



## SII POS for .NET Service object Application Programmer's Guide

Rev.05

[Products]

DSP-A01 Series

Seiko Instruments Inc.

Rev.01 July 2019  
Rev.02 November 2019  
Rev.03 September 2020  
Rev.04 December 2021  
Rev.05 July 2023


Copyright© 2019-2023 by Seiko Instruments Inc.  
All rights reserved.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation in the U.S., Japan, and other countries.

Bluetooth® is a registered trademark of Bluetooth SIG, Inc.

Seiko Instruments Inc. (hereinafter referred to as "SII") has prepared this manual for use by SII personnel, licensees, and customers. The information contained herein is the property of SII and shall not be reproduced in whole or in part without the prior written approval of SII.

SII reserves the right to make changes without notice to the specifications and materials contained herein and shall not be responsible for any damages (including consequential) caused by reliance on the materials presented, including but not limited to typographical, arithmetic, or listing errors.

SII  is a trademark of Seiko Instruments Inc.

---

# Introduction

---

This manual describes the display function of "SII POS for .NET Service Object" (hereinafter referred to as the "software") provided by Seiko Instruments Inc. (hereinafter referred to as "SII").

## Target Products

---

This section lists the product supported by the software.

	Device Name	Description in This Manual
LineDisplay	DSP-A01	Display

See "UnifiedPOS Retail Peripheral Architecture Version1.12" (hereinafter "UPOS APG V 1.12") and "Microsoft Point of Service for .NET – POS for .NET v1.12 SDK Documentation" before using this software.

## Supported Driver

---

SII driver is required for using Display.  
See "1.1 Configuration" for supported SII driver.

## Notation in This Manual

---

The notation in this manual is described.

### Operation and Display

In principle, this manual is written on the basis of the following conditions:

- Screenshots and display layouts of Windows 10
- Operating instructions with a mouse and a keyboard

### Operating System Abbreviations

The operating system abbreviations used in this manual are listed below.

Operating System	Abbreviation
General Microsoft® Windows®	Windows
Microsoft® Windows® 11	Windows 11
Microsoft® Windows® 11 IoT Enterprise	
Microsoft® Windows® 10	Windows 10
Microsoft® Windows® 10 IoT Enterprise	

### Terms

The terms used in this manual are defined as below.

Definition	Description
Configuration program	The program that executes addition and setting change of devices provided by this software. When this software is installed, it will be installed as [SII POS for .NET Utility] on the computer.
Default	The value immediately after satisfying the availability condition.
Display command	Command for controlling the display described in "DSP-A01 SERIES CUSTOMER DISPLAY TECHNICAL REFERENCE".

## Symbols

The symbols used in this manual are described below.

### Caution

- ◆ Notes and limitations are described.

### Reference

- Supplemental information and related matters are described.

---

# Table of Contents

---

## Chapter 1 Overview

---

1.1	Configuration.....	1-1
1.1.1	Structural Diagram .....	1-1
1.2	Operating Environment .....	1-3
1.2.1	System Environment.....	1-3
1.3	Display Settings .....	1-3
1.4	Limitations .....	1-4
1.4.1	Line Display Control.....	1-4

## Chapter 2 Installation

---

2.1	Installation .....	2-1
2.2	Uninstallation.....	2-4

## Chapter 3 How to Operate Configuration Program

---

3.1	Startup.....	3-1
3.2	Screen Layout.....	3-2
3.2.1	Menu Bar.....	3-3
3.2.2	Tool Bar.....	3-3
3.2.3	Device View .....	3-3
3.2.4	Setting View .....	3-4
3.3	Functions.....	3-6
3.3.1	Addition of Device .....	3-6
3.3.2	Changing Device Settings.....	3-10
3.3.3	Deletion of Device .....	3-11
3.3.4	Device Interactive Test .....	3-11
3.3.5	Log Setting .....	3-13

## Chapter 4 Properties, Methods, and Events

---

4.1	Summary .....	4-1
4.2	Data Characters and Escape Sequences.....	4-5
4.3	Common Properties .....	4-6
	CapCompareFirmwareVersion Property.....	4-6
	CapPowerReporting Property.....	4-6
	CapStatisticsReporting Property.....	4-6
	CapUpdateFirmware Property .....	4-7
	CapUpdateStatistics Property.....	4-7
	CheckHealthText Property.....	4-7
	Claimed Property .....	4-8
	DeviceDescription Property .....	4-8
	DeviceEnabled Property R/W .....	4-8
	DeviceName Property.....	4-9
	FreezeEvents Property R/W .....	4-9
	PowerNotify Property R/W .....	4-9
	PowerState Property.....	4-9
	ServiceObjectDescription Property .....	4-10
	ServiceObjectVersion Property.....	4-10
	State Property .....	4-10
	SynchronizingObject Property .....	4-11
4.4	Specific Properties .....	4-12
	CapBrightness Property.....	4-12
	CapCharacterSet Property .....	4-12
	CapScreenMode Property .....	4-12
	CharacterSet Property R/W .....	4-13
	CharacterSetList Property.....	4-13
	Columns Property .....	4-14
	CursorColumn Property R/W .....	4-14
	CursorRow Property R/W .....	4-14
	CursorUpdate Property R/W .....	4-14
	DeviceBrightness Property R/W .....	4-15
	DeviceColumns Property .....	4-15
	DeviceRows Property .....	4-16
	DeviceWindows Property .....	4-16
	Rows Property .....	4-16
	ScreenMode Property R/W .....	4-17
	ScreenModeList Property .....	4-17
4.5	Common Methods.....	4-18
	CheckHealth Method .....	4-18
	Claim Method .....	4-18
	Close Method.....	4-19
	CompareFirmwareVersion Method.....	4-19
	DirectIO Method .....	4-19
	Open Method .....	4-20
	Release Method.....	4-21
	ResetStatistic(string) Method.....	4-21

	ResetStatistics() Method.....	4-21
	ResetStatistics(StatisticCategories) Method.....	4-21
	ResetStatistics(string[]) Method.....	4-21
	RetrieveStatistic(string) Method.....	4-21
	RetrieveStatistics() Method.....	4-21
	RetrieveStatistics(StatisticCategories) Method.....	4-22
	RetrieveStatistics(string[]) Method.....	4-22
	UpdateFirmware Method .....	4-22
	UpdateStatistic Method.....	4-22
	UpdateStatistics(Statistic[]) Method.....	4-22
	UpdateStatistics(StatisticCategories, Object) Method.....	4-22
4.6	Specific Methods.....	4-23
	ClearText Method .....	4-23
	DisplayText Method .....	4-23
	DisplayTextAt Method .....	4-24

## Appendix A Exceptions

---

A.1	Exception Error List.....	A-1
-----	---------------------------	-----



---

# Chapter 1 Overview

---

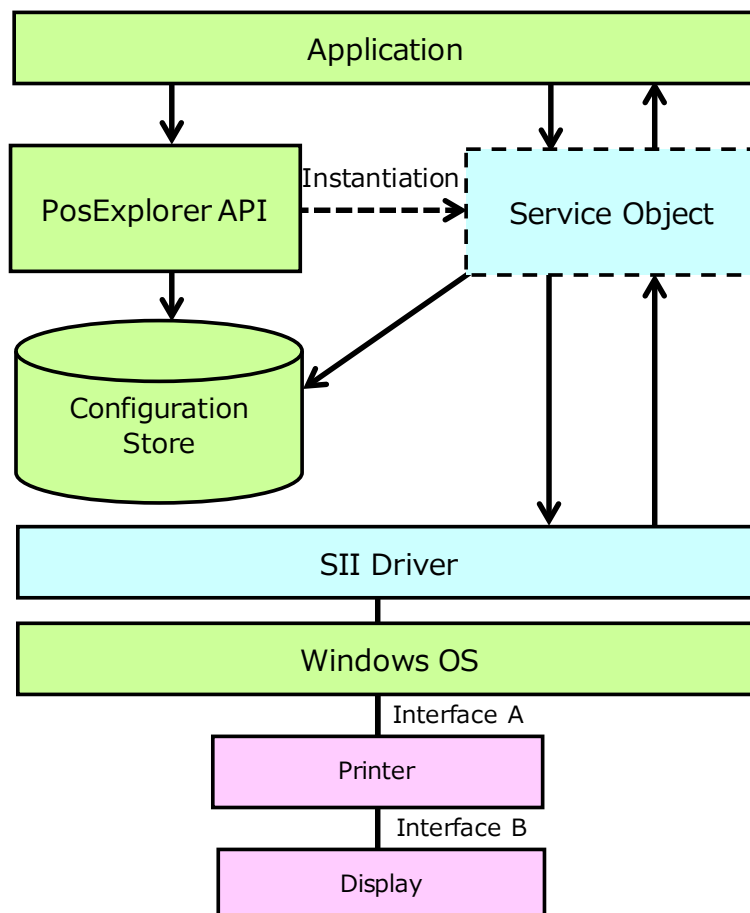
This chapter describes the overview of the software.

## 1.1 Configuration

### 1.1.1 Structural Diagram

The structure of the software is as follows, and the scope of this manual is indicated by broken lines. In addition, the supported SII driver is described. There are 2 types for connecting Display.

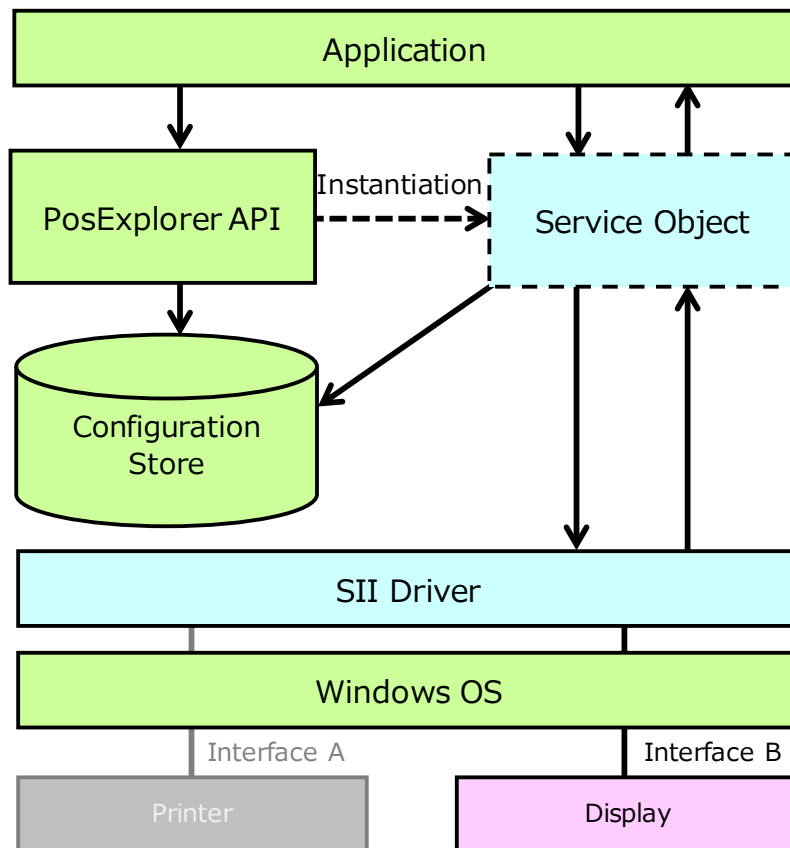
- (1) Configuration that use Display by connecting to SII printer (hereinafter described as "use via a printer")



- Supported driver

SII Driver	Interface A	Printer	Interface B	Display
"SII Printer Driver for Windows" for RP-F10/G10 series	USB Bluetooth TCP/IP	RP-F10 series	USB	DSP-A01 series

(2) Configuration that use Display alone (hereinafter described as "use alone")



- Supported driver

SII Driver	Interface A	Printer*1	Interface B	Display
"SII Printer Driver for Windows" for RP-F10/G10 series	-	-	USB	DSP-A01
"SII Printer Driver for Windows" for DSP-A01 series	-	-	USB	DSP-A01

\*1: SII printer that supports Display is as follows.

