

Windows SDK for Sanei Layout Designer2 User's Manual

March 8, 2023 Rev1.0.2

This manual provides design guideline information that customers need to build applications regarding the Windows SDK for Sanei Layout Designer2.



History of revision of this manual

Revision	Date	Description of revision
Rev1.0.0	Jun 09, 2021	Release of 1st edition (SDK Version1.00, EngineVer1.03)
Rev1.0.1	Sep 30, 2022	Changed OFFSET_DAY_MIN, OFFSET_MONTH_MIN, and OFFSET_YEAR_MIN constants from 1 to 0. "3.31 AlternativeFontNameSet" has been added.
Rev1.0.2	Mar 8, 2023	Supported standalone macros.

Caution

- Windows SDK for Sanei LayoutDesigner2 is a copyrighted work of Sanei Electric Co., Ltd. (hereinafter referred to as Sanei Electric). Copyright and other rights concerning this product belong to Sanei Electric.
- Sanei Electric grants the user the right to use Windows SDK for Sanei LayoutDesigner2 (free to copy and distribute) for the purpose of using Sanei Electric products that are compatible with Windows SDK for Sanei LayoutDesigner2.
- Sanei Electric does not guarantee the absence of defects in the Windows SDK for Sanei LayoutDesigner2 and is not liable for any damages resulting from the use of the information contained in this manual.
- Sanei Electric shall never be liable under any circumstances for any direct or indirect loss or damage in connection with the use of Windows SDK for Sanei LayoutDesigner2.
- Users cannot directly or indirectly export all or part of the Windows SDK for Sanei LayoutDesigner2 without obtaining necessary permission from the Japanese government or the government of the relevant country.

Sanei Electric Co., Ltd. 2021

Unauthorized reproduction prohibited.

The contents of this document are subject to change without notice.

Windows is a trademark of the US Microsoft Corporation in the United States, Japan, and other countries.

They are trademarks or registered trademarks, and they shall be used under license.

Other product names and company names are trademarks or registered trademarks of the respective companies.

1. Introduction.....	6
1.1. Operating Environment.....	6
1.2. Related Software.....	6
1.3. How to Implement the Class Library.....	6
2. LayoutDesigner2_SDK class.....	9
2.1. Global Area.....	10
2.2. Setting file.....	10
2.3. About standalone macros.....	10
3. API details.....	12
3.1. GetApiVersion.....	13
3.2. InitSdk.....	13
3.3. NewSopFile.....	13
3.4. LoadSopFile.....	14
3.5. SaveSopFile.....	14
3.6. PrintSopFile.....	14
3.7. PreviewSoPFile.....	15
3.8. InitSerial.....	15
3.9. SetSerialNumber.....	16
3.10. SetDateTime.....	17
3.11. GetObjectList.....	17
3.12. DelObjectData.....	18
3.13. GetTextData.....	18
3.14. SetTextData.....	19
3.15. GetPTextData.....	20
3.16. SetPTextData.....	21
3.17. GetImageData.....	22
3.18. SetImageData.....	22
3.19. GetLineData.....	23
3.20. SetLineData.....	23
3.21. GetRectData.....	24
3.22. SetRectData.....	24
3.23. GetFillData.....	25
3.24. SetFillData.....	25
3.25. GetBarcode1dData.....	26
3.26. SetBarcode1dData.....	27
3.27. GetBarcode2dData.....	28
3.28. SetBarcode2dData.....	29
3.29. GetMacroData.....	30
3.30. SetMacroData.....	30
3.31. AlternativeFontNameSet.....	30
3.32. InitStandaloneMacro.....	31
3.33. GetStandaloneMacroData.....	31
3.34. SetStandaloneMacroData.....	31
3.35. GetStandaloneMacroTitle.....	32
3.36. SetStandaloneMacroTitle.....	32
3.37. GetStandaloneMacroCount.....	32
3.38. LoadStandaloneMacro.....	33
3.39. SaveStandaloneMacro.....	33
3.40. RegistStandaloneMacro.....	33

4. Constant definition 34

5. Commands that can be used in document macros 40

6. Commands that can be used in standalone macros..... 40

APPENDIX: SAMPLE PROGRAMS 43

1. Introduction

Windows SDK for Sanei Layout Designer2 is an assistive library that enables mutual editing by API based on the layout file (SOP file) of the Layout Designer2 software main body.

This class library file consists of the following library files (DLL files).

