II/601/651/801/851/601II/651II/801II/851II/801III/851III/751/2000/

4000/4500 series. Recommended value of search time is more than 3 seconds. When the search time is shorter than the second, a search may fail by the network situation.

In the case of CMP_PORT_BLUETOOTH for the connection type, CT-D151, CT-E601/651, CT-S251/281/601II/651II/801II/851II/751/4500 can be searched. It does not depend on the setting of searchTime, and the search is finished immediately.

In the case of CMP_PORT_USB for the connection type, CT-D151, CT-E601/651, CT-S251/751/4500 can be searched. It does not depend on the setting of searchTime, and the search is finished immediately.

Return value

A list of information of the searched printer is given back on success. An empty list is returned on failure. The list of information of the printer is stored as a CitizenPrinterInfo-type, and available information varies according to ifType parameter.

ifType	CitizenPrinterInfo	Information to be obtained
CMP_PORT_WiFi	ipAddress	IP Address
	macAddress	MAC Address
	deviceName	(Empty character)

	bdAddress	(Empty character)
	usbSerialNo	(Empty character)
CMP_PORT_BLUETOOTH	ipAddress	(Empty character)
	macAddress	(Empty character)
	deviceName	(Empty character) / Bluetooth device name *depend on the printer model
	bdAddress	(Empty character) / BD Address * in the case of iOS6 or more use
	usbSerialNo	(Empty character)
CMP_PORT_USB	ipAddress	(Empty character)
	macAddress	(Empty character)
	deviceName	Model name
	bdAddress	(Empty character)
	usbSerialNo	USB serial number

Example

```

var result = Int32(0)
let array = ESCPOSPrinter.searchCitizenPrinter(ESCPOSConst.CMP_PORT_WiFi,
    withSearchTime: 4, result: &result)!
for var i = 0; i < array.count; i++ {
    let prninfo = array[i] as? CitizenPrinterInfo
    println("IP Address : \(prninfo.ipAddress)")
    println("MAC Address : \(prninfo.macAddress)")
}

```

2.3.31. searchESCPOSPrinter method

Syntax

```
class func searchESCPOSPrinter(ifType: Int32, withSearchTime searchTime: Int32, result:
    UnsafeMutablePointer<Int32>) -> [AnyObject]?
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
ifType	[IN]	Connect type	CMP_PORT_WiFi: Wi-Fi connection CMP_PORT_BLUETOOTH: Bluetooth connection CMP_PORT_USB: USB connection
searchTime	[IN]	Search time (sec)	1 - 30
result	[OUT]	Error code	---

Return CMP_SUCCESS (0) to result in success. Please check the description of the error codes below in the case of failure. Please refer to ["2.3.1 Return value"](#) for the error code except it.

Error codes	Description
CMP_E_ILLEGAL (1101)	Invalid parameter. (1) The connect type is unsupported. (2) The search time is out of range.
CMP_E_FAILURE (1104)	Error occurs during the search, the search failed.
CMP_E_NO_LIST (1106)	As a result of search, the printer cannot be found.

Description

This method is used to search the printer and to obtain the list of the addresses. Please specify the type of the printer connection and the search time. Only WiFi connection is available at the time of the simulator use. After search time passed, set a result to the result parameter and return the information of the found printers as NSArray type.

In the case of CMP_PORT_WiFi for the connection type, you can search only the printers of CT-D101/150/151, CT-E301/351/601/651, CT-S251/310II/601/651/801/851/601II/651II/801II/851II/801III/851III/751/2000/4000/4500 series. Recommended value of search time is more than 3 seconds. When the search time is shorter than the second, a search may fail by the network situation.

In the case of CMP_PORT_BLUETOOTH for the connection type, CT-D151, CT-E601/651, CT-S251/281/601II/651II/801II/851II/751/4500 can be searched. It does not depend on the setting of searchTime, and the search is finished immediately.

In the case of CMP_PORT_USB for the connection type, CT-D151, CT-E601/651, CT-S251/751/4500 can be searched. It does not depend on the setting of searchTime, and the search is finished immediately.

Return value

In the case of CMP_PORT_WiFi for the connection type, return the list of IP addresses of the printers when a search succeeded. When a search fails, return the empty list.

In the case of CMP_PORT_BLUETOOTH for the connection type and iOS6 or more, return the list of Bluetooth device addresses of the printers when a search succeeded. When a search fails, return the empty list. Because it does not correspond to information on the Bluetooth device address for less than iOS6, the empty list is returned.

In the case of CMP_PORT_USB for the connection type, return the list of USB serial number of the printers when a search succeeded. When a search fails, return the empty list.

Example

```
var result = Int32(0)
let array = ESCPOSPrinter.searchESCPOSPrinter(ESCPOSConst.CMP_PORT_WiFi,
    withSearchTime: 4, result: &result)!
```

2.3.32. printerCheckEx method

Syntax

- 1) func printerCheckEx(status: UnsafeMutablePointer<Int32>, withConnectType connectType: Int32, withAddress addr: String?) -> Int32
- 2) func printerCheckEx (status: UnsafeMutablePointer<Int32>, withConnectType connectType: Int32, withAddress addr: String?, withPort port: Int32) -> Int32
- 3) func printerCheckEx (status: UnsafeMutablePointer<Int32>, connectType: Int32, withAddress addr: String?, withPort port: Int32, withTimeout timeout: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
status	[OUT]	Status codes	
connectType	[IN]	Connect type	CMP_PORT_WiFi: Wi-Fi connection CMP_PORT_BLUETOOTH: Bluetooth connection CMP_PORT_USB: USB connection (Lightning/Type-C) CMP_PORT_SNMP: SNMP connection (Network Only)
addr	[IN]	IP address to connect or Bluetooth device address or Bluetooth device name or serial number	WiFi/SNMP: 0.0.0.0 - 255.255.255.255 Bluetooth: 00:00:00:00:00:00 – FF:FF:FF:FF:FF:FF Device name USB: Blank or serial number
port	[IN]	Connection port number	
timeout	[IN]	Timeout (msec)	

Description

This method is used to connect printer and get the status of the printer. After the process is complete, disconnect the connection. (except connection type CMP_PORT_SNMP)

The CMP_PORT_SNMP in the connect type can be used with printers connected to the network. By using this connection type, you can get the status regardless of other connections. In order to use this connection type, the printer supported with this function.

If the acquisition of the status is successful, set the following status code to the status parameters in the logical sum.

Status codes	Description
CMP_STS_NORMAL (0)	The printer is normal.
CMP_STS_PRINTEROFF (128)	The printer is off-line.
CMP_STS_PAPER_EMPTY (32)	The printer is out of paper.
CMP_STS_COVER_OPEN (16)	The cover of the printer opens.
CMP_STS_PAPER_NEAREMPTY (4)	Paper near empty.
CMP_STS_DRAWER_LEVEL_H (2)	Status of pin 3 of drawer kick-out connector = H.

Return value

Return CMP_SUCCESS (0) in success. Please check the description of the error codes below in the case of failure. Please refer to "[2.4.1 Return value](#)" for the error code except it.

Error codes	Description
CMP_E_NOTCONNECT (1003)	Failed connection to the printer. (1) The printer is under none-connection status. (2) The printer is not turned ON. (3) Cannot obtain handle of interface board. (4) The printer is in use by another connection. (Except SNMP)
CMP_E_CONNECT_NOTFOUND (1004)	Failed to check the support model after connecting to the printer. (1) The model is not supported.

Example

```

var status = Int32(0)
var result = escp!.printerCheckEx(&status,
                                   withConnectType: ESCPOSConst.CMP_PORT_WiFi,
                                   withAddress: "192.168.182.100")
if ESCPOSConst.CMP_SUCCESS == result {
    if ESCPOSConst.CMP_STS_NORMAL == status {
        // No Error
    } else {
        if (ESCPOSConst.CMP_STS_COVER_OPEN & status) > 0 {
            // Cover Open
        }
        if (ESCPOSConst.CMP_STS_PAPER_EMPTY & status) > 0 {
            // Paper Empty
        }
        if (ESCPOSConst.CMP_STS_PRINTEROFF & status) > 0 {
            // Printer Offline
        }
        if (ESCPOSConst.CMP_STS_PAPER_NEAREMPTY & status) > 0 {
            // Paper Near Empty
        }
        if (ESCPOSConst.CMP_STS_DRAWER_LEVEL_H & status) > 0 {
            // Pin 3 of drawer kick-out connector = H
        }
    }
} else {
    // printerCheckEx Error
}

```

2.3.33. openDrawerEx method

Syntax

- 1) func openDrawerEx(drawer: Int32, withPulseLength pulsLen: Int32, connectType: Int32, withAddress addr: String?) -> Int32
- 2) func openDrawerEx(drawer: Int32, withPulseLength pulsLen: Int32, connectType: Int32, withAddress addr: String?, withPort port: Int32) -> Int32
- 3) func openDrawerEx(drawer: Int32, withPulseLength pulsLen: Int32, connectType: Int32, withAddress addr: String?, withPort port: Int32, withTimeout timeout: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
drawer	[IN]	Cash drawer number	CMP_DRAWER_1: Drawer 1 CMP_DRAWER_2: Drawer 2
pulseLen	[IN]	Signal length	1 - 8 (x 100) msec
connectType	[IN]	Connect type	CMP_PORT_WiFi: Wi-Fi connection CMP_PORT_BLUETOOTH: Bluetooth connection CMP_PORT_USB: USB connection (Lightning/Type-C)
addr	[IN]	IP address to connect or Bluetooth device address or Bluetooth device name or serial number	WiFi: 0.0.0.0 - 255.255.255.255 Bluetooth: 00:00:00:00:00:00 – FF:FF:FF:FF:FF:FF Device name USB: Blank or serial number
port	[IN]	Connection port number	
timeout	[IN]	Timeout (msec)	

Description

This method is used to connect printer and open the cash drawer is connected to the printer. After the process is complete, disconnect the connection.

This method can execute even if the printer error (cover open or paper empty).

Return value

Return CMP_SUCCESS (0) in success. Please check the description of the error codes below in the case of failure. Please refer to "[2.4.1 Return value](#)" for the error code except it.

Error codes	Description
CMP_E_NOTCONNECT (1003)	Failed connection to the printer. (1) The printer is under none-connection status. (2) The printer is not turned ON. (3) Cannot obtain handle of interface board. (4) The printer is in use by another connection. (Except SNMP)
CMP_E_CONNECT_NOTFOUND (1004)	Failed to check the support model after connecting to the printer. (1) The model is not supported.

Example