- RP-F10 series

## 1.2 Operating Environment

### 1.2.1 System Environment

The system environment of this software is shown below.

Item	Specifications
Operating System	Microsoft® Windows® 11 (64 bits) Microsoft® Windows® 11 IoT Enterprise (64 bits) Microsoft® Windows® 10 (32 bits and 64 bits) Microsoft® Windows® 10 IoT Enterprise (32 bits and 64 bits)
.NET Framework* <sup>1</sup>	.NET Framework 3.5, or .NET Framework 4.0
Microsoft POS for .NET SDK* <sup>1</sup>	POS for .NET 1.12, or POS for .NET 1.14
Driver	SII driver: "SII Printer Driver for Windows" for RP-F10/G10 Series or "SII Printer Driver for Windows" for DSP-A01 Series

\*1: Before installing this software, it is required to install .NET Framework 3.5 and Microsoft POS for .NET 1.12 or .NET Framework 4.0 and Microsoft POS for .NET 1.14 in advance.

## 1.3 Display Settings

The memory switches of Display are set to [Value] in the following table when using the software.  
See "DSP-A01 SERIES Customer Display USER'S GUIDE" for details about the memory switches.

MS	Function	Value	Note
1-1 to 1-8	Brightness Selection (Brightness)	00000000B : 10% 00000001B : 20% 00000010B : 30% 00000011B : 40% 00000100B : 50% 00000101B : 60% 00000110B : 70% 00000111B : 80% 00001000B : 90% 00001001B : 100%	Any one of [Value] on the left can be set by [Brightness(%)] in the configuration program.

## 1.4 Limitations

The limitations of this software are described.

### 1.4.1 Line Display Control

All interfaces of Line Display Device Class defined by UnifiedPOS Retail Peripheral Architecture Version1.9 are provided, with the following limitations.

- (a) The method and property settings related to Teletype Mode and Marquee Mode are not supported.
- (b) The following functions are not supported.
  - Display bitmap
  - Reverse video
  - Blink
- (c) All the following methods always return *ErrorCode.Illegal* after they are enabled.
  - **DisplayBitmap**
  - **SetBitmap**
- (d) The following events are not supported.
  - **StatusUpdateEvent**
  - **DirectIOEvent** (device-specific event)

---

## Chapter 2 Installation

---

This chapter describes the procedure of installation and uninstallation of the software.

It is necessary to install the driver before installing this software.

For the installation procedure of the driver, see the installation part of "SII Printer Driver for Windows User's Guide" described in "1.1 Configuration".

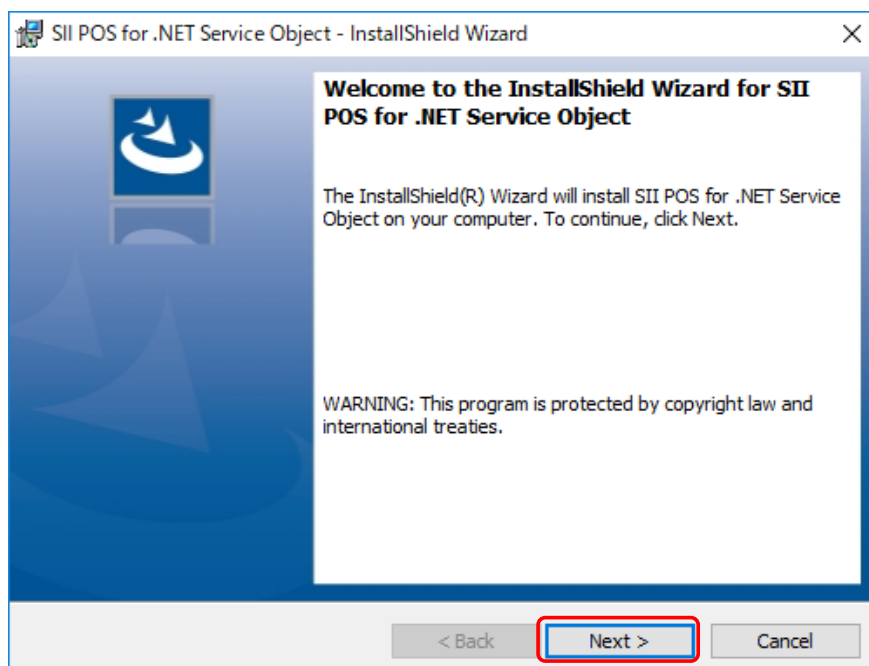
### Caution

- ◆ This installation requires logon to the computer with administrator privileges.

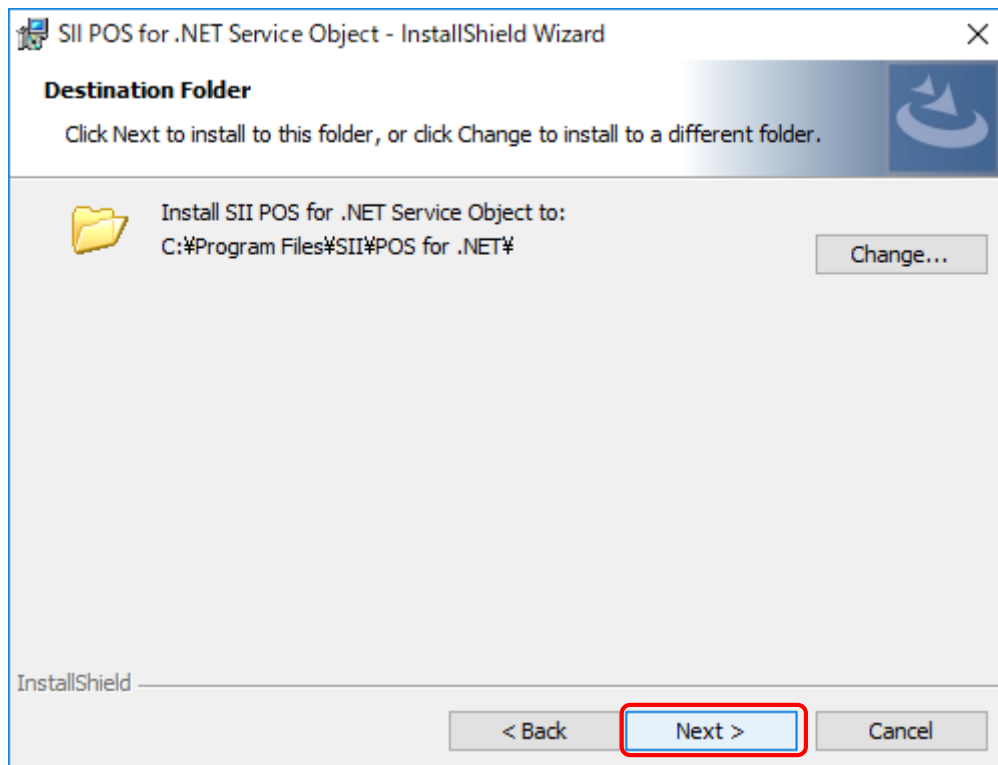
## 2.1 Installation

The installation of this software is described below.

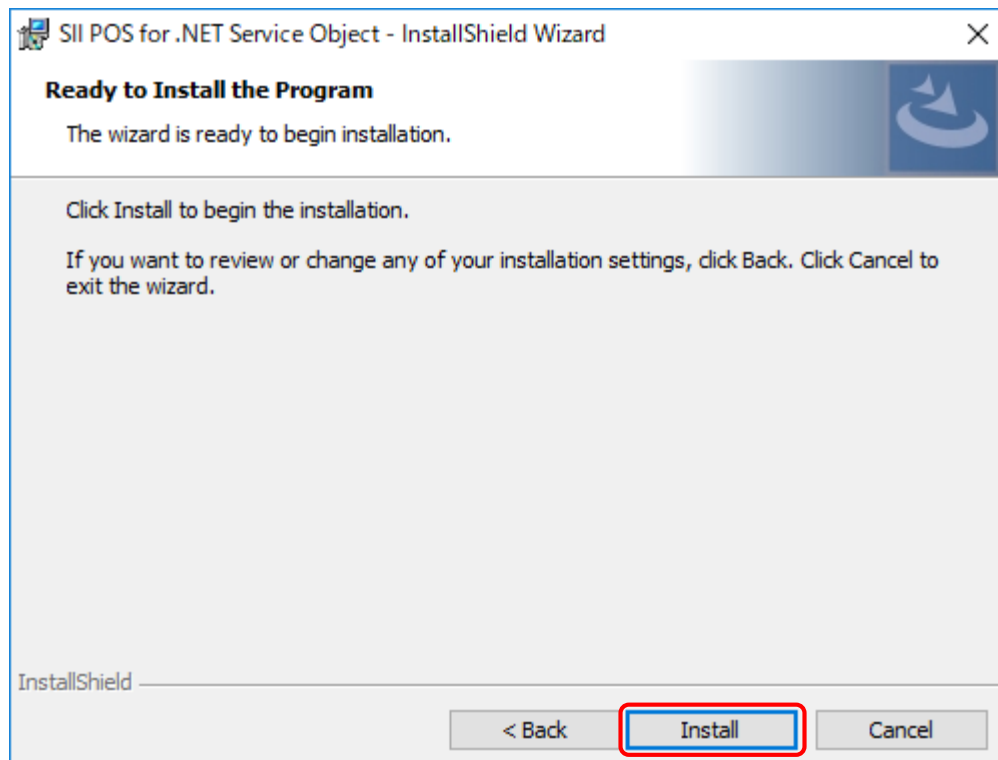
- (1) Start the setup program (SetupOpos.exe).  
For 32-bit OS : SetupPosNet.exe  
For 64-bit OS : SetupPosNet64.exe
- (2) When the installer starts up, click the [Next >] button.



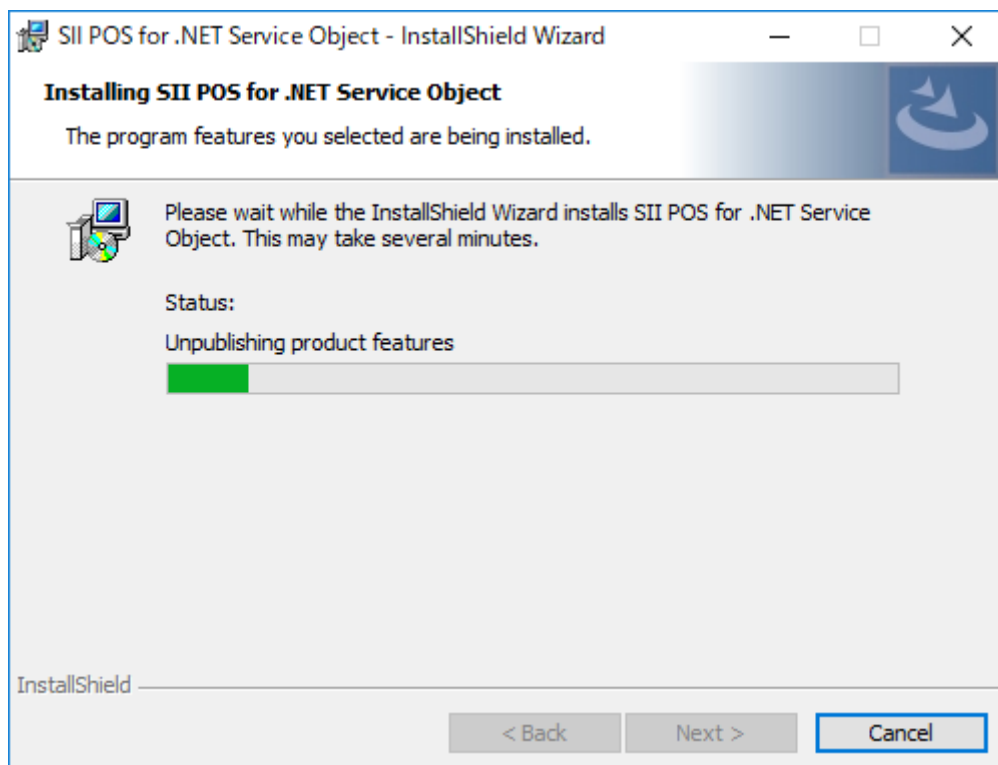
- (3) Specify the destination folder and click the [Next >] button.