Class library file	LayoutDesigner2_SDK.dll
Barcode library	SaneiBarcodeLibrary.dll
2D Code Library	2d_Code_Dll.dll
Communication Library	Sk1_Api_dll.dll

1.1. Operating Environment

OS: Microsoft Windows 7 SP1
 Microsoft Windows 8
 Microsoft Windows 10

Supported printer models:

SK1-21/31/41 Series
SK1-21H/31H Series
SK1-211/311 Series
SD3-21/22 Series
SK4-21/31 Series
SM4-21/31 Series
SM1-21 Series
BLM-80 Series
SM2-41 Series
BL2-58 Series

Installation of .NET Framework 4.0 or higher is required to use this SDK.

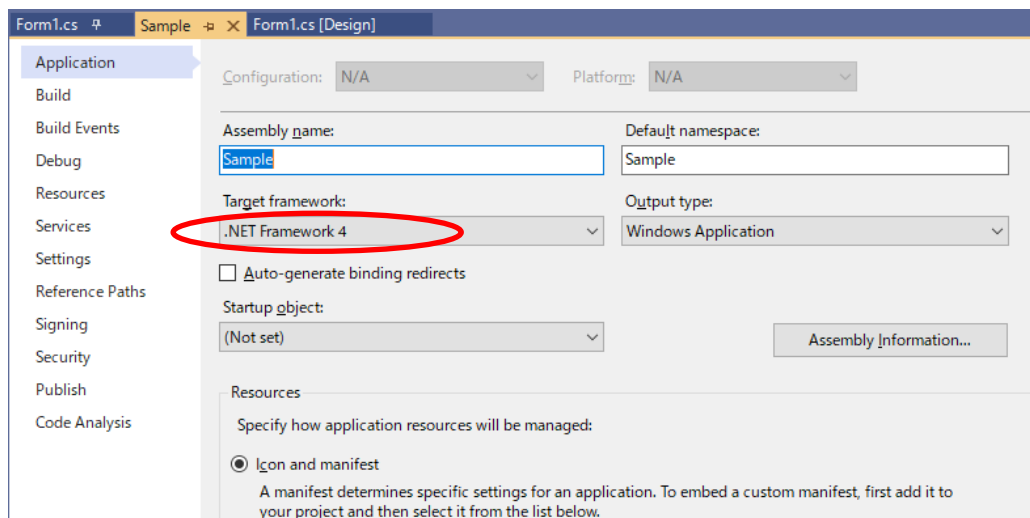
1.2. Related Software

There is no related software in this version.

1.3. How to Implement the Class Library

- (1) Applications that implement the Sanei Layout Designer2 SDK set the target framework to NET Framework 4.0 or higher.

You can view the target framework from the project properties.

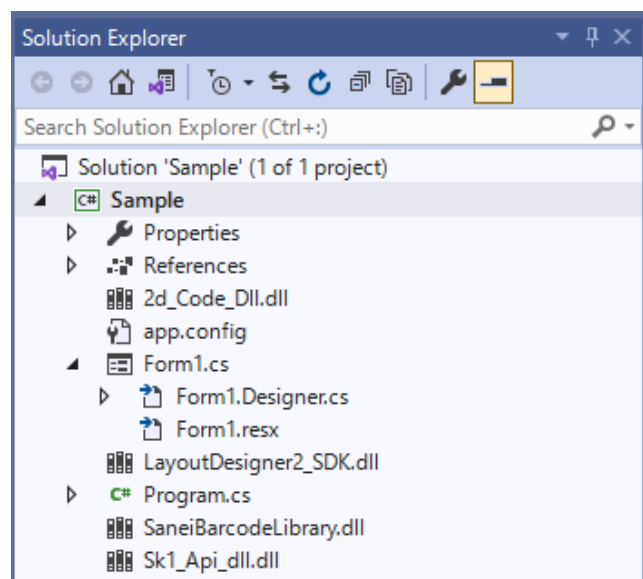


(2) Add the following DLL to the project folders of the application.