```
escpl.openDrawerEx(ESCPOSConst.CMP_DRAWER_1, withPulseLength: 1,
    withConnectType: ESCPOSConst.CMP_PORT_WiFi,
    withAddress: "192.168.182.100")
```

2.3.34. setPrintCompletedTimeout method

Syntax

```
func setPrintCompletedTimeout(timeout: Int32) -> Int32
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
timeout	[IN]	Timeout of print completion notification (msec)	0: Automatically adjusts the timeout. Other Values: Specify the timeout. Expressed in milliseconds.

Description

This method is used to set the timeout to check the print completion notification.

When you create an instance, the timeout is initialized to 0.

Please refer to "[2.4.1. Function to detect the completion of printing](#)" for details of the function to detect the completion of printing.

Return value

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Example

```
// Automatically adjusts
escp!.setPrintCompletedTimeout(0)

// Fixed 90sec
escp!.setPrintCompletedTimeout(90000)
```


2.3.35. setLog method

Syntax

```
func setLog(mode: Int32, withPath path: String?, withMaxSize maxSize: Int32)
```

Parameters

Parameter	[IN/OUT]	Description	Setting range
mode	[IN]	Logging mode	0: None 1: Access logs 2: Error logs
path	[IN]	File path to store	The folder in the application sharing folder.
maxSize	[IN]	Maximum Log Size	0: Unlimited 1 - : Maximum size (MB)

Description

Sets the logging function. See "[2.4.3 Logging function](#)" for more details.

Return value

none

Example

```
escp!.setLog(1, withPath: "/", withMaxSize: 10)
```

2.3.36. getVersionCode method

Syntax

```
func getVersionCode() -> Int32
```

Parameter

Not exist.

Description

This method is used to get a numerical value for the version number of this SDK.

Return value

Return a numerical value for the version number of this SDK. (Ver1.00 is 100)

Example

```
escp!.getVersionCode()
```

2.3.37. getVersionName method

Syntax

```
func getVersionName() -> String?
```

Parameter

Not exist.

Description

This method is used to get a string for the version number of this SDK.

Return value

Return a string for the version number of this SDK. (Ver1.00 is "1.00")

Example

```
escp!.getVersionName()
```

2.3.38. PageModeArea property

Type

String

Attribute

Read only

Description

This property holds the page area expressed in the unit of Dot. The string consists of two ASCII numbers separated by a comma, in the following order: horizontal size, vertical size.

This page area is determined by the hardware capability of the printer.

[CT-S251 Series] : "432,1662"

[CT-S281 Series] : "384,938"

[CT-D101/150/151, CT-E301/351/601/651, CT-S310II/601/651/801/851/601II/651II/801II/851II/801III/851III/2000/751 Series] :

"576,1662"

[CT-S4000/4500 Series] : "832,1662"

[CMP-20/20II Series] : "384,938"

[CMP-30/30II Series] : "576,938"

[CMP-40 Series] : "832,938"

For example, if the string is "384,938", then the page size is 384 horizontal units by 938 vertical units, and the station print area is a rectangle beginning at the top left point (0,0), and continuing up to the bottom right point (383,937).

The connect method must be complete before accessing this property. This property is set in connect method.

Set property

Not exist.

Get property

```
func getPageModeArea() -> String?
```

Returns the page area as the return value.

2.3.39. PageModePrintArea property

Type

String

Attribute

Read/Write

Description

This property holds the print area of Page Mode, expressed in the unit of dot. The maximum print area is the page area.

The string consists of four ASCII numbers separated by commas, in the following order: horizontal start, vertical start, horizontal size, vertical size.

Text written to the right edge of the print area will wrap to the next line. Any text or image written beyond the bottom of the print area will be truncated.

For example, if the string is "50,100,200,400", then the station print area is a rectangle beginning at the point (50,100), and continuing up to the point (249,499).

The connect method must be complete before accessing this property. This property is initialized to "0,0,0,0" at connect method.

Set property

```
func setPageModePrintArea(area: String?) -> Int32
```

Please specify the property value that you want to set in the parameter.

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Get property

```
func getPageModePrintArea() -> String?
```

Returns the Page Mode print area that is set as the return value.

2.3.40. PageModePrintDirection property

Type

Int32

Attribute

Read/Write

Description

This property holds the print direction of the Page Mode print area. The print direction values are as follows.

Value	Meaning
CMP_PD_LEFT_TO_RIGHT	Print left to right, starting at top left position of the print area, i.e., normal printing.
CMP_PD_BOTTOM_TO_TOP	Print bottom to top, starting at the bottom left position of the print area, i.e., rotated left 90° printing.
CMP_PD_RIGHT_TO_LEFT	Print right to left, starting at the bottom right position of the print area, i.e., upside down printing.
CMP_PD_TOP_TO_BOTTOM	Print top to bottom, starting at the top right position of the print area, i.e., rotated right 90° printing.

Setting this property may also change PageModeHorizontalPosition and PageModeVerticalPosition. Setting this property will have an effect on the current print area. By changing the print area, it is possible to generate a receipt or slip with text printed in multiple rotations.

The connect method must be complete before accessing this property. This property is initialized to CMP_PD_LEFT_TO_RIGHT at connect method.

Set property

```
func setPageModePrintDirection(direction: Int32) -> Int32
```

Please specify the property value that you want to set in the parameter.

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Get property

```
func getPageModePrintDirection() -> Int32
```

Returns the print direction of Page Mode print area that is set as the return value.

2.3.41. PageModeHorizontalPosition property

Type

Int32

Attribute

Read/Write

Description

This property holds the horizontal start position offset within the Page Mode print area, expressed in the unit of dot.

The horizontal direction is the same as the actual PageModePrintDirection property.

A read/get on this property will return the horizontal position offset set by the last write/set and not the current position.

The connect method must be complete before accessing this property. This property is initialized to zero (0) at connect method.

Set property

```
func setPageModeHorizontalPosition(position: Int32) -> Int32
```

Please specify the property value that you want to set in the parameter.

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Get property

```
func getPageModeHorizontalPosition() -> Int32
```

Returns the horizontal position of Page Mode print area that is set as the return value.

2.3.42. PageModeVerticalPosition property

Type

Int32

Attribute

Read/Write

Description

This property holds the vertical start position offset within the Page Mode print area, expressed in the unit of dot.

The vertical direction is perpendicular to the direction specified in the actual PageModePrintDirection property.

A read/get on this property will return the vertical position offset set by the last write/set and not the current position.

The connect method must be complete before accessing this property. This property is initialized to zero (0) at connect method.

Set property

```
func setPageModeVerticalPosition(position: Int32) -> Int32
```

Please specify the property value that you want to set in the parameter.

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Get property

```
func getPageModeVerticalPosition() -> Int32
```

Returns the vertical position of Page Mode print area that is set as the return value.

2.3.43. RecLineSpacing property

Type

Int32

Attribute

Read/Write

Description

This property holds the spacing of each single-high print line, including both the printed line height plus the whitespace between each pair of lines, expressed in the unit of dot.

Depending upon the current line spacing, a multi-high print line might exceed this value. In this case the whitespace is zero.

The connect method must be complete before accessing this property. This property is initialized to 34 at connect method.

Set property

```
func setRecLineSpacing(spacing: Int32) -> Int32
```

Please specify the property value that you want to set in the parameter.

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Get property

```
func getRecLineSpacing() -> Int32
```

Returns the spacing of each single-high print line that is set as the return value.

2.3.44. MapMode property

Syntax

Int32

Attribute

Read/Write

Description

This property holds the mapping mode of the printer. The mapping mode defines the unit of measure used for other properties, such as line heights and line spacing. The map mode values are as follows.

Value	Meaning
CMP_MM_DOTS	The printer's dot width.
CMP_MM_TWIPS	1/1440 of an inch.
CMP_MM_ENGLISH	0.001 inch.
CMP_MM_METRIC	0.01 millimeter.