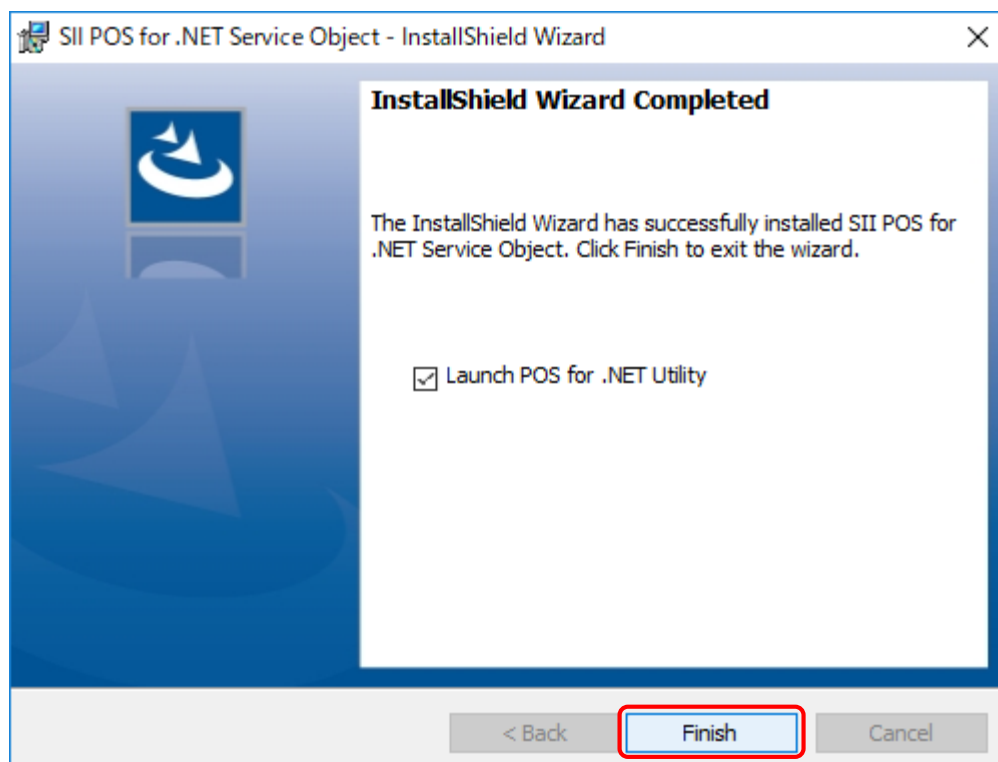
- (4) Click the [Install] button.



- (5) The installation starts, and the progress status bar is displayed.



- (6) Click the [Finish] button. When the [Finish] button is clicked with the "Launch POS for .NET Utility" checkbox on, the setup program ends and the configuration program (POS for .NET Utility) starts up.



## **2.2 Uninstallation**

When the software is no longer used, click [Uninstall a program] in [Programs and Features] in the Control Panel. When the [Uninstall or change a program] window is displayed, select "SII POS for .NET Service Object", and click the [Uninstall] button.



---

## Chapter 3 How to Operate Configuration Program

---

This chapter describes the configuration program provided by this software.

### 3.1 Startup

The startup procedure of the configuration program is described.

- For Windows 11:  
Select [All apps] - [POS for .NET Utility] from the Start menu, and then the configuration program starts up.
- For Windows 10:  
Select [SII POS for .NET] - [POS for .NET Utility] from the Start menu, and then the configuration program starts up.

### Caution

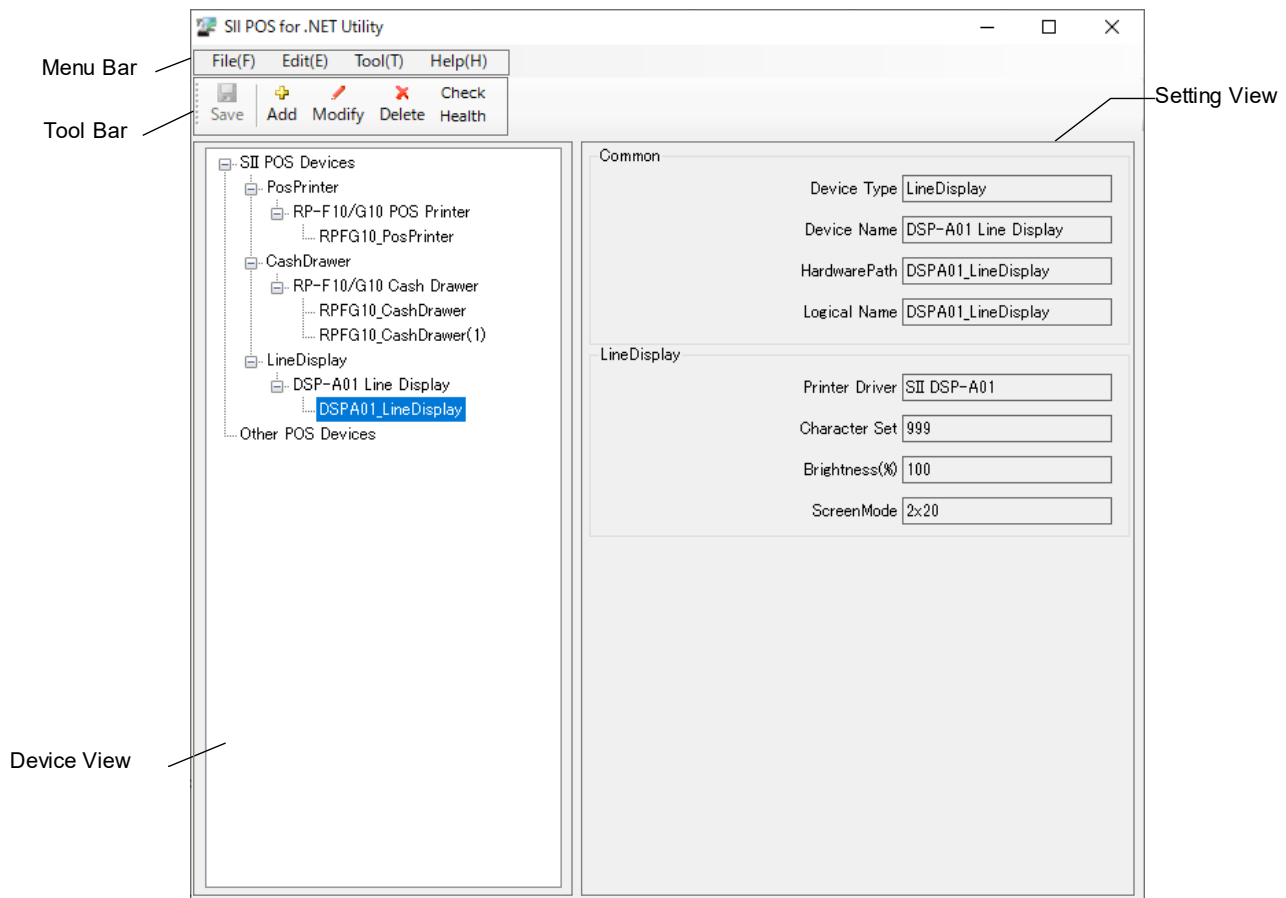
- ◆ This software operates using a printer driver. The printer driver is required to be installed on the computer for using this software.
- ◆ Using this software requires logon to the computer with administrator privileges.

### Reference

- Display driver is recognized as the printer driver by the computer.

## 3.2 Screen Layout

The screen layout of the configuration program is described.



Item	Description
Menu Bar	The menu bar of the configuration program. See "3.2.1 Menu Bar" for items in the menu bar.
Tool Bar	The tool bar of the configuration program. See "3.2.2 Tool Bar" for items in the tool bar.
Device View	The type, the name, and the logical name of the device registered in the system are displayed in a tree.
Setting View	Displays setting contents of the device selected in "Device View". See "3.2.4(1) Display setting items" for items of each device.

### 3.2.1 Menu Bar

Item		Description
File(F)	Save(S)	Saves the data being edited in configuration.xml.
	Restore(R)	Discards the data being edited and re-reads the data saved in configuration.xml.
	End(E)	Ends the configuration program.
Edit(E)	Add(A)	Adds a new device.
	Modify(M)	Changes the setting contents for the device being selected.
	Delete(D)	Deletes the device being selected.
Tool(T)	CheckHealth	Executes an interactive test on the device being selected.
	LogSetting	Performs common log settings for all devices. See "3.3.5 Log Setting" for details of log settings.
Help(H)	About SII POS for .NET Utility(A)	Displays the version information of the configuration program.
	Language Mode(L)	Japanese(J) Displays the configuration program in Japanese. English(E) Displays the configuration program in English.

### 3.2.2 Tool Bar

Item	Description
Save	Performs the same process as Menu Bar - [File(F)] - [Save(S)].
Add	Performs the same process as Menu Bar - [Edit(E)] - [Add(A)].
Modify	Performs the same process as Menu Bar - [Edit(E)] - [Modify(M)].
Delete	Performs the same process as Menu Bar - [Edit(E)] - [Delete(D)].
CheckHealth	Performs the same process as Menu Bar - [Tool(T)] - [CheckHealth].