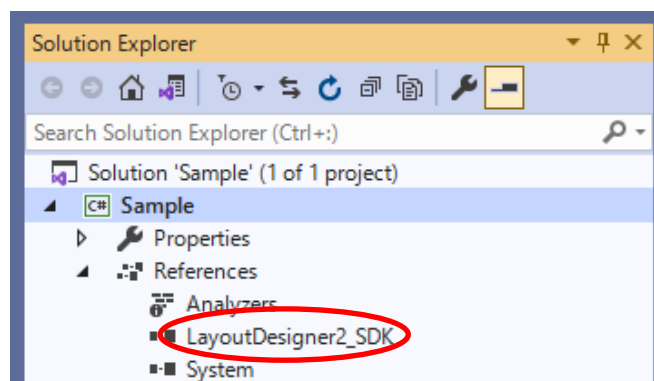
LayoutDesigner2_SDK.dll	Class library file
SaneiBarcodeLibrary.dll	Barcode library
2d_Code_DLL.dll	2D barcode library
Sk1_Api_dll.dll	Communication library

Name	Date modified	Type	Size
Properties	5/31/2021 3:35 PM	File folder	
obj	5/31/2021 3:35 PM	File folder	
bin	5/31/2021 3:41 PM	File folder	
app.config	5/31/2021 3:46 PM	XML Configuratio...	1 KB
Program.cs	5/31/2021 3:35 PM	Visual C# Source f...	1 KB
Form1.Designer.cs	5/31/2021 3:40 PM	Visual C# Source f...	3 KB
Form1.cs	5/31/2021 3:46 PM	Visual C# Source f...	1 KB
Sample.csproj	5/31/2021 4:39 PM	Visual C# Project f...	4 KB
Form1.resx	5/31/2021 3:40 PM	Microsoft .NET M...	6 KB
Sk1_Api_dll.dll	6/30/2015 11:51 AM	Application exten...	59 KB
SaneiBarcodeLibrary.dll	5/18/2021 1:53 PM	Application exten...	2,239 KB
LayoutDesigner2_SDK.dll	5/21/2021 2:13 PM	Application exten...	2,440 KB
2d_Code_DLL.dll	9/12/2019 9:27 AM	Application exten...	202 KB

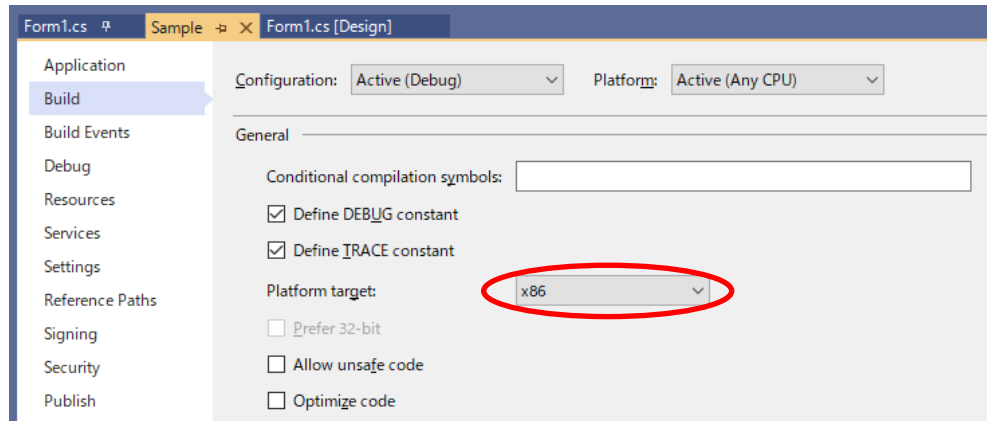
(3) Add the above four DLL files in "Add from Solution Explorer -> Existing Items".



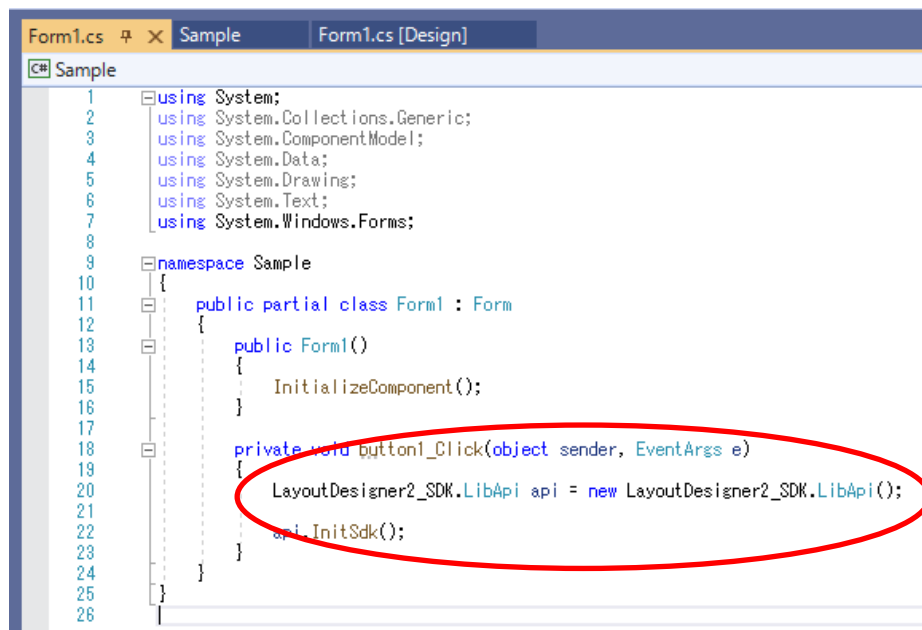
(4) Add LayoutDesigner2_SDK.dll in Add -> Ref from Solution Explorer.