The method and the properties to be affected by the MapMode property are as follows.

[printBitmap method](#) : width, alignment
[printBitmapData method](#) : width, alignment
[setNVBitmap method](#) : width
[printBarcode method](#) : height, width, alignment
[printPDF417 method](#) : moduleWidth, alignment
[printQRCode method](#) : moduleSize, alignment)
[printGS1DataBarStacked method](#) : moduleSize, maxSize, alignment
[unitFeed method](#) : ufCount
[watermarkPrint method](#) : pass, feed
[PageModeArea property](#)
[PageModePrintArea property](#)
[PageModeHorizontalPosition property](#)
[PageModeVerticalPosition property](#)
[RecLineSpacing property](#)

The connect method must be complete before accessing this property. This property is initialized to CMP_MM_DOTS at connect method.

Set property

```
func setMapMode (number: Int32)->Int32
```

Please specify the property value that you want to set in the parameter.

Return CMP_SUCCESS (0) in success. Please refer to "[2.3.1 Return value](#)" for the error code except it.

Get property

```
func getMapMode ()->Int32
```

Returns the mapping mode that is set as the return value.

2.3.45. ErrorCodeExtended property

Syntax

Int32

Attribute

Read only

Description

This property holds the extended error code when there is an error connecting to the printer. The values are as follows.

Value	Meaning
CMP_EX_DEV_NO_PRINTER (62000)	Failed connection to the printer. (1) The printer is under none-connection status. (2) The printer is not turned ON. (3) Cannot obtain handle of interface board.
CMP_EX_DEV_NO_SERIALNO (62012)	Printer serial number does not match. (USB connection only) (1) Wrong specified serial number.
CMP_EX_DEV_OPEN_ERROR (62100)	Unable to connect socket or stream. (1) Communication error. (2) Wrong address specification when connecting to WiFi.
CMP_EX_DEV_NO_RESPONSE (62102)	No command response from the printer. (1) The printer is in use by another application.

This property is initialized to 0 at the beginning of the connection process. Refer to this after executing the [connect method](#), [printerCheckEx method](#), and [openDrawerEx method](#).

Set property

Not exist.

Get property

```
func getResultCodeExtended() -> Int32
```

Returns the extended error code as the return value.

2.4. Notes

Notes of this SDK are as follows.

2.4.1. Function to detect the completion of printing

In this library, after the printing output, the SDK waits for the printing completion reply from a printer and judge the success / failure of the method.

The function to detect the completion of printing is processed in the following cases.

- (1) At the time of completion of transaction processing (transactionPrint method)
- (2) At the time of completion of page mode (pageModePrint method)
- (3) At the time of data output of the methods except during the buffering process in transaction or page mode

The function to detect the completion of printing need a few time to wait for the response of the printer. If you want to print continuously multiple methods, smooth printing is possible by using transaction processing. (transactionPrint method)

The timeout for checking the print completion notification is automatically adjusted according to the print data. Depending on the print data, the timeout error may occur each time. In that case, set the timeout with the [setPrintCompletedTimeout method](#) according to the actual print time.

2.4.2. About printing UTF-8 encode characters

This SDK supports printing UTF-8 encoded characters.

This feature is focusing on providing a way to interoperate East Asian legacy double-byte character sets for Japanese, Korean, Simplified and Traditional Chinese.

Example

```
printer.setEncoding(String.Encoding.utf8);
```

Supported models

Model	Firmware Version	Conditions
CT-S251	EM01-0304 or newer	*1
CT-S310II	DT00-1000 or newer DT10-1100 or newer	
CT-S601II	EE00-0200 or newer	*2
CT-S651II	EA00-0200 or newer	
CT-S801II	ED00-0200 or newer	
CT-S851II	DY00-0200 or newer	
CT-D101/150/151	All versions	*3
CT-E301/351/601/651		
CT-S751/801III/851III/4500		

Note

- *1 These models don't support interoperating East Asian legacy double-byte character sets for Japanese, Korean, Simplified and Traditional Chinese. The available language for printing is depending on the region where the printer unit was purchased.
- *2 These models don't support interoperating East Asian legacy double-byte character sets for Japanese, Korean, Simplified and Traditional Chinese. The available language for printing is depending on the encoding selected for the MSW9-4.
- *3 These models support interoperating East Asian legacy double-byte character sets for Japanese, Korean, Simplified and Traditional Chinese. The printer picks up available characters one by one based on the language assigned for the MSW9-4 selection. Please note that this may result in an inconsistency of the font typeface.

Language and typeface (CT-D150/151, CT-E351/651, CT-S801III/851III Series)

Language	Typeface
Japanese Korean	"Gothic" (Sans-serif)
Simplified Chinese Traditional Chinese	"Mincho" (Serif)

Language and typeface (CT-D101, CT-E301/601, CT-S751/4500 Series)

Language	Typeface
Japanese Korean Simplified Chinese Traditional Chinese	"Gothic" (Sans-serif)

2.4.3. Logging function

This SDK has a logging function which keeps history of executing methods or reading/writing properties. This can be enabled by using the [setLog method](#), or placing a file "CSJPOSLib.cfg" in the same folder as the SDK.

< Example of CSJPOSLib.cfg >

```
[LogSetting]      ...Section name (Fixed)
LogMode=1         ...Specifies the log mode.
LogPath=/         ...Specifies the folder in the application sharing folder to store the log files.
LogMaxSize=10     ...Specifies the maximum size of log file in MB.
```

Setting items

- LogMode

Specifies a log mode:

- 0: None
- 1: Access log
- 2: Error log

- LogPath

Specifies a folder to store the log files. "/" of the application sharing folder by default.

- LogMaxSize

Specifies maximum size to log file in MB. "0" sets the size to "unlimited."

Log file name

Log files will be stored with a file name "CSJPOSLib_" and a number which indicates the day of week(0 to 6. 0: Sunday, 1: Monday...), and a file extension ".log."

Example: CSJPOSLib_1.log

When the same file name exists, the file will be overwritten if the file is one week older, new logs will be added to the existing file if the file is created on the same day.

Log format

The log file keeps the information of executed methods, accessed properties, timestamps and results.

```
--- Example 1, method (connect) ---
2022/04/06 11:54:13.548 001 METHOD call   connect(0, 192.168.234.100, 9100, 4000)
2022/04/06 11:54:13.962 001 METHOD result connect() -> Success(0)

--- Example 2, method (printText) ---
2022/04/06 11:54:16.672 001 METHOD call   printText([See below], 1, 0, 0)
-----Parameter Detail-----
Print text 1
Print text 2
-----
2022/04/06 11:54:16.948 001 METHOD result printText() -> Success(0)

--- Example 3, set to a property (RecLineSpacing) ---
2022/04/06 11:54:16.950 001 PROPERTY set  RecLineSpacing <- 24 : Success(0)
```

```
--- Example 4, get from a property (RecLineSpacing) ---
```

```
2022/04/06 11:54:16.954 001 PROPERTY get RecLineSpacing -> 24
```

- * The logging function could be a "bottleneck" of processing since it tries to keep the information of every single execution of a method or access to a property.
- * Logging could fail without a notification for the reason below.
 - A write-protected device is selected as a storage location.
 - No enough space in the selected storage device.
 - The storage location contains a file with write-protection.
 - No access privilege to a file or folder.
 - Another program is using the log file.

2.4.4. Predefined Constants List

No	Type	Name	Data type	Value	Description
1	Result/Error	CMP_SUCCESS	Int32	0	Successfully completed
		CMP_E_CONNECTED	Int32	1001	Already connected
		CMP_E_DISCONNECT	Int32	1002	Not connected
		CMP_E_NOTCONNECT	Int32	1003	Failed to connect
		CMP_E_CONNECT_NOTFOUND	Int32	1004	Non supported model
		CMP_E_CONNECT_OFFLINE	Int32	1005	Failed printer status
		CMP_E_ILLEGAL	Int32	1101	Unsupported or invalid parameter
		CMP_E_OFFLINE	Int32	1102	Off-line
		CMP_E_NOEXIST	Int32	1103	File does not exist
		CMP_E_FAILURE	Int32	1104	Process failure
		CMP_E_TIMEOUT	Int32	1105	Timeout
		CMP_E_NO_LIST	Int32	1106	Printer cannot be found
		CMP_EPTR_COVER_OPEN	Int32	1201	Cover opens
		CMP_EPTR_REC_EMPTY	Int32	1202	Out of paper
		CMP_EPTR_BADFORMAT	Int32	1203	Unsupported file format
		CMP_EPTR_CMP_EPTR_TOOBIG	Int32	1204	Bitmap size too big
2	Connection interface	CMP_PORT_WiFi	Int32	0	Network
		CMP_PORT_BLUETOOTH	Int32	1	Bluetooth
		CMP_PORT_USB	Int32	3	USB (Lightning/Type-C)
		CMP_PORT_SNMP	Int32	6	SNMP
3	Status	CMP_STS_NORMAL	Int32	0	Normal
		CMP_STS_DRAWER_LEVEL_H	Int32	2	Status of pin 3 of drawer kick-out connector = H
		CMP_STS_PAPER_NEAREMPTY	Int32	4	Paper near empty
		CMP_STS_BATTERY_LOW	Int32	8	Printer battery capacity is low
		CMP_STS_COVER_OPEN	Int32	16	Cover opens
		CMP_STS_PAPER_EMPTY	Int32	32	Paper empty
		CMP_STS_MSR_READ	Int32	64	Currently MSR in read mode
		CMP_STS_PRINTEROFF	Int32	128	Off-line
4	Alignment	CMP_ALIGNMENT_LEFT	Int32	0	Left alignment
		CMP_ALIGNMENT_CENTER	Int32	1	Center alignment
		CMP_ALIGNMENT_RIGHT	Int32	2	Right alignment
5	Text attribute	CMP_FNT_DEFAULT	Int32	0	Default font
		CMP_FNT_FONTB	Int32	1	Font B
		CMP_FNT_FONTC	Int32	2	Font C
		CMP_FNT_BOLD	Int32	8	Bold
		CMP_FNT_REVERSE	Int32	16	Reverse
		CMP_FNT_UNDERLINE	Int32	128	Underline
		CMP_FNT_ITALIC	Int32	256	Italic
		CMP_FNT_STRIKEOUT	Int32	512	Strikeout
6	Text size	CMP_TXT_1WIDTH	Int32	0	1 times width
		CMP_TXT_2WIDTH	Int32	16	2 times width
		CMP_TXT_3WIDTH	Int32	32	3 times width
		CMP_TXT_4WIDTH	Int32	48	4 times width
		CMP_TXT_5WIDTH	Int32	64	5 times width
		CMP_TXT_6WIDTH	Int32	80	6 times width
		CMP_TXT_7WIDTH	Int32	96	7 times width
		CMP_TXT_8WIDTH	Int32	112	8 times width