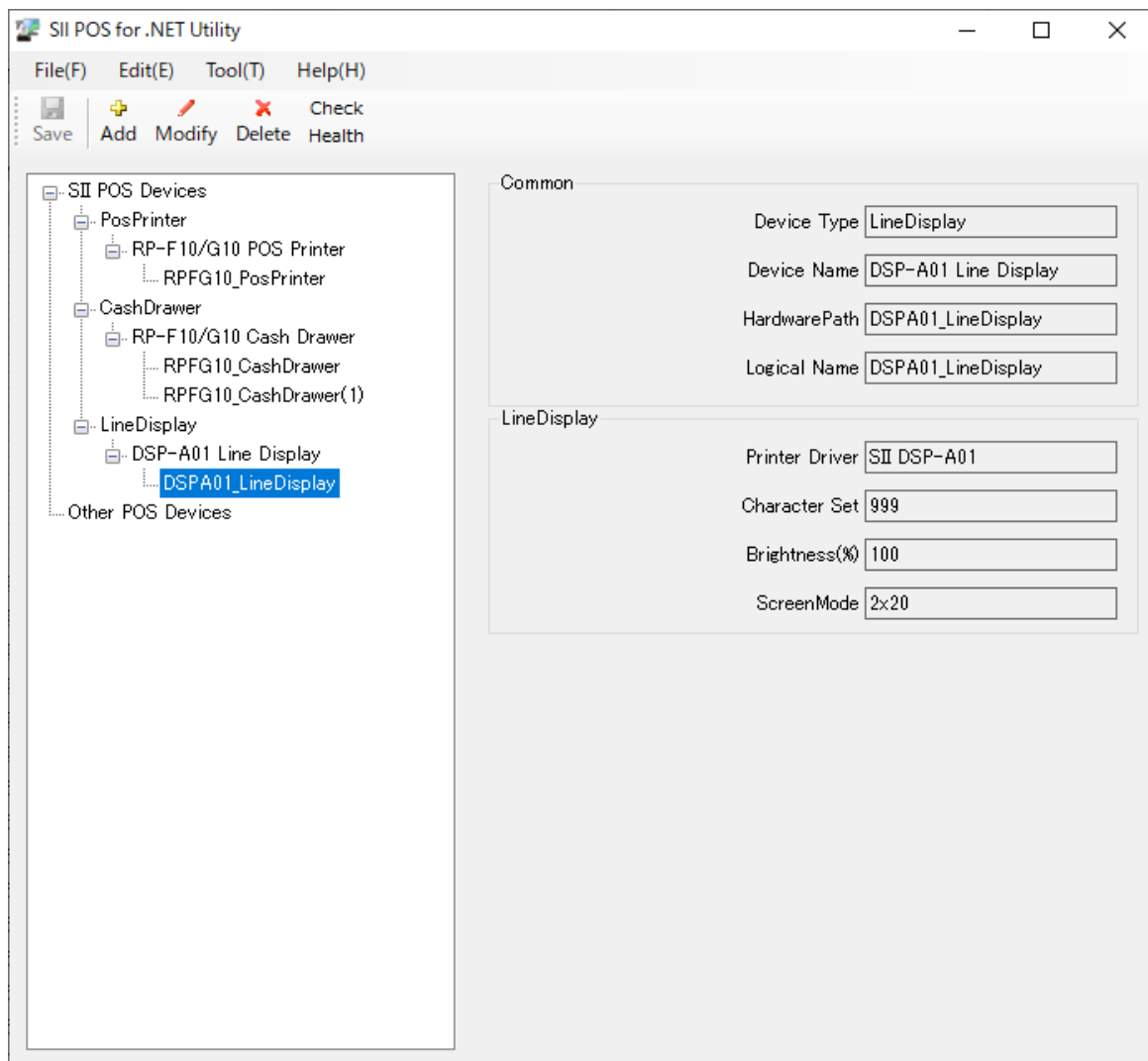
### 3.2.3 Device View

Name	Description
SII POS Devices	Displays SII devices. When the logical name is selected in "Device View", setting contents of the device can be changed or deleted.
Other POS Devices	Displays devices other than SII devices. Device settings cannot be changed or deleted.

### 3.2.4 Setting View

#### (1) Display setting items

The items displayed in "Setting View" and setting contents are described below.



Item	Description	Setting Content (" " : Default)
Common		
Device Type	Device type	[LineDisplay]
Device Name	Device name	DSP-A01 Line Display
HardwarePath	Set automatically. It cannot be changed.	-
Logical Name	Any logical name entered	-

Item	Description	Setting Content (" " : Default)
LineDisplay		
Printer Driver	SII driver used for connecting with Display	-
Character Set	Character set type <b>CharacterSet</b> is initialized with this value. See <b>CharacterSet</b> for details.	437 737 850 852 855 857 858 860 863 865 866 932*1 999*2 1250 1251 1252 1253 1254
Brightness (%)	Brightness setting	0 10 20 30 40 50 60 70 80 90 100
ScreenMode	Screen mode of the device Number of rows × number of columns	2×20 5×20 2×40 5×40 8×40

\*1: Default for Japanese

\*2: Default for English

### 3.3 Functions

The functions of the configuration program are described.

#### 3.3.1 Addition of Device

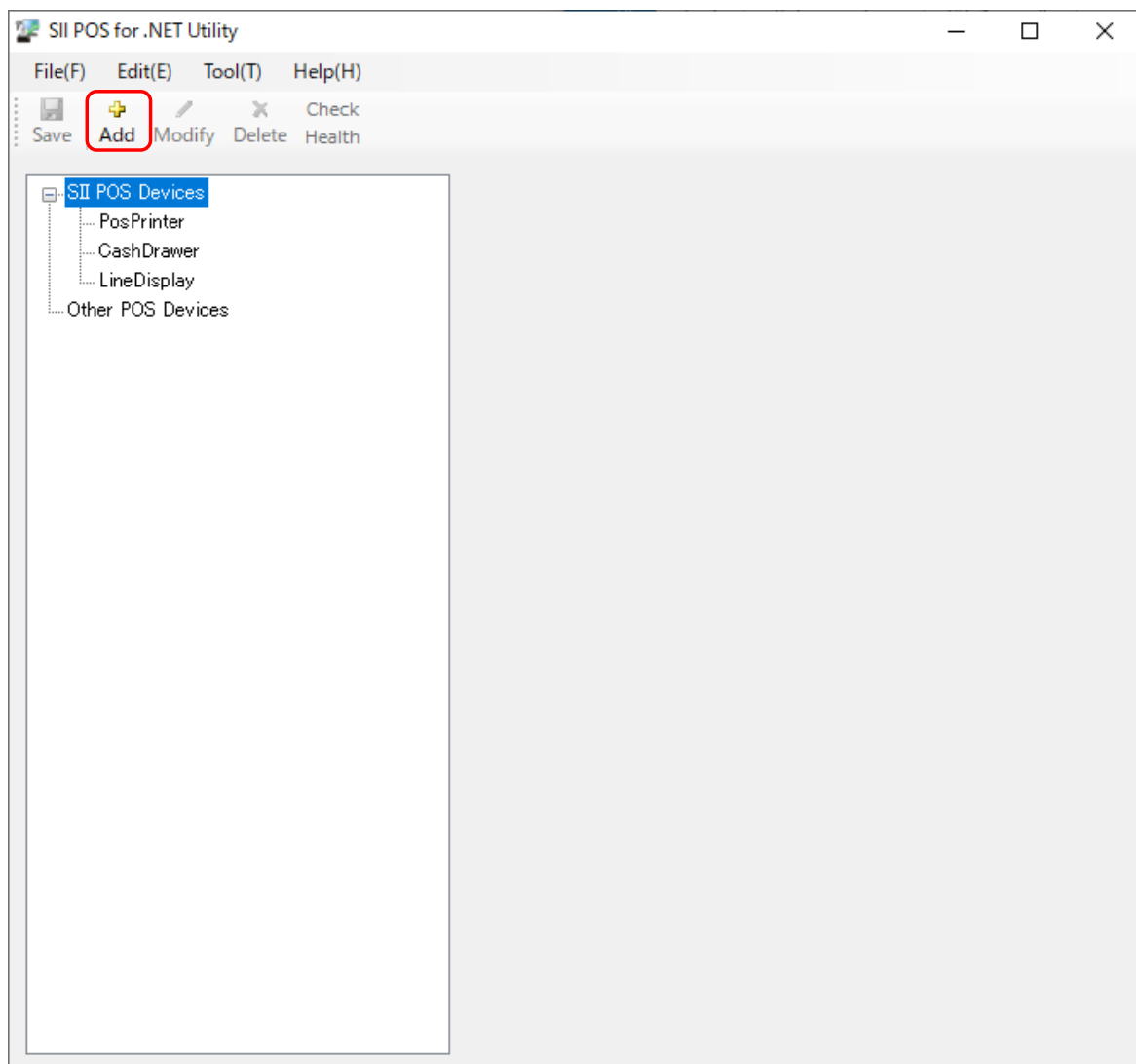
The procedure for adding a device is described.

When the configuration program is started up immediately after installing this software, a device needs to be added since no device has been added.

When a new device is added, it is necessary to install the printer driver for the communication port to be used in advance. See "SII Printer Driver for Windows User's Guide" described in "1.1 Configuration" for installation of the printer driver.

##### (1) Addition of Display

(a) When the configuration program starts, the following window is displayed. Click the [Add] button.



- (b) Select "LineDisplay" for [Device Type] and "DSP-A01 Line Display" for [Device Name], then click the [Next] button.

The screenshot shows a dialog box titled "Add PosDevice" with a close button (X) in the top right corner. The main area is labeled "Select Device". It contains two dropdown menus: "Device Type" with "LineDisplay" selected, and "Device Name" with "DSP-A01 Line Display" selected. At the bottom right, there are two buttons: "Next" and "Cancel". The "Next" button is highlighted with a red rectangular border.

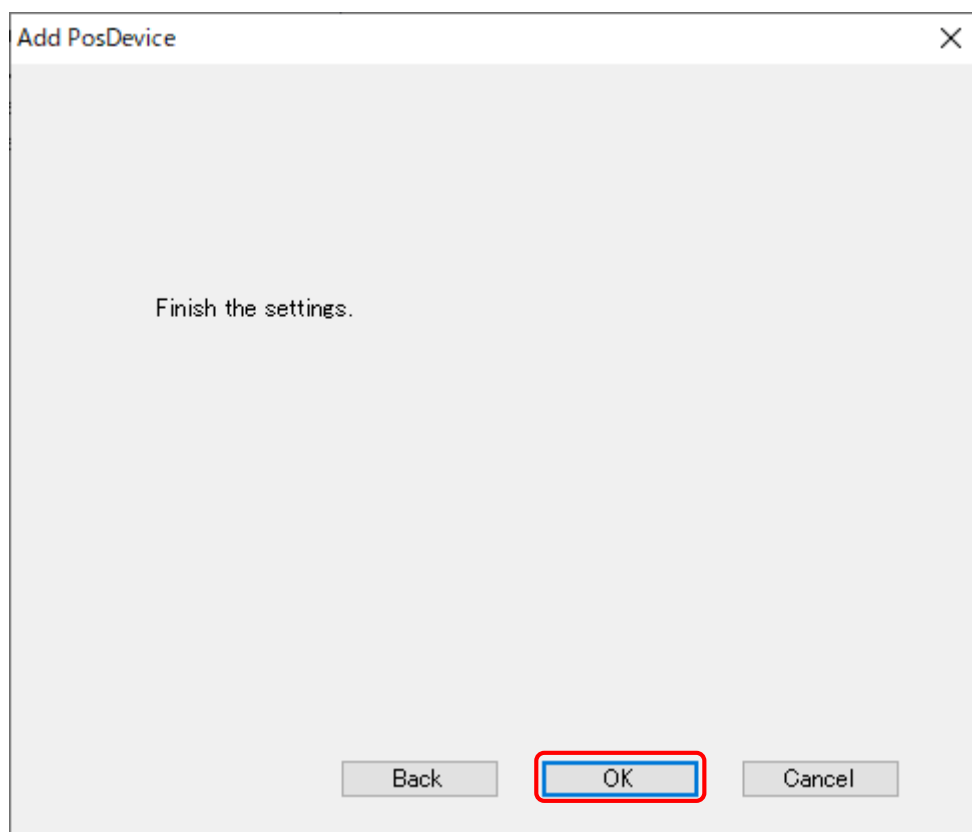
- (c) Enter or select settings of the device, and then click the [Next] button.