(5) Set the Platform target for the project to "x86".



(6) Declare the class of SDK to the source code.

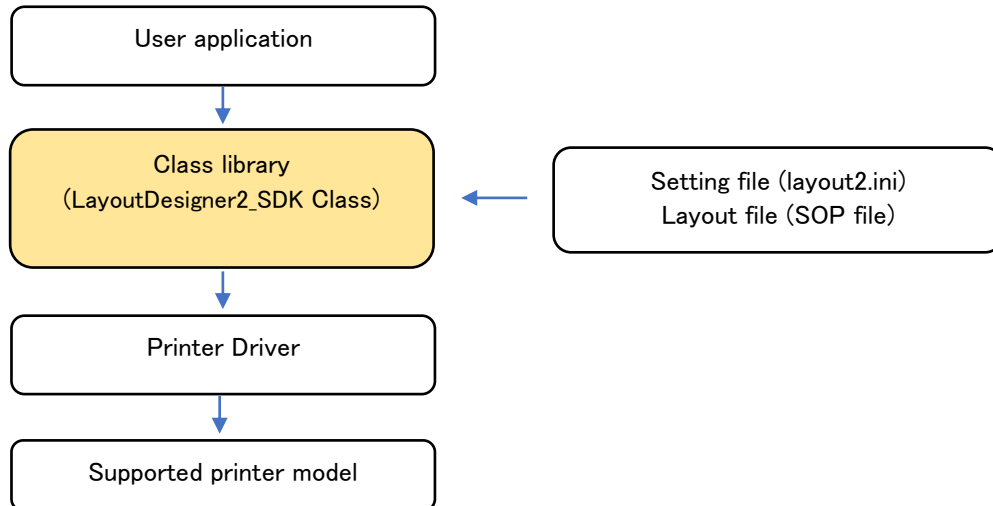


This completes the implementation and allows the `LayoutDesigner2_SDK` class to be used.

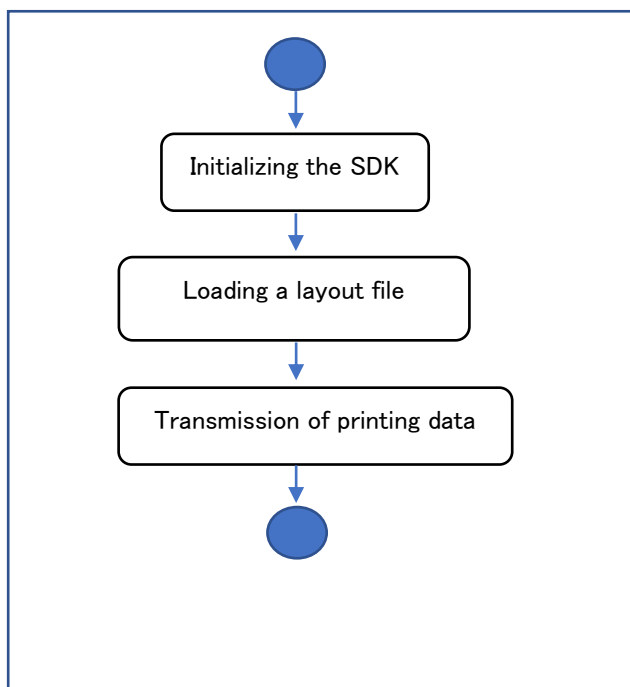
2. LayoutDesigner2_SDK class

The overall configuration using the LayoutDesigner2_SDK class and the process until printing are shown below.