		CMP_TXT_1HEIGHT	Int32	0	1 times height
		CMP_TXT_2HEIGHT	Int32	1	2 times height
		CMP_TXT_3HEIGHT	Int32	2	3 times height
		CMP_TXT_4HEIGHT	Int32	3	4 times height
		CMP_TXT_5HEIGHT	Int32	4	5 times height
		CMP_TXT_6HEIGHT	Int32	5	6 times height
		CMP_TXT_7HEIGHT	Int32	6	7 times height
		CMP_TXT_8HEIGHT	Int32	7	8 times height
7	Side	CMP_SIDE_RIGHT	Int32	0	Right side
		CMP_SIDE_LEFT	Int32	1	Left side
8	Bitmap width	CMP_BM_ASIS	Int32	-11	One bitmap pixel per printer dot
9	Bitmap mode	CMP_BM_MODE_CMD_RASTER	Int32	1	Monochrome print (Raster command)
		CMP_BM_MODE_CMD_BITIMAGE	Int32	2	Monochrome print (Bit image command)
		CMP_BM_MODE_CMD_MONO	Int32	8	Monochrome register
		CMP_BM_MODE_CMD_GRAY16	Int32	8	Grayscale print/ register
		CMP_BM_MODE_HT_THRESHOLD	Int32	16	Halftone (Threshold)
		CMP_BM_MODE_HT_DITHER	Int32	32	Halftone (Dither)
		CMP_BM_MODE_CMD_GRAY16DOW NLOAD	Int32	256	Grayscale print (Download graphics command)
10	Barcode symbology	CMP_BCS_UPCA	Int32	101	UPC-A
		CMP_BCS_UPCE	Int32	102	UPC-E
		CMP_BCS_EAN8	Int32	103	EAN8
		CMP_BCS_EAN13	Int32	104	EAN13
		CMP_BCS_JAN8	Int32	105	JAN8
		CMP_BCS_JAN13	Int32	106	JAN13
		CMP_BCS_ITF	Int32	107	Interleaved 2 of 5
		CMP_BCS_Codabar	Int32	108	Codabar
		CMP_BCS_Code39	Int32	109	Code30
		CMP_BCS_Code93	Int32	110	Code93
		CMP_BCS_Code128	Int32	111	Code128
		CMP_BCS_GS1DATABAR	Int32	131	GS1 DataBar Omnidirectional
		CMP_BCS_GS1DATABAR_E	Int32	132	GS1 DataBar Expanded
		CMP_BCS_GS1DATABAR_S	Int32	133	GS1 DataBar Stacked
		CMP_BCS_GS1DATABAR_E_S	Int32	134	GS1 DataBar Expanded Stacked
		CMP_BCS_GS1DATABAR_T	Int32	135	GS1 DataBar Truncated
		CMP_BCS_GS1DATABAR_L	Int32	136	GS1 DataBar Limited
		CMP_BCS_GS1DATABAR_S_O	Int32	137	GS1 DataBar Stacked Omnidirectional
11	HRI characters	CMP_HRI_TEXT_NONE	Int32	0	None
		CMP_HRI_TEXT_ABOVE	Int32	1	Above the barcode
		CMP_HRI_TEXT_BELOW	Int32	2	Below the barcode
12	Error correction level (PDF417)	CMP_PDF417_EC_LEVEL_0	Int32	48	Level 0
		CMP_PDF417_EC_LEVEL_1	Int32	49	Level 1
		CMP_PDF417_EC_LEVEL_2	Int32	50	Level 2
		CMP_PDF417_EC_LEVEL_3	Int32	51	Level 3
		CMP_PDF417_EC_LEVEL_4	Int32	52	Level 4
		CMP_PDF417_EC_LEVEL_5	Int32	53	Level 5
		CMP_PDF417_EC_LEVEL_6	Int32	54	Level 6
		CMP_PDF417_EC_LEVEL_7	Int32	55	Level 7
13	Error correction level (QR Code)	CMP_QRCODE_EC_LEVEL_L	Int32	48	Level L (7%)
		CMP_QRCODE_EC_LEVEL_M	Int32	49	Level M (15%)

		CMP_QRCODE_EC_LEVEL_Q	Int32	50	Level Q (25%)
		CMP_QRCODE_EC_LEVEL_H	Int32	51	Level H (30%)
14	Cut type	CMP_CUT_FULL	Int32	-1	Full cut
		CMP_CUT_PARTIAL	Int32	-2	Partial cut
		CMP_CUT_FULL_PREFEED	Int32	-3	Feed and full cut
		CMP_CUT_PARTIAL_PREFEED	Int32	-4	Feed and partial cut
15	Mark feed type	CMP_MF_TO_CUTTER	Int32	2	Feed and cut
		CMP_MF_TO_NEXT_TOF	Int32	8	Feed to the next top
16	Drawer number	CMP_DRAWER_1	Int32	1	Drawer 1
		CMP_DRAWER_2	Int32	2	Drawer 2
17	Transaction control	CMP_TP_TRANSACTION	Int32	11	Begin transaction
		CMP_TP_NORMAL	Int32	12	End transaction
18	Rotation control	CMP_RT_NORMAL	Int32	0x0001	End rotation
		CMP_RT_ROTATE180	Int32	0x0103	Begin upside-down rotation
		CMP_RP_BARCODE	Int32	0x1000	Begin barcode rotation
		CMP_RP_BITMAP	Int32	0x2000	Begin bitmap rotation
19	Page mode control	CMP_PM_PAGE_MODE	Int32	1	Begin page mode
		CMP_PM_PRINT_SAVE	Int32	2	Print and save canvas
		CMP_PM_NORMAL	Int32	3	Print and exit page mode
		CMP_PM_CANCEL	Int32	4	Cancel page mode
20	Page mode direction	CMP_PD_LEFT_TO_RIGHT	Int32	1	Normal printing
		CMP_PD_BOTTOM_TO_TOP	Int32	2	Rotated left 90° printing
		CMP_PD_RIGHT_TO_LEFT	Int32	3	Upside down printing
		CMP_PD_TOP_TO_BOTTOM	Int32	4	Rotated right 90° printing
21	Watermark control	CMP_WM_STOP	Int32	0	End watermark
		CMP_WM_START	Int32	1	Begin watermark
22	Map mode type	CMP_MM_DOTS	Int32	1	The printer's dot width
		CMP_MM_TWIPS	Int32	2	1/1440 of an inch
		CMP_MM_ENGLISH	Int32	3	0.001 inch
		CMP_MM_METRIC	Int32	4	0.01 millimeter
23	Extended error code	CMP_EX_DEV_NO_PRINTER	Int32	62000	Printer not connected
		CMP_EX_DEV_NO_SERIALNO	Int32	62012	Printer serial number mismatch
		CMP_EX_DEV_OPEN_ERROR	Int32	62100	Failed to connect socket or stream
		CMP_EX_DEV_NO_RESPONSE	Int32	62102	Command no response

3. Line Display Control

3.1. Program structure

Here is an example program which uses the SDK

<pre>// Create an instance. var display: LineDisplay? = CSJPOSLibSwift.LineDisplay() var result: Int32 = 0 // Connect Linedisplay result = display!.connect(LineDisplayConst.CDP_PORT_WiFi, withAddress: "192.168.0.10") if LineDisplayConst.CMP_SUCCESS == result { // Set encoding _ = display!.setEncoding(String.Encoding.shiftJIS) // Clear text _ = display!.clearDisplay(LineDisplayConst.CDP_AREA_ALL) // Display text _ = display!.displayText("123456"); // Set cursor position _ = display!.setCursorPosition(1, y: 2) // Display text (Reverse) result = display!.displayText("123456", reverseFlag: true) // Disconnect _ = display!.disconnect() if LineDisplayConst.CMP_SUCCESS != result { messageBox("DisplayText Error : \(result)", withTitle: "Error", withAutoDismiss: true) } } else { // Connect Error Log.e("Citizen_POS_sample1", "Connect or LineDisplay Error : " + Integer.toString(result)); }</pre>	<div style="margin-bottom: 10px;"> } Class definition </div> <div style="margin-bottom: 10px;"> } Connect </div> <div style="margin-bottom: 10px;"> } Linedisplay processes </div> <div style="margin-bottom: 10px;"> } Disconnect </div>
---	---

3.2. Functions list

This SDK provides the following functions.