The screenshot shows the same "Add PosDevice" dialog box, but now the main area is labeled "Set LineDisplay properties". It contains five input fields: "Logical Name" with the text "DSPA01\_LineDisplay", "Printer Driver" with a dropdown menu showing "SII DSP-A01", "Character Set" with a dropdown menu showing "999", "Brightness(%)" with a dropdown menu showing "100", and "ScreenMode" with a dropdown menu showing "2x20". At the bottom, there are three buttons: "Back", "Next", and "Cancel". The "Next" button is highlighted with a red rectangular border.

## Caution

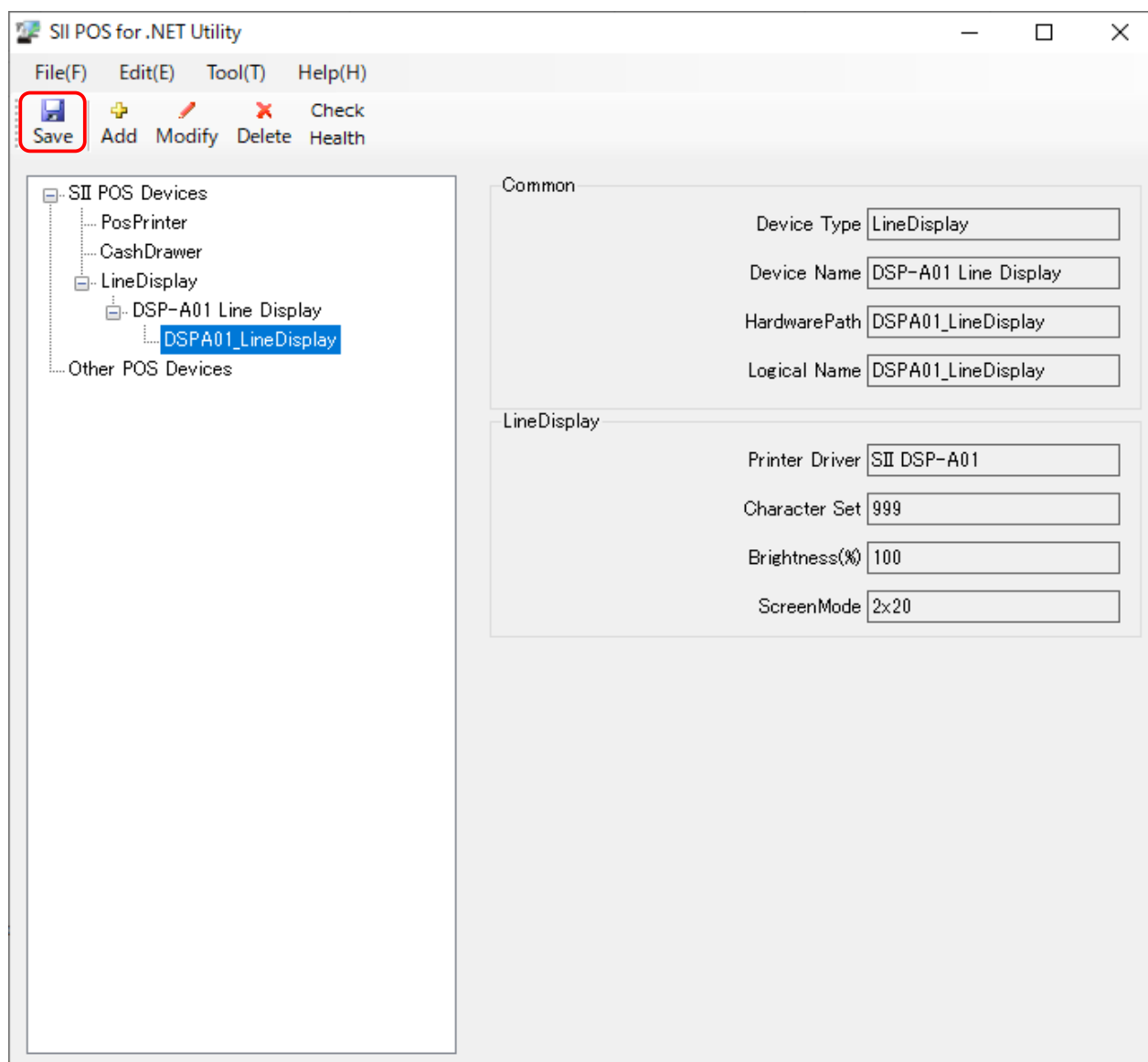
- ◆ The same logical name cannot be set for multiple service objects.

(d) Click the [OK] button.





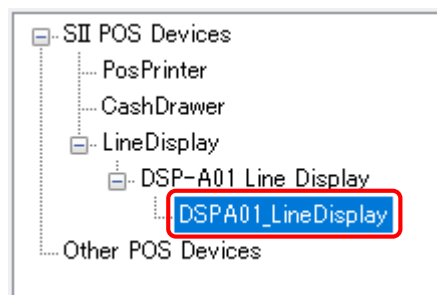
(e) Confirm the contents in "Setting View", and click the [Save] button.



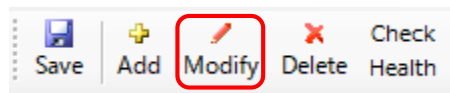
### 3.3.2 Changing Device Settings

The settings of the added device can be changed with the [Modify] button.

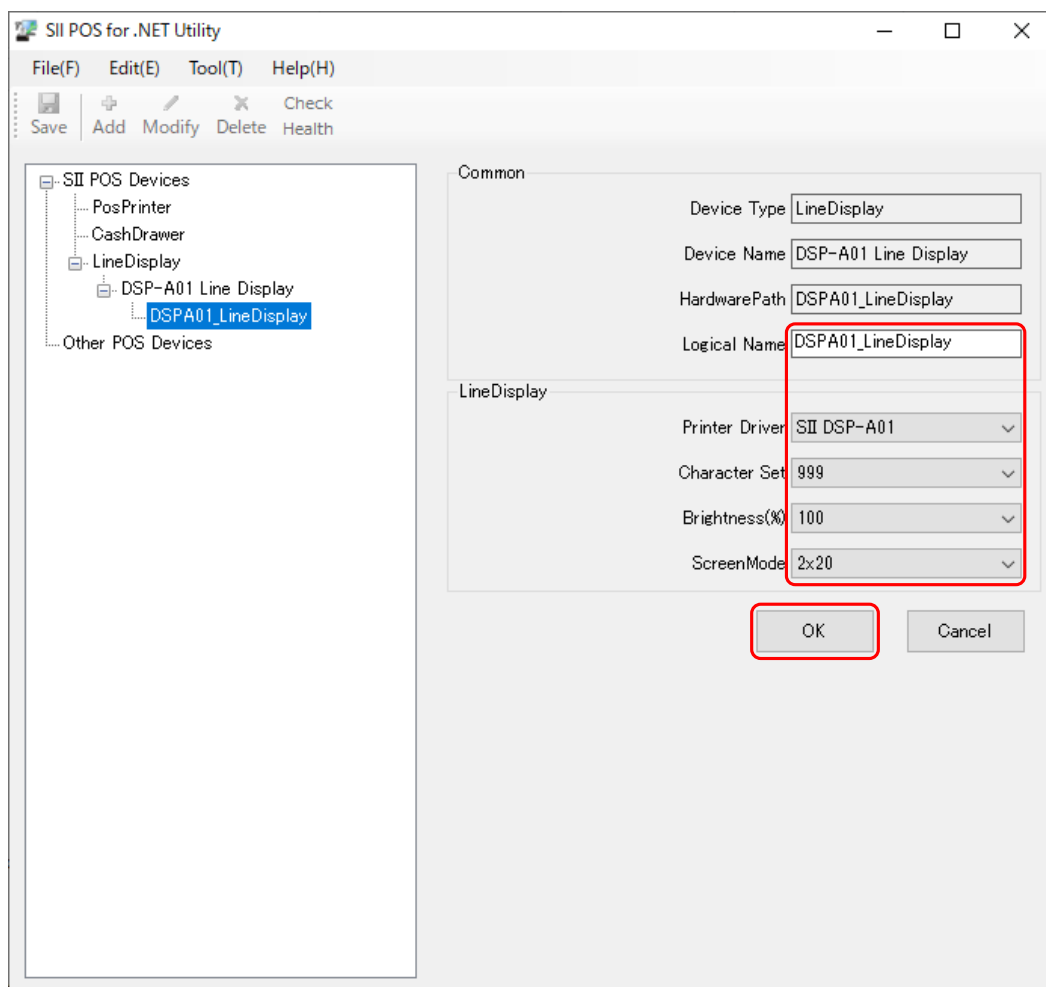
- (a) Select the logical name of device to change from "Device View".



- (b) Click the [Modify] button in "Tool Bar".



- (c) "Setting View" is displayed in editable state. Click the [OK] button after changing the contents.



- (d) Click the [Save] button in "Tool Bar".

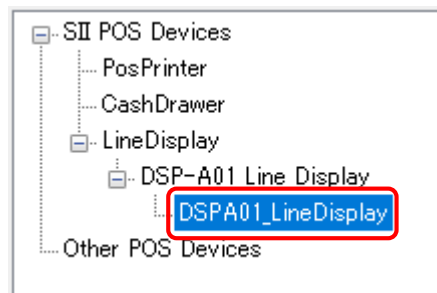
### 3.3.3 Deletion of Device

The added device can be deleted with the [Delete] button.  
Select the target logical name, and click the [Delete] button.

### 3.3.4 Device Interactive Test

In the configuration program, an interactive test can be performed on the device selected in "Device View".  
The procedure of the interactive test is described below.

- (a) Select the logical name of device for interactive test from "Device View".



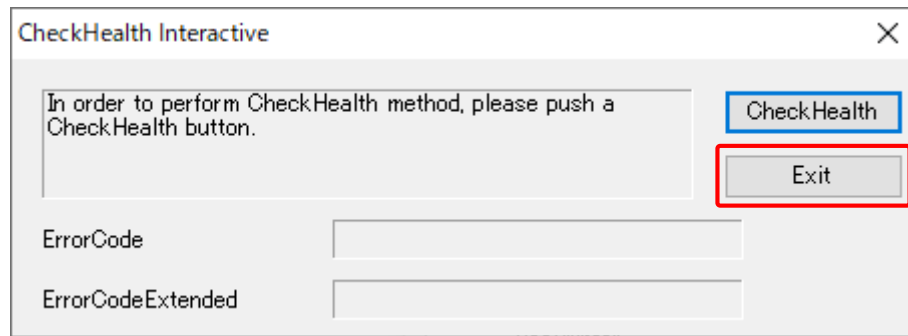
- (b) Click the [CheckHealth] button in "Tool Bar".



- (c) The preparation for the interactive test is started.

[When the preparation for the interactive test succeeded]

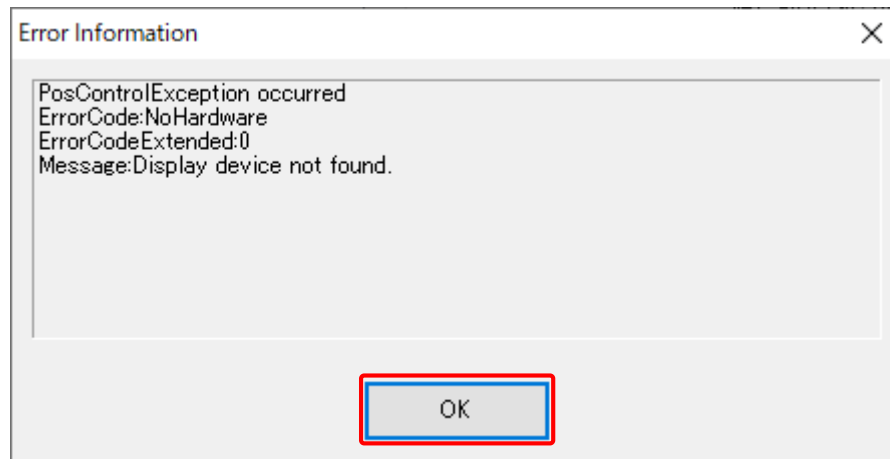
- (d) The CheckHealth Interactive dialogue to perform the interactive test is displayed.