● Overall configuration



● Process to print the layout file



Sample code image:

```
//SDK initialization.
ret = api.InitSdk();
//Initialization failure?
if (ret == false)
{
    return;
}

//File load.
int file_size = 0;
ret = api.LoadSopFile(@"sample.sop", ref file_size);
//Load failed?
if (ret == false)
{
    return;
}

//Printing.
int copies = 1;
api.PrintSopFile(@"SANEI SK1-31S-UNI-JP", copies);
```

2.1. Global Area

The global regions that can be referenced are.

`LibApi.ResultErrorCode` Error values by each API are stored.
For error values, refer to the error constant definition list in Chapter 4.

2.2. Setting file

The setting file refers to the Layout Designer2 preference file "layout2.ini".
The storage location of this file is located directly under the DLL folders.
In your app, place this configuration file in the same folders as LayoutDesigner2_SDK.dll.

Configuration of the preference file

`image-print=1` Set whether to print image data.

0: Do not print anything other than the printer text.
1: Print everything.

`auto-save=0` Sets whether the SOP file is automatically saved each time printing is performed.

0: The SOP file is not saved automatically.
1: The SOP file is automatically saved.

For layout data using a serial number, setting this setting to "1" automatically saves the updated data of the serial number each time it is printed.

`compress-print=1` Set whether to perform compression printing.

0: Compression printing is not performed.
1: Compression printing is performed.

2.3. About standalone macros

A standalone macro is a mechanism for registering in advance the printing process you want the printer to perform, and calling it with a single button.
Since it can handle variable data such as dates and counters, it is possible to perform print processing that could not be done without a host until now.
Up to 3 standalone macros can be registered and selected when used.

Background : Draws the SOP file that exists in the memory when registering the standalone macro and makes it the background.
Since the standalone macro itself has no association with the SOP file, it is necessary to read the SOP file in advance.

Date : You can print the date and embed the date in the QR code.
Since the date is obtained from the Internet, please set it to connect to the router in advance.
The date will not be printed if the router cannot be connected.

Counter : You can print counters.
The counter is stored in non-volatile memory, so even if the power is turned off and printing is interrupted, printing can continue from the next time.

- Variable : You can use 8 string variables that can be embedded in the QR code.
You can make the macro easier to read by setting the strings you want to change as variables.
- QR code : You can embed dates, counters and variables inside your QR code.
Unlike QR codes in SOP files, standalone macro QR codes generate print data just before printing, so variable data can be handled.
- Display name : This is the display name when the standalone macro is selected.
This name will appear on the printer display.
- File : The file name of the standalone macro is "standalone_macro.dat".
The folder will be the same folder you use to read and write SOP files.
Therefore, it is necessary to read and write the SOP file first before dealing with the standalone macro.

3. API details

The function list for the LayoutDesigner2_SDK class is as follows:

Category	API	Function
Get version	GetApiVersion	Gets the SDK and internal engine version values.
Initialization	InitSdk	Initialize this class.
SOP file	NewSopFile	Create a new SOP file.
	LoadSopFile	Load SOP file.
	SaveSopFile	Save the SOP file.
	PrintSopFile	Print the data in the current SOP file.
	PreviewSopFile	Creates a preview image of the data in the current SOP file.
Serial number	InitSerial	Initialize the serial number.
	SetSerialNumber	Set serial number.
Date/Time	SetDateTime	Set the date or time.
Object Editing	GetObjectList	Get object list
	DelObjectData	Delete object data.
	GetTextData	Get the text object data.
	SetTextData	Sets the text object data.
	GetPTextData	Get the printer text object data.
	SetPTextData	Sets the printer text object data.
	GetImageData	Get the image object data.
	SetImageData	Sets the image object data.
	GetLineData	Get the straight line object data.
	SetLineData	Sets the straight line object data.
	GetRectData	Get the rectangle object data.
	SetRectData	Sets the rectangle object data.
	GetFillData	Get the fill object data.
	SetFillData	Sets the fill object data.
	GetBarcode1dData	Get the object data of the 1D barcode.
	SetBarcode1dData	Sets the object data of the 1D barcode.
	GetBarcode2dData	Get the object data of the 2D barcode.
	SetBarcode2dData	Sets the object data of the 2D barcode.
	GetMacroData	Get the document macro object data.
	SetMacroData	Sets the document macro object data.
Standalone macro editing.	InitStandaloneMacro	Initialize a standalone macro.
	GetStandaloneMacroData	Get the standalone macro data.
	SetStandaloneMacroData	Sets the standalone macro data.
	GetStandaloneMacroTitle	Get the display name of a standalone macro.
	SetStandaloneMacroTitle	Sets the display name of the standalone macro.
	GetStandaloneMacroCount	Get the number of standalone macro data.
	LoadStandaloneMacro	Load a standalone macro file.
	SaveStandaloneMacro	Save a standalone macro file.
Others	RegistStandaloneMacro	Register the standalone macro with the printer.
	AlternativeFontNameSet	Sets an alternative font name if the specified font name does not exist.