Methods list

No	Function	Detail
1	Create class (instance)	This is instance method.
2	Connect display (connect method)	This method connects to the line display
3	Disconnect display (disconnect method)	This method disconnects the line display connection.
4	Display the text (displayText method)	This method is used to display text.
5	Clear the displayed text (clearDisplay method)	This method clears the displayed text.
6	Blink the display (blinkDisplay method)	This method causes the entire display screen to blink.
7	Set display mode (setDisplayMode method)	This method sets the following display modes.
8	Set display config (setDisplayConfig method)	This method changes the brightness of the display screen.
9	Set cursor Position (setCursorPosition method)	This method is used to set the cursor position.
10	Move cursor (moveCursor method)	This method is used to move the cursor.
11	Show cursor position (setCursorPosition method)	This displays the current cursor position on the display.
12	Initialize (initializeDisplay method)	This method initializes the device.
13	Send command (displayData method)	This method sends the command.
14	Set encoding (setEncoding method)	This method sets the encoding of character.
15	Set code page (setCodePage method)	This method sets the code page of character.
16	Set international character set (setInternationalCharacterSet method)	This sets the following international character sets.
17	Check display status (checkDisplay method)	This method is used to check the display connection status.
18	Log function (setLog method)	Set the log function.
19	Get version code (getVersionCode method)	This method gets a numerical value for the version number of this SDK.
20	Get version name (getVersionName method)	This method gets a string for the version number of this SDK.

Properties List

No	Function	Attribute	Detail
1	Extended error code (ErrorCodeExtended property)	R	Show the extended error code when connecting.

3.3. Library interfaces

The following are the interfaces of this SDK.

3.3.1. Return value

Methods to be described later return the value in the list below.

Return value	Description
CDP_SUCCESS (0)	The operation is success.
CDP_E_CONNECTED (1001)	The device is already connected.
CDP_E_DISCONNECT (1002)	The device is not connected.
CDP_E_NOTCONNECT (1003)	Failed connection to the device.
CDP_E_CONNECT_NOTFOUND (1004)	Failed to check the support model after connecting to the device.
CDP_E_ILLEGAL (1101)	Unsupported operation with the Device, or an invalid parameter value was used.
CDP_E_OFFLINE (1102)	The printer is off-line.
CDP_E_FAILURE (1104)	The Service cannot perform the requested procedure.

3.3.2. Instance

Syntax

LineDisplay

Description

To generate the class and to initialize it.

Example

```
let display: LineDisplay? = CSJPOSLibSwift.LineDisplay()
```

3.3.3. connect method

Syntax

- 1) func connect(connectType: Int32, withAddress addr: String?) -> Int32
- 2) func connect(connectType: Int32, withAddress addr: String?, withPort port: Int32) -> Int32
- 3) func connect(connectType: Int32, withAddress addr: String?, withPort port: Int32, withTimeout timeout: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
connectType	[IN]	Connection type of the printer	CDP_PORT_WiFi: Wi-Fi connection CDP_PORT_BLUETOOTH: Bluetooth connection CDP_PORT_USB: USB connection (Lightning/Type-C)
addr	[IN]	IP address to connect or Bluetooth device address or Bluetooth device name or serial number	Wi-Fi: 0.0.0.0 - 255.255.255.255 Bluetooth: 00:00:00:00:00:00 – FF:FF:FF:FF:FF:FF Device name USB: Blank or serial number
port	[IN]	Connection port number or USB stream number	Wi-Fi: Port number USB: 1 - 3
timeout	[IN]	Timeout (msec)	

Description

This method is used to connect the line display. Please specify the type and address of the printer to which the line display is connected.

When Bluetooth is used, it is possible to connect with only the printer with which pairing and connecting by the BLUETOOTH setting of the iOS device (Please refer to "1.6 Use of Bluetooth"). And, it should be iOS6 or more to be connected by using the Bluetooth device address. Please connect by using the Bluetooth device name besides.

When USB is used, if you specify a space for the address, the connection will be established automatically. It is also to connect to a specific printer by specifying the printer serial number.

Connection port number is valid only if Wi-Fi or USB is specified for the connection type. If it is omitted, it connects with number 9200 for Wi-Fi and stream 1 for USB.

Timeout is gives the maximum number of milliseconds to connect line display. If it is omitted, it connects with 4000 milliseconds when using Wi-Fi/USB and 8000 milliseconds in the case of Bluetooth.

When communication with the line display is not necessary, must execute the [disconnect method](#) to disconnect the line display connection. When not disconnect, the next connection will be an error.

Return value

Return CDP_SUCCESS (0) in success. Please check the description of the error codes below in the case of failure. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Error codes	Description
CDP_E_NOTCONNECT (1003)	Failed connection to the line display. (1) The line display r is under none-connection status.

	(2) The printer is not turned ON. (3) Cannot obtain handle of interface board.
--	---

Example

```
display!.connect(LineDisplayConst.CDP_PORT_WiFi,  
    withAddress: "192.168.182.100", withPort: 9200)  
  
display!.connect(LineDisplayConst.CDP_PORT_BLUETOOTH,  
    withAddress: "00:01:90:F0:81:AB", withPort: 9200, withTimeout:8000)  
  
display!.connect(LineDisplayConst.CDP_PORT_USB, withAddress: "")
```


3.3.4. disconnect method

Syntax

```
func disconnect() -> Int32
```

Parameter

Not exist.

Description

This method is used to disconnect the line display connection.

When the end of the line display or some kind of error occurs, please disconnect the connection by the execution of this method.

Return value

Return CDP_SUCCESS(0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.disconnect()
```

3.3.5. displayText method

Syntax

```
func displayText(data: String?) -> Int32
```

```
func displayText(data:String?, reverseFlag flag: Bool) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
Data	Text data	String
ReverseFlag	Reverse specification flag	false: Standard true: Reverse When the argument is omitted, it is treated as false.

Description

This method is used to display text from the current cursor position.

Reverse can be specified for the text attribute.

Return value

Return CDP_SUCCESS (0) in success. Please refer to ["3.3.1 Return value"](#) for the error code except it.

Example

```
display!.displayText("Hello, World!")
```

3.3.6. clearDisplay method

Syntax

```
int clearDisplay (displayArea: UInt32) -> Int32)
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
displayArea	Clear area	CDP_AREA_ALL(0): Entire area CDP_AREA_CURSORLINE(1):Cursor line When the argument is omitted, it is treated as CDP_AREA_ALL.

Description

This method clears the displayed text.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.clearDisplay(LineDisplayConst.CDP_AREA_ALL)
```

3.3.7. blinkDisplay method

Syntax

```
func blinkDisplay(intervalBlinkg: UInt32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
IntervalBlink	Blink interval (msec)	From 0

Description

This method causes the entire display screen to blink.

The blink interval (msec) specifies the interval for on and off. If 0 is specified for the blink interval, blinking is disabled.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.blinkDisplay(1000)
```

3.3.8. setDisplayMode method

Syntax

```
func setDisplayMode(displayMode: Int32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
DisplayMode	Display mode	CDP_OVERWRITE(1): Overwrite mode CDP_VERTICALSCROLL(2): Vertical scroll mode CDP_HORIZONTALSCROLL(3): Horizontal scroll mode

Description

This method sets the following display modes.

DisplayMode	Overview
Overwrite	Overwrites the text at the cursor position and moves the cursor to the right. (The cursor moves to the bottom left edge for input when it is at the top right edge, and the cursor moves to the top left edge for input when it is at the bottom right edge.)
VerticalScroll	Scrolls the display line of the top edge to the bottom edge by cursor up movement when the cursor is at the top edge (or by left movement when it is at the left edge). Scrolls the display line of the bottom edge to the top edge by cursor down movement when the cursor is at the bottom edge (or by right movement when it is at the right edge).
HorizontalScroll	Scrolls the text leftward in respect to the current cursor line by cursor right movement (or by text input) when the cursor is at the right edge. Scrolls the text rightward in respect to the current cursor line by cursor left movement when the cursor is at the left edge.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.setDisplayMode(LineDisplayConst.CDP_VERTICALSCROLL)
```

3.3.9. setDisplayConfig method

Syntax

```
func setDisplayConfig(brightness: UInt32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
Brightness	Brightness (%)	0 to 100

Description

This method changes the brightness of the display screen.

The higher the numerical value, the brighter the brightness becomes. If 0 is specified, the screen turns off (the display content is retained).

After this is set, blinking of the entire display screen is disabled.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.setDisplayConfig(40);
```