To start the interactive test, click the [CheckHealth] button.  
To exit the interactive test, click the [Exit] button.

[When the preparation for the interactive test failed]

- (d) The Error Information dialogue is displayed.

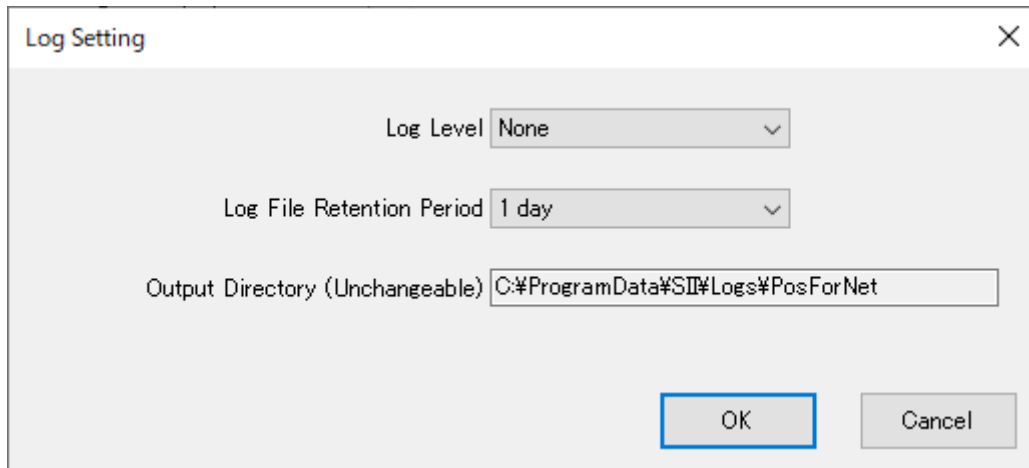


Confirm ErrorCode displayed in the dialogue. See "Appendix A Exceptions" for ErrorCode.  
Click the [OK] button after confirming ErrorCode.

### 3.3.5 Log Setting

In the configuration program, common log settings can be made for all devices.

Select [Tool] – [LogSetting] from "Menu Bar" to display the following window.

The image shows a 'Log Setting' dialog box with a title bar and a close button (X). It contains three settings: 'Log Level' set to 'None', 'Log File Retention Period' set to '1 day', and 'Output Directory (Unchangeable)' set to 'C:\ProgramData\SII\Logs\PosForNet'. At the bottom right are 'OK' and 'Cancel' buttons. The 'OK' button is highlighted with a blue border.

The level of the log and the contents to be output are as follows.

Item	Description (" " : Default)	
Log Level	None	No logs are output.
	Error	The following logs are output. • Error at execution
	Info	The following logs are output. • Error at execution • Highlighted event at execution
	Debug	The following logs are output. • Error at execution • Highlighted event at execution • More detailed information for debugging
Log File Retention Period	Select the retention period for log files. • 1 day • 3 days • 10 days • 30 days • 90 days  Log files past the retention period are deleted when logs are output. The actual retention period may be longer by 1 day at maximum. The maximum size of a log file is 32 MB. When the log file exceeds the maximum size, a new log file is created and stored up to the retention period.	
Output Directory (Unchangeable)	Log output directory. The log output directory and file name are as follows. Output Directory: <System Drive>\ProgramData\SII\Logs\PosForNet The output directory cannot be changed. File Name: <yyyyMMdd>.log However, when the log file exceeds the maximum size, the file name is changed to <yyyyMMdd_hhmmssff>.log, and a new <yyyyMMdd>.log is created.*1	

\*1: Meanings of the symbols used for the file name are as follows. Each value comes from System Clock of Windows.

yyyy : Year  
MM : Month  
dd : Day  
hh : Hour  
mm : Minute  
ss : Second  
fff : Millisecond

#### (1) Log setting procedure

The log setting procedure is described below.

- (a) Select [Tool] – [LogSetting] from "Menu Bar".
- (b) Select the log level to output from [Log Level].
- (c) Select the log file retention period from [Log File Retention Period], and click the [OK] button.
- (d) Click the [Save] button in the main screen. The log settings will be applied from the next **Open**.

---

## Chapter 4 Properties, Methods, and Events

---

This chapter describes properties, methods, and events implemented in this software.

### 4.1 Summary

(1) Common properties

Property Name	Type	Access	Availability Condition	Default
CapCompareFirmwareVersion	bool	R	Open	<i>false</i>
CapPowerReporting	PowerReporting	R	Open	<i>None</i>
CapStatisticsReporting	bool	R	Open	<i>false</i>
CapUpdateFirmware	bool	R	Open	<i>false</i>
CapUpdateStatistics	bool	R	Open	<i>false</i>
CheckHealthText	string	R	Open	<i>""</i>
Claimed	bool	R	Open	<i>false</i>
DeviceDescription	string	R	Open	"SII DSP-A01 Line Display"
DeviceEnabled	bool	R/W	Open & Claim	<i>false</i>
DeviceName	string	R	Open	"DSP-A01 Line Display"
FreezeEvents	bool	R/W	Open & Claim	<i>false</i>
PowerNotify	PowerNotification	R/W	Open	<i>Disabled</i>
PowerState	PowerState	R	Open	<i>Unknown</i>
ServiceObjectDescription	string	R	Open	"SII DSP-A01 LineDisplay Service Object, Copyright(C) 20xx Seiko Instruments Inc."
ServiceObjectVersion	Version	R	Open	1.12.x.x
State	ControlState	R	--	<i>Closed</i>
SynchronizingObject	System.ComponentModel.ISynchronizeInvoke	R/W	Open	Depends on the application.

(2) Specific Properties  
(When ScreenMode = 2 × 20, CharacterSet = 999)

Property Name	Type	Access	Availability Condition	Default
CapBrightness	bool	R	Open	true
CapCharacterSet	CharacterSetCapability	R	Open	Kanji
CapScreenMode	bool	R	Open	true
CharacterSet	int	R/W	Open, Claim, & Enable	999 <sup>*1</sup>
CharacterSetList	int[]	R	Open	{ 437, 737, 850, 852, 855, 857, 858, 860, 863, 865, 866, 932, 999, 1250, 1251, 1252, 1253, 1254 }
Columns	int	R	Open	The value of DeviceColumns
CursorColumn	int	R/W	Open	0
CursorRow	int	R/W	Open	0
CursorUpdate	bool	R/W	Open	true
DeviceBrightness	int	R/W	Open, Claim, & Enable	100 <sup>*1</sup>
DeviceColumns	int	R	Open	20
DeviceRows	int	R	Open	2
DeviceWindows	int	R	Open	0
Rows	int	R	Open	The value of DeviceRows
ScreenMode	int	R/W	Open & Claim	1 <sup>*1</sup>
ScreenModeList	DisplayScreenMode[]	R	Open	{ (20,2), (20,5), (40,2), (40,5), (40,8) }

\*1: Can be modified by the configuration program.

The following specific properties are provided but the operation is not supported.

Property Name	Type	Access	Availability Condition	Default
BlinkRate	int	R/W <sup>*1</sup>	Open	0
CapBitmap	bool	R	Open	false
CapBlink	DisplayBlink	R	Open	None
CapBlinkRate	bool	R	Open	false
CapCursorType	DisplayCursors	R	Open	None
CapCustomGlyph	bool	R	Open	false
CapDescriptors	bool	R	Open	false
CapICharWait	bool	R	Open	false
CapHMarquee	bool	R	Open	false
CapMapCharacterSet	bool	R	Open	false
CapReadBack	DisplayReadBack	R	Open	None
CapReverse	DisplayReverse	R	Open	None
CapVMarquee	bool	R	Open	false
CurrentWindow	int	R/W <sup>*1</sup>	Open	0
CursorType	DisplayCursors	R/W <sup>*1</sup>	Open	None



Property Name	Type	Access	Availability Condition	Default
CustomGlyphList	RangeOfCharacters	R	Open	{}
DeviceDescriptors	int	R	Open	0
GlyphHeight	int	R	Open	24
GlyphWidth	int	R	Open	24
InterCharacterWait	int	R/W*1	Open	0
MapCharacterSet	bool	R/W*1	Open	false
MarqueeFormat	DisplayMarqueeFormat	R/W*1	Open	Walk
MarqueeRepeatWait	int	R/W*1	Open	0
MarqueeType	DisplayMarqueeType	R/W*1	Open	None
MarqueeUnitWait	int	R/W*1	Open	0
MaximumX	int	R	Open	0
MaximumY	int	R	Open	0

\*1: When a value other than the default is set, *ErrorCode.Illegal*.

### (3) Common Methods

Method Name	Availability Condition
CheckHealth	Open, Claim, & Enable
Claim	Open
Close	Open
CompareFirmwareVersion	Open, Claim, & Enable
DirectIO	Open, Claim, & Enable
Open	-
Release	Open & Claim
ResetStatistic(string)	Open, Claim, & Enable
ResetStatistics()	Open, Claim, & Enable
ResetStatistics(StatisticCategories)	Open, Claim, & Enable
ResetStatistics(string[])	Open, Claim, & Enable
RetrieveStatistic(string)	Open, Claim, & Enable
RetrieveStatistics()	Open, Claim, & Enable
RetrieveStatistics(StatisticCategories)	Open, Claim, & Enable
RetrieveStatistics(string[])	Open, Claim, & Enable
UpdateFirmware	Open, Claim, & Enable
UpdateStatistic	Open, Claim, & Enable
UpdateStatistics(Statistic[])	Open, Claim, & Enable
UpdateStatistics(StatisticCategories, Object)	Open, Claim, & Enable

(4) Specific Methods

Method Name	Availability Condition
<b>ClearText</b>	Open, Claim, & Enable
<b>DisplayText</b>	Open, Claim, & Enable
<b>DisplayTextAt</b>	Open, Claim, & Enable

The following specific properties are provided but the operation is not supported.

Method Name	Availability Condition
<b>ClearDescriptors</b>	Open, Claim, & Enable
<b>CreateWindow</b>	Open, Claim, & Enable
<b>DefineGlyph</b>	Open, Claim, & Enable
<b>DestroyWindow</b>	Open, Claim, & Enable
<b>DisplayBitmap</b>	Open, Claim, & Enable
<b>ReadCharacterAtCursor</b>	Open, Claim, & Enable
<b>RefreshWindow</b>	Open, Claim, & Enable
<b>ScrollText</b>	Open, Claim, & Enable
<b>SetBitmap</b>	Open, Claim, & Enable
<b>SetDescriptor</b>	Open, Claim, & Enable

(5) Events

Events are not supported.

## 4.2 Data Characters and Escape Sequences

### (1) Escape Sequence Operated When Specified

Name	Data	Remarks
Display bitmap	ESC #B	Not supported.

### (2) Escape Sequence Operated When Displayed

Name	Data	Remarks
Reverse video	ESC rvC	Not supported.
Blink	ESC kC	Not supported.
Normal	ESC N	Not supported.

### 4.3 Common Properties

This section describes the details of the common properties.  
For details of the thrown exception errors, see "Appendix A Exceptions".