Note:

For the arguments of each API, refer to the constant definitions in Chapter 4

3.1. GetApiVersion

Gets the version value of the SDK and internal engine.

Declaration:

```
bool    LayoutDesigner2_SDK.LibApi.GetApiVersion(ref int sdk_ver, ref int eng_ver )
```

Argument:

int	sdk_ver	SDK version value
int	eng_ver	Internal engine version value

Return value:

true	Normal end
false	Error

Note:

A version value of 1.02 is returned as 102.

The version value of the internal engine returns the version of the layout designer 2 main unit.

3.2. InitSdk

Initialize this class.

Declaration:

```
bool    LayoutDesigner.LibApi.InitSdk()
```

Argument: None

Return value:

true	Normal end
false	Error

3.3. NewSopFile

Create a new SOP file.

Declaration:

```
bool    LayoutDesigner.LibApi.NewSopFile(int printer, int width, int height, int top, int left)
```

Argument:

int	printer	Printer model number
int	width	Print width (unit: dot pitch (1/8 mm))
int	height	Print height (unit: dot pitch (1/8 mm))
int	top	Top margin (unit is dot pitch (1/8 mm))
int	left	Left margin (unit is dot pitch (1/8 mm))

Return value:

true	Normal end
false	Error

3.4. LoadSopFile

Load SOP file.

Declaration:

```
bool    LayoutDesigner.LibApi.LoadSopFile(string file, ref int size)
```

Argument:

string	file	File name of the SOP file
int	size	File size of the SOP file

Return value:

true	Normal end
false	Error

3.5. SaveSopFile

Save the SOP file.

Declaration:

```
bool    LayoutDesigner.LibApi.SaveSopFile(string file)
```

Argument:

string	file	File name of the SOP file
--------	------	---------------------------

Return value:

true	Normal end
false	Error

3.6. PrintSopFile

Print the data in the current SOP file.

Declaration:

```
bool    LayoutDesigner.LibApi.PrintSopFile(string printer, int num)
```

Argument:

string	printer	Name of the printer driver
int	num	Number of copies printed

Return value:

true	Normal end
false	Error

3.7. PreviewSoPFile

Creates a preview image of the data in the current SOP file.

Declaration:

bool LayoutDesigner.LibApi.PreviewSoPFile(string file)

Argument:

string	file	Name of the file of the image to be previewed. Creating an image format in "PNG" format.
--------	------	---

Return value:

true	Normal end
false	Error

3.8. InitSerial

Initialize the serial number.

Declaration:

bool LayoutDesigner.LibApi.InitSerial()

Argument: None

Return value:

true	Normal end
false	Error

3.9. SetSerialNumber

Set serial number.

Declaration:

```
bool LayoutDesigner.LibApi.SetSerialNumber(int SelectData, int SerialNumberType, int SerialIncDec,
                                           int ObjectIncDec, int Counter, string SerialFormat, int CounterInit, int CounterMax)
```

Argument:

int	SelectData	Specifying object types 0: Text system 1: Barcode system
int	SerialNumberType	(Serial number) type specification 0: Decimal number
int	SerialIncDec	(Serial number) Serial number increase/decrease specification 0: Sequential number increase 1: Sequential decrease
int	ObjectIncDec	Specifying the number of prints using the same serial number
int	Counter	To specify the number of increments/decrements
string	SerialFormat	Serial number format Write in ToString method format of .NET Framework
int	CounterInit	To specify the initial count number
int	CounterMax	Specifying the maximum count number

Return value:

true	Normal end
false	Error

Note:

If you want the ToString method format to be a decimal number, please refer to the following.

Format specifier	Name	Description	Example
D or d	Decimal number	Supports integer types only The precision specifier is the minimum number of digits Zeros are inserted on the left if the minimum number of digits is not reached	If the number was 1234 Format:"D" Print result:"1234" Format:"D6" Print result:"001234" If the number was -1234 Format:"D5" Print result:"-01234"

3.10. SetDateTime

Set the date or time.