3.3.10. setCursorPosition method

Syntax

```
func setCursorPosition(x: UInt32, y: UInt32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
x	Digit position	From 1
y	Line position	From 1

Description

This method is used to set the cursor position.

The cursor position is the movement coordinates of the cursor, and specifies the digit position and line position.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.setCursorPostistion(1, y:2);
```

3.3.11. moveCursor method

Syntax

```
func moveCursor(dx: Int32, y dy: Int32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Value	Meaning	Settable range
dx	Rightward/leftward movement amount	-128 to 127
dy	Upward/downward movement amount	-128 to 127

Description

This method is used to move the cursor.

Movement is from the current cursor position. Specify the leftward/rightward movement amount (-: leftward, +: rightward) and upward/downward movement amount (-: upward, +: downward) for the cursor movement amount.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.moveCursor(2, dy:0);
```


3.3.12. setCursorType method

Syntax

```
func setCursorType(cursorType: UInt32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Element	Meaning	Settable range
CursorType	Cursor type specification	CDP_TYPE_NONE(0): Hide cursor CDP_TYPE_UNDERLINE(1): Display cursor (Omittable element, TYPE_UNDERLINE when omit)

Description

This displays the current cursor position on the display.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.setCursorType(LineDisplayConst.CDP_TYPE_UNDERLINE);
```

3.3.13. initializeDisplay method

Syntax

```
func initializeDisplay() -> Int32
```

Parameter

None

Description

Initializes the device.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.initializeDisplay();
```

3.3.14. displayData method

Syntax

```
func displayData(data: UnsafeMutablePointer<UInt8>, withLength length: UInt) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Element	Meaning	Settable range
data	Send data	

Description

This method is used to transmit byte data as it is to the device.

Be careful not to affect other methods when using it.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
// Execute self test
var data: [Int8] = [0x1f, 0x40]
res = display!.displayData(&data)
```

3.3.15. setEncoding method

Syntax

```
func setEncoding(charset: String.Encoding) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Element	Meaning	Settable range
data	Send data	

Description

This method is used to set the encoding of the send data to the display.

When you create an instance, it is initialized to the default character set of the OS.

Please set the encoding by the setting of the memory switch of the printer. (Please refer to "[3.1 Supported models](#)")

When used in Japanese, it is necessary to specify the "Shift-JIS".

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
// Japanese
display!.setEncoding(NSShiftJISStringEncoding)

// Chinese(Simplified Chinese)
let encode = CFStringConvertEncodingToNSStringEncoding( 0x0632 )
display!.setEncoding(encode)

// Chinese(Traditional Chinese)
let encode = CFStringConvertEncodingToNSStringEncoding( 0x0A03 )
display!.setEncoding(encode)

// Korean
let encode = CFStringConvertEncodingToNSStringEncoding( 0x0940 )
display!.setEncoding(encode)
```

3.3.16. setCodePage method

Syntax

```
func setCodePage(codePage: UInt32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Element	Meaning	Settable range
codePage	Code page specification	0 - 255

Description

Please refer to the command reference "ESC t" command of the utilization device for the set point

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
display!.setCodePage(1)
```

3.3.17. setInternationalCharacterSet method

Syntax

```
func setInternationalCharacterSet(characterSet: UInt32) -> Int32
```

Parameter

The meanings and settable values of the parameters are as follows.

Element	Meaning	Settable range
codePage	International character specification	0 - 16

Description

Set the following international character set.

characterSet	InternationalCharacterSet	characterSet	InternationalCharacterSet
0	America	9	Norway
1	France	10	Denmark II
2	Germany	11	Spain II
3	England	12	Latin America
4	Denmark I	13	Korea
5	Sweden	14	Croatia
6	Italy	15	China
7	Spain I	16	Vietnam
8	Japan		

Return value

Return CDP_SUCCESS (0) in success. Please refer to ["3.3.1 Return value"](#) for the error code except it.

Example

```
display!.setInternationalCharacterSet(8)
display!.displayText("Total:¥¥1,010")
```

3.3.18. displayCheck method

Syntax

```
func displayCheck() -> Int32
```

Parameter

Not exist.

Description

This method is used to check the display connection status.

When the execution result of this method is successful, you can confirm that the display is connected.

When the execution result of this method fails, communication error or device error may have occurred.

In this case, reconnect using [disconnect method](#) and [connect method](#).

In the case of network connection, it will be disconnected automatically when left for a long time. To keep the connection, please execute this method periodically.

Return value

Return CDP_SUCCESS (0) in success. Please refer to "[3.3.1 Return value](#)" for the error code except it.

Example

```
var result = display!.displayCheck()
```

3.3.19. setLog method

Syntax

```
func setLog(mode: Int32, withPath path: String?, withMaxSize maxSize: Int32)
```

Parameters

Parameter	[IN/OUT]	Description	Setting range
mode	[IN]	Logging mode	0: None 1: Access logs 2: Error logs
path	[IN]	File path to store	The folder in the application sharing folder.
maxSize	[IN]	Maximum Log Size	0: Unlimited 1 - : Maximum size (MB)

Description

Sets the logging function. See "[3.4.1 Logging function](#)" for more details.

Return value

none

Example

```
display!.setLog(1, withPath: "/", withMaxSize: 10)
```


3.3.20. getVersionCode method

Syntax

```
func getVersionCode() -> Int32
```

Parameter

Not exist.

Description

This method is used to get a numerical value for the version number of this SDK.

Return value

Return a numerical value for the version number of this SDK. (Ver1.00 is 100)

Example

```
var vno = display!.getVersionCode()  
alert("SDK Version:" + vno/100)
```

3.3.21. getVersionName method

Syntax

```
func getVersionName() -> String?
```

Parameter

Not exist.

Description

This method is used to get a string for the version number of this SDK.

Return value

Return a string for the version number of this SDK. (Ver1.00 is "1.00")

Example

```
var vname = display!.getVersionName()  
alert("SDK Version:" + vname)
```

3.3.22. ErrorCodeExtended property

Syntax

Int32

Attribute

Read only

Description

This property holds the extended error code when there is an error connecting to the display. The values are as follows.

Value	Meaning
CDP_EX_DEV_NO_PRINTER (62000)	Failed connection to the printer. (1) The printer is under none-connection status. (2) The printer is not turned ON. (3) Cannot obtain handle of interface board.
CDP_EX_DEV_NO_DEVICE (62001)	Display is not connected. (1) The display is not connected. (2) The setting of the interface port is incorrect.
CDP_EX_DEV_USING (62002)	Display is in use. (1) The display is in use by another application.
CDP_EX_DEV_CONFIRM (62003)	Failed to check the status after connecting to the display. (1) Wrong specified port number.
CDP_EX_DEV_NO_STREAMNO (62011)	Stream number does not match. (USB connection only) (1) Wrong specified port number.
CDP_EX_DEV_NO_SERIALNO (62012)	Printer serial number does not match. (USB connection only) (1) Wrong specified serial number.
CDP_EX_DEV_STATUS_GET_ERROR (62013)	Failed to get device information. (USB connection only) (1) Communication error.
CDP_EX_DEV_OPEN_ERROR (62100)	Unable to connect socket or stream. (1) Communication error.
CDP_EX_DEV_SEND_ERROR (62101)	Failed to send command to display. (1) Communication error.
CDP_EX_DEV_NO_RESPONSE (62102)	No command response from the display. (1) An unsupported display is connected. (2) The setting of the interface board is incorrect.

This property is initialized to 0 at the beginning of the connection process. Refer to this after executing the [connect method](#).

Set property

Not exist.

Get property

```
func getResultCodeExtended() -> Int32
```

Returns the extended error code as the return value.

3.4. Notes

3.4.1. Logging function

This SDK has a logging function which keeps history of executing methods or reading/writing properties. This can be enabled by using the [setLog method](#), or placing a file "CSJPOSLibD.cfg" in the same folder as the SDK.

< Example of CSJPOSLibD.cfg >

```
[LogSetting]      ...Section name (Fixed)
LogMode=1         ...Specifies the log mode.
LogPath=/         ...Specifies the folder in the application sharing folder to store the log files.
LogMaxSize=10     ...Specifies the maximum size of log file in MB.
```

Setting items

- LogMode
Specifies a log mode:
0: None
1: Access log
2: Error log
- LogPath
Specifies a folder to store the log files. "/" of the application sharing folder by default.
- LogMaxSize
Specifies maximum size to log file in MB. "0" sets the size to "unlimited."

Log file name

Log files will be stored with a file name "CSJPOSLibD_" and a number which indicates the day of week(0 to 6. 0: Sunday, 1: Monday...), and a file extension ".log."

Example: CSJPOSLibD_1.log

When the same file name exists, the file will be overwritten if the file is one week older, new logs will be added to the existing file if the file is created on the same day.

Log format

The log file keeps the information of executed methods, accessed properties, timestamps and results.