#### CapCompareFirmwareVersion Property

Type **bool**

Description Indicates the function that compares firmware version.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	This property is not supported.

This property is initialized to *false* by **Open**.

#### CapPowerReporting Property

Type **PowerReporting**

Description Identifies the reporting capabilities of the device.  
The following table shows the valid property values.

Value	Meaning
<i>PowerReporting.None</i>	This property is not supported.

This property is initialized to *PowerReporting.None* by **Open**.

#### CapStatisticsReporting Property

Type **bool**

Description Identifies the reporting capabilities of the device.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	This property is not supported.

This property is initialized to *false* by **Open**.

## CapUpdateFirmware Property

Type **bool**

Description Indicates whether the device supports firmware updating.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	This property is not supported.

This property is initialized to *false* by **Open**.

## CapUpdateStatistics Property

Type **bool**

Description Indicates the function that some or all device statistics can be reset.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	This property is not supported.

This property is initialized to *false* by **Open**.

## CheckHealthText Property

Type **string**

Description Holds the results of the immediately preceding call to the **CheckHealth**.  
The following examples show the results of diagnosis.

Method Parameter	Method Result	CheckHealthText
<i>HealthCheckLevel.External</i>	Success	"External HCheck: Successful"
	Fail	"External HCheck: Failure"
<i>HealthCheckLevel.Interactive</i> <sup>*1</sup>	Success	"Interactive HCheck: Successful"
	Fail	"Interactive HCheck: Failure"
<i>HealthCheckLevel.Internal</i>	Success	"Internal HCheck: Successful"
	Fail	"Internal HCheck: Failure"

<sup>\*1</sup>: In the case of *HealthCheckLevel.Interactive*, if the dialog box is closed without testing after execution, "Interactive HCheck: Canceled" is set.

This property is initialized to empty string by **Open**.

## Claimed Property

Type **bool**

Description Indicates whether the device is claimed for exclusive access.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	The device is released for sharing with other applications.
<i>true</i>	The exclusive access to the device is obtained.

This property is initialized to *false* by **Open**.

## DeviceDescription Property

Type **string**

Description Identifies devices and related information.  
This property depends on **DeviceName**.  
This property is initialized to the following values by **Open**.

DeviceName	Default
"DSP-A01 Line Display"	"SII DSP-A01 Line Display"

## DeviceEnabled Property R/W

Type **bool**

Description Selects Enable/Disable of the device.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	The device has been disabled. If changed to <i>false</i> , then the device is physically disabled.
<i>true</i>	The device has been placed in an operational state. If changed to <i>true</i> , then the device is brought to an operational state

The application must set this property to *true* before using the device.

If **State** is other than *ControlState.Idle*, **DeviceEnabled** cannot be changed from *true* to *false*.

This property is initialized to *false* by **Open**.

## DeviceName Property

Type **string**

Description Identifies devices and related information.  
This property is initialized to the following values by **Open**.

Display	Value
DSP-A01	"DSP-A01 Line Display"

## FreezeEvents Property R/W

Type **bool**

Description Selects whether to notify events.  
The following table shows the valid property values.

Value	Meaning
<i>false</i>	This property is not supported.

This property is initialized to *false* by **Open**.

## PowerNotify Property R/W

Type **PowerNotification**

Description Contains the type of power notification selection made by the application.  
The following table shows the valid property values.

Value	Meaning
<i>PowerNotification.Disabled</i>	This property is not supported.

This property is initialized to *PowerNotification.Disabled* by **Open**.

## PowerState Property

Type **PowerState**

Description Contains the current power condition of the device.  
The following table shows the valid property values.

Value	Meaning
<i>PowerState.Unknown</i>	This property is not supported.

This property is initialized to *PowerState.Unknown* by **Open**.

## ServiceObjectDescription Property

Type            **string**

Description    Contains a string for identifying the service object to this property.  
This property is initialized to the following values by **Open**.

DeviceName	Default
"DSP-A01 Line Display"	"SII DSP-A01 LineDisplay Service Object, Copyright (C) 20xx Seiko Instruments Inc."

## ServiceObjectVersion Property

Type            **Version**

Description    This property holds the Service Object version number.  
Version numbers consist of four integers; Major, Minor, Build, and Revision.  
The Major and Minor version numbers correspond to the UnifiedPOS version that the Service Object implements.  
When Build version is A, Revision version is B, this property is initialized to 1.12.A.B by **Open**.

## State Property

Type            **ControlState**

Description    Contains the current state of the Control.  
The following table shows the valid property values.

Value	Meaning
<i>ControlState.Closed</i>	The device is closed.
<i>ControlState.Idle</i>	The device is in a good state and is not busy.

This property is always readable.

This property is initialized to *ControlState.Idle* by **Open**.



## SynchronizingObject Property

Type	<b>System.ComponentModel.ISynchronizeInvoke</b>
Description	<p>Contains <b>ISynchronizeInvoke</b> class instance. The application can specify threads that events are notified using this properties.</p> <p>When <b>SynchronizingObject</b> is set to <i>null</i>, events are raised on a system thread owned by the Service Object.</p> <p>The application using a Windows form sets the <i>this</i> pointer of the <b>Form</b> class of the main form to <b>SynchronizationObject</b>, so that events are notified to the main application thread as required by the <b>Form</b> class.</p>

## 4.4 Specific Properties

This section describes the details of the specific properties.

For exception errors of specific properties that are not supported, see "Appendix A Exceptions".

### CapBrightness Property

Type            **bool**

Description    Indicates whether the brightness can be controlled.  
The following table shows the valid property values.

Value	Meaning
<i>true</i>	The brightness control is supported.

This property is initialized to *true* by **Open**.

### CapCharacterSet Property

Type            **CharacterSetCapability**

Description    Holds the character setting that can be displayed.  
This property has the following value.

Value	Meaning
<i>CharacterSetCapability.Kanji</i>	The character setting supports Code Page932, including ASCII characters 0x20 through 0x7F and the one-byte katakana characters 0xA1 through 0xDF. It also includes the Shift-JIS code characters defined in JIS 1st and 2nd levels.

This property is initialized to *CharacterSetCapability.Kanji* by **Open**.

### CapScreenMode Property

Type            **bool**

Description    Indicates whether the device can support changing the screen mode (for example, the number of text rows and columns).  
The following table shows the valid property values.

Value	Meaning
<i>true</i>	The screen mode changing is supported.

This property is initialized to *true* by **Open**.

## CharacterSet Property R/W

Type            **int**

Description    Holds the character set for displaying characters.  
One of the following values is set to this property.

Value	Meaning
437	Selects Code Page437 character set.
737	Selects Code Page737 character set.
850	Selects Code Page850 character set.
852	Selects Code Page852 character set.
855	Selects Code Page855 character set.
857	Selects Code Page857 character set.
858	Selects Code Page858 character set.
860	Selects Code Page860 character set.
863	Selects Code Page863 character set.
865	Selects Code Page865 character set.
866	Selects Code Page866 character set.
932	Selects the Katakana as the Code Page932 character set (Shift-JIS Code).
999	Sets the Windows ANSI characters.*1
1250	Selects Code Page1250 character set.
1251	Selects Code Page1251 character set.
1252	Selects Code Page1252 character set.*1
1253	Selects Code Page1253 character set.
1254	Selects Code Page1254 character set.

\*1: Windows ANSI character set is equal to Code Page1252 character set.

For this property, the default can be changed by setting of the configuration program.  
This property is initialized to the value of character set which is set in [Character Set] of the configuration program when the device is first enable.

## CharacterSetList Property

Type            **int[]**

Description    Holds the character set numbers supported by the printer as a numeric array.

{ 437, 737, 850, 852, 855, 857, 858, 860, 863, 865, 866, 932, 999, 1250, 1251, 1252, 1253, 1254 }

This property is initialized to the above value by **Open**.

## Columns Property

Type	<b>int</b>
Description	<p>Holds the number of columns for the current window.</p> <p>This property is the same as <b>DeviceColumns</b>.</p> <p>This property is initialized to the value of <b>DeviceColumns</b> by <b>Open</b>.</p>

## CursorColumn Property R/W

Type	<b>int</b>
Description	<p>Holds the column in the current window to which the next displayed character will be output.</p> <p>Legal values range from 0 through <b>Columns</b>. (See the description of <b>CursorColumn = Columns</b> in <b>DisplayText</b>.)</p> <p>This property is initialized to 0 by <b>Open</b> and <b>ClearText</b>.</p> <p>If <b>CursorUpdate</b> is <i>true</i>, it is also updated when <b>DisplayText</b> or <b>DisplayTextAt</b> is called.</p>

## CursorRow Property R/W

Type	<b>int</b>
Description	<p>Holds the row in the current window to which the next displayed character will be output.</p> <p>Legal values range from 0 through 1 less than <b>Rows</b>.</p> <p>This property is initialized to 0 by <b>Open</b> and <b>ClearText</b>.</p> <p>If <b>CursorUpdate</b> is <i>true</i>, it is also updated when <b>DisplayText</b> or <b>DisplayTextAt</b> is called.</p>

## CursorUpdate Property R/W

Type	<b>bool</b>
Description	<p><b>CursorRow</b> and <b>CursorColumn</b> will be updated to point to the character beyond the last character output when characters are displayed using the <b>DisplayText</b> or <b>DisplayTextAt</b> method.</p> <p>The following table shows the valid property values.</p>

Value	Meaning
<i>true</i>	<b>CursorRow</b> and <b>CursorColumn</b> will be updated to point to the character beyond the last character output when characters are displayed using the <b>DisplayText</b> or <b>DisplayTextAt</b> method.
<i>false</i>	The cursor properties will not be updated when characters are displayed.

This property is initialized to *true* by **Open**.

## DeviceBrightness Property R/W

Type            **int**

Description    Holds the device brightness value, expressed as a percentage between 0 and 100.  
The following table shows the valid property values.

Value	Meaning
0	0 : 0% (blank)
1 to 10	10: 10% intensity
11 to 20	20: 20% intensity
21 to 30	30: 30% intensity
31 to 40	40: 40% intensity
41 to 50	50: 50% intensity
51 to 60	60: 60% intensity
61 to 70	70: 70% intensity
71 to 80	80: 80% intensity
81 to 90	90: 90% intensity
91 to 100	100: 100% intensity

For this property, the default can be changed by setting of the configuration program.  
This property is initialized to the value of brightness which is set in [Brightness (%)] of the configuration program when the device is first enabled.

## DeviceColumns Property

Type            **int**

Description    Holds the number of columns on this device.

This property is initialized by **Open**.

This property is updated to one of the following values which is being set in [ScreenMode] of the configuration program by calling **ClaimDevice** immediately after **Open**. In addition, this property is updated when the index value of **ScreenMode** is changed.

ScreenMode	DeviceColumns
2×20	20
5×20	20
2×40	40
5×40	40
8×40	40

## DeviceRows Property

Type            **int**

Description    Holds the number of rows on this device.

This property is initialized by **Open**.

This property is updated to one of the following values which is being set in [ScreenMode] of the configuration program by calling **ClaimDevice** immediately after **Open**. In addition, this property is updated when the index value of **ScreenMode** is changed.

ScreenMode	DeviceRows
2×20	2
5×20	5
2×40	2
5×40	5
8×40	8

## DeviceWindows Property

Type            **int**

Description    Holds the maximum window number supported by this device.  
The following table shows the valid property values.

Value	Meaning
0	Only the device window is supported. No windows may be created.

This property is initialized to 0 by **Open**.

## Rows Property

Type            **int**

Description    Holds the number of rows for this window.  
This property is the same as **DeviceRows**.