Declaration:

```
bool LayoutDesigner.LibApi.SetDateTime(int DateTimeSelect, int FormatType, int Offset1, int Offset2)
```

Argument:

int	DateTimeSelect	Select date or time 0: Date 1: Time
int	FormatType	Specification of the format Date format number: 0 to 24 Format number of the time: 0 to 19
int	Offset1	Offset type specification Date format 0: 「Day」 1: 「Month」 2: 「Year」 Time format Time offset
int	Offset2	Specifying the offset range Date format Day offset: 0 to 30 Month offset: 0 to 11 Year offset: 0 to 98 Time format Minute offset

Return value:

true	Normal end
false	Error

Note:

If you want no offset, set the offset to 0.

For times, negative offsets are allowed.

3.11. GetObjectList

Get object list

Declaration:

```
bool LayoutDesigner.LibApi.GetObjectList(ref string[] object_list, ref int object_count)
```

Argument:

string[]	object_list	Object list
int	object_count;	Number of objects

Return value:

true	Normal end
false	Error

3.12. DelObjectData

Delete object data.

Declaration:

```
bool    LayoutDesigner.LibApi.DelObjectData(string object)
```

Argument:

string	object	Name of the object
--------	--------	--------------------

Return value:

true	Normal end
false	Error

3.13. GetTextData

Get the text object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetTextData(string object, ref Rectangle rect, ref string TextData,  
                                           ref int TextSource, ref string FontName, ref int FontPoint, ref int FontBold,  
                                           ref int FontItalic, ref int FontUnderline, ref int FontStrikeout, ref int FontCharset,  
                                           ref int TextAlign, ref bool AutoSize, ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
string	TextData	Get text data
int	TextSource	Get data source
string	FontName	Get a font typeface
int	FontPoint	(Font) Get point size
int	FontBold	(Font) Get enhanced modifications
int	FontItalic	(Font) Get italic modifications
int	FontUnderline	(Font) Get underline qualification
int	FontStrikeout	(Font) Get undo line qualification
int	FontCharset	(Font) Get character set
int	TextAlign	Get text alignment
bool	AutoSize	Get text auto-sizing
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.14. SetTextData

Sets the text object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetTextData(ref string object, Rectangle rect, string TextData,
                                           int TextSource, string FontName, int FontPoint, int FontBold, int FontItalic,
                                           int FontUnderline, int FontStrikeout, int FontCharset, int TextAlign, bool AutoSize,
                                           int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
string	TextData	Specification of text data
int	TextSource	Specifying a data source 0: Text data 1: Date/Time 2: Serial number
string	FontName	Specify font typeface
int	FontPoint	(Font) Specify point size
int	FontBold	(Font) Specifying enhancement modifications 0: Disable 1: Enable
int	FontItalic	(Font) Specifying italic modifications 0: Disable 1: Enable
int	FontUnderline	(Font) Specifying underline modifications 0: Disable 1: Enable
int	FontStrikeout	(Font) Designation of undo line modification 0: Disable 1: Enable
int	FontCharset	(Font) Specifying a character set 0: European sentence 128: Japanese
int	TextAlign	Specifying text alignment 0: Left justified 1: Centering 2: Right justified
bool	AutoSize	Specifying automatic text sizing true: Enable false: Disable
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

Note:

When the TextSource is date/time, set the format in advance with SetDateTime.

If the TextSource is a serial number, use SetSerial to determine the format in advance.

3.15. GetPTextData

Get the printer text object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetPTextData(string object, ref Rectangle rect, ref string TextData,  
                                             ref int TextSource, ref int PrinterFont, ref int FontBold, ref int FontItalic,  
                                             ref int FontUnderline, ref int WidthSize, ref int HeightSize, ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
string	TextData	Get text data
int	TextSource	Get data source
int	PrinterFont	Get Printer text font type
int	FontBold	(Printer text) Get enhanced modifications
int	FontItalic	(Printer text) Get italic modifications
int	FontUnderline	(Printer text) Get underline qualification
int	WidthSize	(Printer text) Get horizontal double-angle size
int	HeightSize	(Printer text) Get vertical double-angle size
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.16. SetPTextData

Sets the printer text object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetPTextData(ref string object, Rectangle rect, string TextData,
                                             int TextSource, int PrinterFont, int FontBold, int FontItalic, int FontUnderline,
                                             int WidthSize, int HeightSize, int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
string	TextData	Specification of text data
int	TextSource	Specifying a data source 0: Text data 1: Date/Time 2: Serial number
int	PrinterFont	Specifying the printer text font type 0: Font A (24 dots) 1: Font B (16 dots)
int	FontBold	(Printer text) Specifying enhancement modifications 0: Disable 1: Enable
int	FontItalic	(Printer text) Specifying italic modification 0: Disable 1: Enable
int	FontUnderline	(Printer text) Specifying underline qualification 0: Disable 1: Enable
int	WidthSize	(Printer text) Specifying the horizontal double-angle size
int	HeightSize	(Printer text) Designation of vertical double-angle size 1 to 8: 1 to 8 times
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

Note::

When the TextSource is date/time, set the format in advance with SetDateTime.