```
--- Example 1, method (connect) ---
2022/04/06 12:54:21.165 001 METHOD call   connect(0, 192.168.234.100, 9200, 4000)
2022/04/06 12:54:25.970 001 METHOD result connect() -> Success(0)

--- Example 2, method (displayText) ---
2022/04/06 13:13:03.795 001 METHOD call   displayText([See below], 0)
-----Parameter Detail-----
Thank you so much!
-----
2022/04/06 13:13:03.798 001 METHOD result displayText() -> Success(0)
```

- * The logging function could be a "bottleneck" of processing since it tries to keep the information of every single execution of a method or access to a property.
- * Logging could fail without a notification for the reason below.
 - A write-protected device is selected as a storage location.
 - No enough space in the selected storage device.
 - The storage location contains a file with write-protection.
 - No access privilege to a file or folder.
 - Another program is using the log file.

3.4.2. Predefined Constants List

No	Type	Name	Data type	Value	Description
1	Result/Error	CDP_SUCCESS	Int32	0	Successfully completed
		CDP_E_CONNECTED	Int32	1001	Already connected
		CDP_E_DISCONNECT	Int32	1002	Not connected
		CDP_E_CONNECT_NOTFOUND	Int32	1004	Non supported model
		CDP_E_ILLEGAL	Int32	1101	Unsupported or invalid parameter
		CDP_E_OFFLINE	Int32	1102	Off-line
		CDP_E_FAILURE	Int32	1104	Process failure
2	Connection interface	CDP_PORT_WiFi	Int32	0	Network
		CDP_PORT_BLUETOOTH	Int32	1	Bluetooth
		CDP_PORT_USB	Int32	3	USB (Lightning/Type-C)
3	Clear area	CDP_AREA_ALL	Int32	0	Entire area
		CDP_AREA_CURSORLINE	Int32	1	Cursor line
4	Display mode	CDP_OVERWRITE	Int32	1	Overwrite mode
		CDP_VERTICALSCROLL	Int32	2	Vertical scroll mode
		CDP_HORIZONTALSCROLL	Int32	3	Horizontal scroll mode
5	Cursor type specification	CDP_TYPE_NONE	Int32	0	Hide cursor
		CDP_TYPE_UNDERLINE	Int32	1	Display cursor
6	Extended error code	CDP_EX_DEV_NO_PRINTER	Int32	62000	Printer not connected
		CDP_EX_DEV_NO_DEVICE	Int32	62001	Peripheral device not connected
		CDP_EX_DEV_USING	Int32	62002	Peripheral device in use
		CDP_EX_DEV_CONFIRM	Int32	62003	Confirmation failure after peripheral device connection
		CDP_EX_DEV_NO_STREAMNO	Int32	62011	Stream number mismatch
		CDP_EX_DEV_NO_SERIALNO	Int32	62012	Printer serial number mismatch
		CDP_EX_DEV_STATUS_GET_ERROR	Int32	62013	Failed to get device information
		CDP_EX_DEV_OPEN_ERROR	Int32	62100	Failed to connect socket or stream
		CDP_EX_DEV_SEND_ERROR	Int32	62101	Failed to send command
		CDP_EX_DEV_NO_RESPONSE	Int32	62102	Command no response

3.4.3. About SDK version when using Line Display

When using the function of Line Display, please use SDK of Swift 4.0 or later.

4. Barcode Scanner Control

4.1. Program Structure

Here is an example program which uses the SDK

<pre> var scanner: CSJPOSLibSwift.Scanner? override func viewDidLoad() { scanner = CSJPOSLibSwift.Scanner() _ = scanner?.setStatusUpdateEventCallback(self) _ = scanner?.setDataEventCallback(self) } // Data event callback func dataEventCallback(_ data: Data!) { messageBox(String(data: data, encoding: .utf8)!, withTitle: "Data Event", withAutoDismiss: false) } // Status update event callback func statusUpdateEventCallback(_ status: Int32) { messageBox("Status : \(status)", withTitle: "Status Update Event", withAutoDismiss: true) } // Connect Scanner func connectScanner() { let result = scanner!.connect(ScannerConst.CSC_PORT_WiFi, withAddress: "192.168.0.10") if ScannerConst.CSC_SUCCESS != result { // Connect Error messageBox("Connect Error : \(result)", withTitle: "Error", withAutoDismiss: true) } } // Disconnect Scanner func disconnectScanner() { scanner!.disconnect() } </pre>	<div style="margin-bottom: 10px;">} Class definition</div> <div style="margin-bottom: 10px;">} Instance</div> <div style="margin-bottom: 10px;">} Callback registration</div> <div style="margin-bottom: 10px;">} Callback processes</div> <div style="margin-bottom: 10px;">} Connect</div> <div style="margin-bottom: 10px;">} Disconnect</div>
--	---

4.2. Functions list

This SDK provides the following functions.

Methods list

No	Function	Detail
1	Create class (instance)	This is instance method.
2	Connect scanner (connect method)	This method connects to the scanner.
3	Disconnect scanner (disconnect method)	This method disconnects the scanner connection.
4	Log function (setLog method)	Set the log function.
5	Get version code (getVersionCode method)	This method gets a numerical value for the version number of this SDK.
6	Get version name (getVersionName method)	This method gets a string for the version number of this SDK.

Properties List

No	Function	Attribute	Detail
1	Extended error code (ErrorCodeExtended property)	R	Show the extended error code when connecting.

Events list

No	Function	Detail
1	Scan data notification (DataEvent event)	Notifies barcode reading data.
2	Status update notification (StatusUpdateEvent event)	Notify device status update.

4.3. Library interfaces

The following are the interfaces of this SDK

4.3.1. Return value

Methods to be described later return the value in the list below.

Return value	Description
CSC_SUCCESS (0)	The operation is success.
CSC_E_CONNECTED (1001)	The device is already connected.
CSC_E_DISCONNECT (1002)	The device is not connected.
CSC_E_NOTCONNECT (1003)	Failed connection to the device.
CSC_E_CONNECT_NOTFOUND (1004)	Failed to check the support model after connecting to the device.
CSC_E_ILLEGAL (1101)	Unsupported operation with the Device, or an invalid parameter value was used.
CSC_E_OFFLINE (1102)	The printer is off-line.
CSC_E_NOEXIST (1103)	The file name does not exist.
CSC_E_FAILURE (1104)	The Service cannot perform the requested procedure.

4.3.2. Instance

Syntax

Scanner

Description

To generate the class and to initialize it.

Example

```
var scanner: Scanner? = CSJPOSLibSwift.Scanner()
```

4.3.3. connect method

Syntax

- 1) func connect(connectType: Int32, withAddress addr: String?) -> Int32
- 2) func connect(connectType: Int32, withAddress addr: String?, withPort port: Int32) -> Int32
- 3) func connect(connectType: Int32, withAddress addr: String?, withPort port: Int32, withTimeout timeout: Int32) -> Int32

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning	Setting range
connectType	[IN]	Connection type of the printer	CSC_PORT_WiFi: Wi-Fi connection CSC_PORT_BLUETOOTH: Bluetooth connection CSC_PORT_USB: USB connection (Lightning/Type-C)
addr	[IN]	IP address to connect or Bluetooth device address or Bluetooth device name or serial number	Wi-Fi: 0.0.0.0 - 255.255.255.255 Bluetooth: 00:00:00:00:00:00 – FF:FF:FF:FF:FF:FF Device name USB: Blank or serial number
port	[IN]	Connection port number or USB stream number	Wi-Fi: Port number USB: 1 - 3
timeout	[IN]	Timeout (msec)	

Description

This method is used to connect the scanner. Please specify the type and address of the printer connection.

When Bluetooth is used, it is possible to connect with only the printer with which pairing and connecting by the BLUETOOTH setting of the iOS device (Please refer to "1.6 Use of Bluetooth"). And, it should be iOS6 or more to be connected by using the Bluetooth device address. Please connect by using the Bluetooth device name besides.

When USB is used, if you specify a space for the address, the connection will be established automatically. It is also to connect to a specific printer by specifying the printer serial number.

Connection port number is valid only if Wi-Fi or USB is specified for the connection type. If it is omitted, it connects with number 9210 for Wi-Fi and stream 2 for USB.

Timeout is gives the maximum number of milliseconds to connect line display. If it is omitted, it connects with 4000 milliseconds when using Wi-Fi/USB and 8000 milliseconds in the case of Bluetooth.

When communication with the scanner is not necessary, must execute the [disconnect method](#) to disconnect the scanner connection. When not disconnect, the next connection will be an error.

Return value

Return CSC_SUCCESS (0) in success. Please check the description of the error codes below in the case of failure. Please refer to "[4.3.1 Return value](#)" for the error code except it.

Error codes	Description
CSC_E_NOTCONNECT (1003)	Failed connection to the scanner.

- | | |
|--|---|
| | (1) The scanner is under none-connection status.
(2) The printer is not turned ON.
(3) Cannot obtain handle of interface board. |
|--|---|

Example

```
scanner!.connect(ScannerConst.CSC_PORT_WiFi, withAddress: "192.168.182.100",  
                withPort: 9210)  
  
scanner!.connect(ScannerConst.CSC_PORT_BLUETOOTH,  
                withAddress: "00:01:90:F0:81:AB", withPort: 9210, withTimeout:8000)  
  
scanner!.connect(ScannerConst.CSC_PORT_USB, withAddress: "")
```

4.3.4. disconnect method

Syntax