This property is initialized to the same value of **DeviceRows** by **Open**.

## ScreenMode Property R/W

Type            **int**

Description    Contains the screen mode value of the device.  
The values of this property are index values contained in **ScreenModeList**.  
The following table shows an example of index values.

Value	Meaning
0	The value of [ScreenMode] in the configuration program
1	2×20
2	5×20
3	2×40
4	5×40
5	8×40

This property can only be updated when the device is opened and claimed, but not enabled.

For this property, the default screen mode can be changed by setting of the configuration program.

This property is initialized to the value which is set in [ScreenMode] of the configuration program by **Open**.

## ScreenModeList Property

Type            **DisplayScreenMode[]**

Description    Contains the comma-delimited list of row-column pairs that are supported by the device.  
  
This property is initialized to { (20,2), (20,5), (40,2), (40,5), (40,8) } by **Open**.

## 4.5 Common Methods

This section describes the details of the common methods.  
For details of the thrown exception errors, see "Appendix A Exceptions".

### CheckHealth Method

Syntax `string CheckHealth(HealthCheckLevel level);`

Parameter	Meaning
<i>level</i>	Specifies the type of health check to be executed on the device.

· Values of *level*

Value	Meaning
<i>HealthCheckLevel.External</i>	Executes a test for indication of Display after confirming communication to Display using the device. This method is failed when exclusive access has being attempting by another applications.
<i>HealthCheckLevel.Interactive</i>	Executes an interactive test of the device. Displays a modal dialog box to execute a complete test using the device and display results. This method is failed when exclusive access has being attempting by another application.
<i>HealthCheckLevel.Internal</i>	Execute a health check without using the device physically.

Description Tests the status of the device.  
A text description of the results of this method is placed in **CheckHealthText**.  
**CheckHealth** is always executed synchronously.

### Claim Method

Syntax `void Claim(int timeout);`

Parameter	Meaning
<i>timeout</i>	Specifies the maximum waiting time (in milliseconds) for acquisition of exclusive access. If it is 0, the method returns the result immediately even if exclusive access of the device cannot be obtained. If <i>WaitForever</i> (-1) is set, the method waits until exclusive access is obtained.

Description Requests exclusive access to the device.  
The LineDisplay device cannot be used until the exclusive access is obtained.  
When it is successful, **Claimed** is set to *true*.  
When the power is OFF or the cable is not connected, **Claim** is not available.



## Close Method

Syntax      **void Close();**

Description      Releases the device and its resources.  
If **DeviceEnabled** is *true*, the device is first disabled.  
If **Claimed** is *true*, exclusive access to the device is first released.  
Do not execute this while the event is in progress (or in the event handler).

## CompareFirmwareVersion Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **CompareFirmwareResult CompareFirmwareVersion(string firmwareFileName);**

## DirectIO Method

Syntax      **DirectIOData DirectIO(int command, int data, object obj);**

Parameter	Meaning
<i>command</i>	Command number. Specific values assigned by the Service Object.
<i>data</i>	Additional numeric data. Specific values vary by command number and Service Object.
<i>obj</i>	Additional data provided by the Service Object. The value varies depending on the command number and what the Service Object sends.

Description      The following functions are supported:  
· Binary data transmission  
· Specified file sending  
**DirectIO** is always executed synchronously.

- **Binary data transmission**

Any Display command can be sent.

Parameter	Description
<i>command</i>	601
<i>data</i>	<i>null</i>
<i>obj</i>	<p>IN Transmission data</p> <p>A string is specified in hexadecimal.</p> <p>Example: When US "LD" 31h 08h 00h 00h 00h "TXW" 31h 32h 33h 34h 35h is transmitted by the Display command "Input Text Data" "1F4C4431080000005458573132333435"</p> <p>See "DSP-A01 SERIES CUSTOMER DISPLAY TECHNICAL REFERENCE" for details of Display command.</p>

- **Specified file sending**

A specified file can be sent.

Parameter	Description
<i>command</i>	602
<i>data</i>	<i>null</i>
<i>obj</i>	<p>IN Sending file name</p> <p>A full path or relative path is specified as a character string.</p> <p>The file formats that can be specified are BIN, XML, JPEG and PNG.</p> <p>The file size that can be sent varies depending on the remaining memory capacity of Display's user area and the file format.</p> <p>See "DSP-A01 SERIES CUSTOMER DISPLAY TECHNICAL REFERENCE" for details.</p> <p>Example: "C:\Temp\binary.bin"</p> <p>"C:\Temp\Template.xml"</p> <p>"C:\Temp\Image.jpg"</p>

## Open Method

Syntax      **void Open();**

Description      Opens the device.

When **Open** is successful, the common properties and other class-specific properties are initialized.

## Release Method

Syntax      **void Release();**

Description      Releases exclusive access to the device.  
If **DeviceEnabled** is *true*, and the device is an exclusive-use device, then the device is disabled.  
Do not execute this while the event is in progress (or in the event handler).

## ResetStatistic(string) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void ResetStatistic(string *statistic*);**

## ResetStatistics() Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void ResetStatistics();**

## ResetStatistics(StatisticCategories) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void ResetStatistics(StatisticCategories *statistics*);**

## ResetStatistics(string[]) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void ResetStatistics(string[] *statistics*);**

## RetrieveStatistic(string) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **string RetrieveStatistic(string *statistic*);**

## RetrieveStatistics() Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **string RetrieveStatistics();**

## RetrieveStatistics(StatisticCategories) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **string** RetrieveStatistics(**StatisticCategories** *statistics*);

## RetrieveStatistics(string[]) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **string** RetrieveStatistics(**string[]** *statistics*);

## UpdateFirmware Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void** UpdateFirmware(**string** *firmwareFileName*);

## UpdateStatistic Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void** UpdateStatistic(**string** *name*, **object** *value*);

## UpdateStatistics(Statistic[]) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void** UpdateStatistics(**Statistic[]** *statistics*);

## UpdateStatistics(StatisticCategories, Object) Method

This method is not supported. For the thrown exception errors, see "Appendix A Exceptions".

Syntax      **void** UpdateStatistics(**StatisticCategories** *statistics*, **object** *value*);

## 4.6 Specific Methods

This section describes the details of the specific methods.

For exception errors of specific methods that are not supported, see "Appendix A Exceptions".

### ClearText Method

Syntax      **void ClearText();**

Remarks      Clears the current window to blanks. **CursorRow** and **CursorColumn** are set to 0. Clears all bitmaps displayed in the window.

### DisplayText Method

Syntax      **void DisplayText(string *data*, int *attribute*);**

Parameter	Meaning
<i>data</i>	The string of characters to display. It consists of displayable characters and escapes sequences, line feeds (LF) and carriage returns (CR).
<i>attribute</i>	Supports only <i>Normal</i> .

Description      The characters in *data* are processed beginning at the location specified by **CursorRow** and **CursorColumn**.

Character processing continues to the next row when the end of a window row is reached. If the end of the window is reached with additional characters to be processed, then the window is scrolled upward by 1 row.

If **CursorUpdate** is *true*, **CursorRow** and **CursorColumn** are updated to point to the character position following the last character of *data*.

Scrolling will not occur when the last character of *data* is placed at the end of a row. In this case, If **CursorUpdate** is *true*, **CursorRow** is set to the row containing the last character, and **CursorColumn** is set to **Columns** (that is, to one more than the final character of the row).

This stipulation ensures that Display does not scroll when a character is written into its last position. Instead, the Control Object will wait until another character is written before scrolling the window.

This method updates the window and view port immediately.

The values and meanings of special characters within *data* are as follows.

Symbol	Operation
LF	Changes the next character's output position to the beginning of the next row. Scroll the window if the current row is the last row of the window.
CR	Changes the next character's output position to the beginning of the current row.

## DisplayTextAt Method

Syntax      **void DisplayTextAt(int row, int column, string data, int attribute);**

Parameter	Meaning
<i>row</i>	Specifies the start row for the text.
<i>column</i>	Specifies the start column for the text.
<i>data</i>	Specifies the string of characters to display. It consists of displayable characters and escapes sequences, line feeds (LF) and carriage returns (CR).
<i>attribute</i>	Supports only <i>Normal</i> .

Description      The characters in *data* are processed beginning at the window location specified by *row* and *column*.

Character processing continues to the next row when the end of a window row is reached. If the end of the window is reached with additional characters to be processed, then the window is scrolled upward by 1 row.

If **CursorUpdate** is *true*, then **CursorRow** and **CursorColumn** are updated to point to the character position following the last character of *data*.

Scrolling will not occur when the last character of *data* is placed at the end of a row. In this case, If **CursorUpdate** is *true*, then **CursorRow** is set to the row containing the last character, and **CursorColumn** is set to **Columns** (that is, to one more than the final character of the row).

This stipulation ensures that Display does not scroll when a character is written into its last position. Instead, the Control Object will wait until another character is written before scrolling the window.

The values and meanings of special characters within *data* are as follows.

Symbol	Operation
LF	Changes the next character's output position to the beginning of the next row. Scroll the window if the current row is the last row of the window.
CR	Changes the next character's output position to the beginning of the current row.

---

## Appendix A Exceptions

---

### A.1 Exception Error List

(1) Property

ErrorCode	ErrorCode Extended	Meaning
Disabled	0	Not enabled. Call this after setting <b>DeviceEnabled</b> to <i>true</i> .
Illegal	0	This property is not supported. Parameter has an error.
NotClaimed	0	Exclusive access is not available. Call <b>Claim</b> to gain exclusive access.

(2) Method

Method	ErrorCode	ErrorCode Extended	Meaning
ClearDescriptors CompareFirmwareVersion CreateWindow DefineGlyph DestroyWindow DisplayBitmap ReadCharacterAtCursor RefreshWindow ResetStatistic(s) RetrieveStatistic(s) ScrollText SetBitmap SetDescriptor UpdateFirmware UpdateStatistic(s)	Illegal	0	This method is not supported.
CheckHealth	Busy	0	Cannot perform while output is in progress or an error occurs.
	Disabled	0	Not enabled. Call this after setting <b>DeviceEnabled</b> to <i>true</i> .
	Illegal	0	Parameter has an error.
	NotClaimed	0	Exclusive access is not available. Call <b>Claim</b> to gain exclusive access.

Method	ErrorCode	ErrorCode Extended	Meaning
<b>Claim</b>	Claimed	0	Attempt was made to access a device that is exclusively accessed by another process.
	Failure	0	Communication with Display failed.
	Illegal	0	Parameter has an error.
	NoHardware	0	Display is powered off or the cable is not connected.
	Timeout	0	Another application has exclusive access to the device and the <i>timeout</i> (in milliseconds) has elapsed before the device is released. Or the device did not become available even though the <i>timeout</i> (in milliseconds) has elapsed.
<b>Close</b>	Busy	0	<b>State</b> is set to <i>ControlState.Busy</i> . This means that the device is busy and cannot be stopped.
	Closed	0	The device is already closed.
<b>Open</b>	Illegal	0	The device is already open.
<b>ClearText DirectIO DisplayText DisplayTextAt</b>	Busy	0	Cannot perform while output is in progress or an error occurs.
	Disabled	0	Not enabled. Call this after setting <b>DeviceEnabled</b> to <i>true</i> .
	Failure	0	Communication with Display failed.
	Illegal	0	Parameter has an error.
	NoHardware	0	The display is powered off or the cable is not connected.
	NotClaimed	0	Exclusive access is not available. Call <b>Claim</b> to gain exclusive access.
	Timeout	0	A head temperature error occurred.
<b>Release</b>	NotClaimed	0	The device is not exclusive.