```
func disconnect() -> Int32
```

Parameter

Not exist.

Description

This method is used to disconnect the scanner connection.

When the end of the scanner or some kind of error occurs, please disconnect the connection by the execution of this method.

Return value

Return CSC_SUCCESS(0) in success. Please refer to "[4.3.1 Return value](#)" for the error code except it.

Example

```
scanner!.disconnect()
```

4.3.5. setLog method

Syntax

```
func setLog(mode: Int32, withPath path: String?, withMaxSize maxSize: Int32)
```

Parameters

Parameter	[IN/OUT]	Description	Setting range
mode	[IN]	Logging mode	0: None 1: Access logs 2: Error logs
path	[IN]	File path to store	The folder in the application sharing folder.
maxSize	[IN]	Maximum Log Size	0: Unlimited 1 - : Maximum size (MB)

Description

Sets the logging function. See "[4.4.1 Logging function](#)" for more details.

Return value

none

Example

```
scanner!.setLog(1, withPath: "/", withMaxSize: 10)
```

4.3.6. getVersionCode method

Syntax

```
func getVersionCode() -> Int32
```

Parameter

Not exist.

Description

This method is used to get a numerical value for the version number of this SDK.

Return value

Return a numerical value for the version number of this SDK. (Ver1.00 is 100)

Example

```
var vno = scanner!.getVersionCode();  
alert("SDK Version:" + vno/100);
```

4.3.7. getVersionName method

Syntax

```
func getVersionName() -> String?
```

Parameter

Not exist.

Description

This method is used to get a string for the version number of this SDK.

Return value

Return a string for the version number of this SDK. (Ver1.00 is "1.00")

Example

```
var vname = scanner!.getVersionName();  
alert("SDK Version:" + vname);
```


4.3.8. ErrorCodeExtended property

Syntax

Int32

Attribute

Read only

Description

This property holds the extended error code when there is an error connecting to the scanner. The values are as follows.

Value	Meaning
CSC_EX_DEV_NO_PRINTER (62000)	Failed connection to the printer. (1) The printer is under none-connection status. (2) The printer is not turned ON. (3) Cannot obtain handle of interface board.
CSC_EX_DEV_NO_DEVICE (62001)	Scanner is not connected. (1) The scanner is not connected. (2) The setting of the interface port is incorrect.
CSC_EX_DEV_USING (62002)	Scanner is in use. (1) The scanner is in use by another application.
CSC_EX_DEV_CONFIRM (62003)	Failed to check the status after connecting to the scanner. (1) Wrong specified port number.
CSC_EX_DEV_NO_STREAMNO (62011)	Stream number does not match. (USB connection only) (1) Wrong specified port number.
CSC_EX_DEV_NO_SERIALNO (62012)	Printer serial number does not match. (USB connection only) (1) Wrong specified serial number.
CSC_EX_DEV_STATUS_GET_ERROR (62013)	Failed to get device information. (USB connection only) (1) Communication error.
CSC_EX_DEV_OPEN_ERROR (62100)	Unable to connect socket or stream. (1) Communication error.

This property is initialized to 0 at the beginning of the connection process. Refer to this after executing the [connect method](#).

Set property

Not exist.

Get property

```
func getResultCodeExtended() -> Int32
```

Returns the extended error code as the return value.

4.3.9. DataEvent event

Syntax

```
func dataEventCallback(_ data: Data!)
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning
data	[IN]	Barcode read data

Description

The barcode scanner notifies of the data read from the barcode.

The callback method receives Data type arguments as information read by the barcode scanner from the barcode.

Set callback

```
func setDataEventCallback(callback: Any!) -> Int32
```

Registers a callback method to receive the information that the barcode scanner read from the barcode. When setDataEventCallback method is executed out several times, it is overwritten an appointed callback method last. When null is set for object designation, a callback is canceled.

Example

```
// Register the scan data callback method
scanner!.setDataEventCallback(self)

// callback method
func dataEventCallback(_ data: Data!) {
    // Data receive process
}
```

4.3.10. StatusUpdateEvent event

Syntax

```
func statusUpdateEventCallback(_ status: Int32)
```

Parameter

The meaning and the setting range of the parameters are as follows.

Value	[IN/OUT]	Meaning
status	[IN]	Status update of device

Description

Notifies device status update.

The callback method receives an Int32 type argument indicating the status as information on the status update of the device.

Status code	Description
CSC_SUE_POWER_ONLINE (2001)	Device is ready
CSC_SUE_POWER_OFF (2002)	Connection fault or not connected to the printer

Set callback

```
func setStatusUpdateEventCallback(callback: Any!) -> Int32
```

Registers a callback method that status receives update information.

When setStatusUpdateEventCallback method is executed out several times, it is overwritten an appointed callback method last. When null is set for object designation, a callback is canceled.

Example

```
// Register the scan data callback method
scanner!.setStatusUpdateEventCallback(self)

// callback method
func statusUpdateEventCallback(_ status: Int32) {
    if ScannerConst.CSC_SUE_POWER_ONLINE == status {
        // Online
    } else {
        // Offline
    }
}
```

4.4. Notes

4.4.1. Logging function

This SDK has a logging function which keeps history of executing methods or reading/writing properties. This can be enabled by using the [setLog method](#), or placing a file "CSJPOSLibS.cfg" in the same folder as the SDK.

< Example of CSJPOSLibS.cfg >

```
[LogSetting]      ...Section name (Fixed)
LogMode=1         ...Specifies the log mode.
LogPath=/         ...Specifies the folder in the application sharing folder to store the log files.
LogMaxSize=10     ...Specifies the maximum size of log file in MB.
```

Setting items

- LogMode
Specifies a log mode:
0: None
1: Access log
2: Error log
- LogPath
Specifies a folder to store the log files. "/" of the application sharing folder by default.
- LogMaxSize
Specifies maximum size to log file in MB. "0" sets the size to "unlimited."

Log file name

Log files will be stored with a file name "CSJPOSLibS_" and a number which indicates the day of week(0 to 6. 0: Sunday, 1: Monday...), and a file extension ".log."

Example: CSJPOSLibS_1.log

When the same file name exists, the file will be overwritten if the file is one week older, new logs will be added to the existing file if the file is created on the same day.

Log format

The log file keeps the information of executed methods, accessed properties, timestamps and results.

```
--- Example 1, method (connect) ---
2022/04/06 12:54:30.450 001 METHOD call    connect(0, 192.168.234.100, 9210, 4000)
2022/04/06 12:54:30.888 001 METHOD result connect() -> Success(0)

--- Example 2, event (DataEvent) ---
2022/04/06 13:13:21.720 015 EVENT          DataEvent -> <31323334 35363738 39303132>
```

* The logging function could be a "bottleneck" of processing since it tries to keep the information of every single execution of a method or access to a property.

* Logging could fail without a notification for the reason below.

- A write-protected device is selected as a storage location.
- No enough space in the selected storage device.
- The storage location contains a file with write-protection.
- No access privilege to a file or folder.
- Another program is using the log file.

4.4.2. Predefined Constants List

No	Type	Name	Data type	Value	Description
1	Result/Error	CSC_SUCCESS	Int32	0	Successfully completed
		CSC_E_CONNECTED	Int32	1001	Already connected
		CSC_E_DISCONNECT	Int32	1002	Not connected
		CSC_E_CONNECT_NOTFOUND	Int32	1004	Non supported model
		CSC_E_ILLEGAL	Int32	1101	Unsupported or invalid parameter
		CSC_E_OFFLINE	Int32	1102	Off-line
		CSC_E_FAILURE	Int32	1104	Process failure
2	Connection interface	CSC_PORT_WiFi	Int32	0	Network
		CSC_PORT_BLUETOOTH	Int32	1	Bluetooth
		CSC_PORT_USB	Int32	3	USB (Lightning/Type-C)
3	Status	CSC_SUE_POWER_ONLINE	Int32	2001	Device is ready
		CSC_SUE_POWER_OFF	Int32	2002	Connection fault or not connected to the printer
4	Extended error code	CSC_EX_DEV_NO_PRINTER	Int32	62000	Printer not connected
		CSC_EX_DEV_NO_DEVICE	Int32	62001	Peripheral device not connected
		CSC_EX_DEV_USING	Int32	62002	Peripheral device in use
		CSC_EX_DEV_CONFIRM	Int32	62003	Confirmation failure after peripheral device connection
		CSC_EX_DEV_NO_STREAMNO	Int32	62011	Stream number mismatch
		CSC_EX_DEV_NO_SERIALNO	Int32	62012	Printer serial number mismatch
		CSC_EX_DEV_STATUS_GET_ERROR	Int32	62013	Failed to get device information
		CSC_EX_DEV_OPEN_ERROR	Int32	62100	Failed to connect socket or stream

4.4.3. About SDK version when using Barcode Scanner

When using the function of Barcode Scanner, please use SDK of Swift 4.0 or later.

iOS POS Print SDK (Swift) Programing Manual

July 18, 2024 For Ver. 2.12

CITIZEN SYSTEMS JAPAN CO., LTD.

<https://csj.citizen.co.jp/>