If the TextSource is a serial number, use SetSerial to determine the format in advance.

3.17. GetImageData

Get the image object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetImageData(string object, ref Rectangle rect, ref Bitmap bmp,
                                           ref bool aspect, ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
Bitmap	bmp	Acquisition of bitmap data
bool	aspect	Get aspect ratio settings (true or false)
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.18. SetImageData

Sets the image object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetImageData(ref string object, Rectangle rect, string name, int type,
                                           int threshold, bool aspect, int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
string	name	Name of the image file
int	type	Specification of conversion type of binarization 0: Simplified binarization 1: Dither transformation 2: Error diffusion
int	threshold	Designation of binarization threshold (0 to 256)
bool	aspect	Specifying the aspect ratio setting true: Maintain the aspect ratio false: Do not maintain aspect ratio
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

3.19. GetLineData

Get the straight line object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetLineData(string object, ref Point st, ref Point ed, ref int width,
                                           ref int type, ref int angle)
```

Argument:

string	object	Name of the object
Point	st	Get start coordinate data
Point	ed	Acquisition of end coordinate data
int	width	Obtaining the line width
int	type	Obtaining the line type
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.20. SetLineData

Sets the straight line object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetLineData(ref string object, Point st, Point ed, int width, int type, int angle)
```

Argument:

string	object	Name of the object
Point	st	Designation of start coordinate data
Point	ed	Designation of end coordinate data
int	width	Designation of line width (1 to 10 dots)
int	type	Designation of line type 0: Solid line 1: Dashed line 2: Dotted line 3: One-dot chain line 4: Two-dot chain line
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

3.21. GetRectData

Get the rectangle object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetRectData(string object, ref Rectangle rect, ref int width,
                                           ref int type, ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
int	width	Obtaining the line width
int	type	Obtaining the line type
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.22. SetRectData

Sets the rectangle object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetRectData(ref string object, Rectangle rect, int width, int type, int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
int	width	Designation of line width (1 to 10 dots)
int	type	Designation of line type 0: Solid line 1: Dashed line 2: Dotted line 3: One-dot chain line 4: Two-dot chain line
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

3.23. GetFillData

Get the fill object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetFillData(string object, ref Rectangle rect, ref int color, ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
int	color	Get fill color
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.24. SetFillData

Sets the fill object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetFillData(ref string object, Rectangle rect, int color, int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
int	color	Specify fill color 0: Fill color black 1: Fill color white 2: Fill color invert
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

3.25. GetBarcode1dData

Get the object data of the 1D barcode.

Declaration:

```
bool    LayoutDesigner.LibApi.GetBarcode1dData(string object, ref Rectangle rect, ref string TextData,
                                             ref int BarcodeSource, ref int BarcodeType, ref int HriPos, ref int BarcodeWidth,
                                             ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
string	TextData	Get text data
int	BarcodeSource	Obtaining a data source for a 1D barcode
int	BarcodeType	Obtaining 1D barcode type
int	HriPos	Obtaining the printing position of HRI characters
int	BarcodeWidth	Get bar width
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.26. SetBarcode1dData

Sets the object data of the 1D barcode.

Declaration:

```
bool    LayoutDesigner.LibApi.SetBarcode1dData(ref string object, Rectangle rect, string TextData,
                                                int BarcodeSource, int BarcodeType, int HriPos, int BarcodeWidth, int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
string	TextData	Specification of text data
int	BarcodeSource	Specifying the 1D barcode data source 0: Text data 1: Serial number
int	BarcodeType	To specify the 1D barcode type. 1: UPC-A 2: UPC-E 3: JAN13 4: JAN8 5: CODE39 6: ITF 7: CODABAR 8: CODE128 9: CODE93
int	HriPos	Designation of print position of HRI character 0: Do not use HRI characters 1: Align HRI characters on the upper side of the barcode 2: Align HRI characters to the bottom of the barcode 3: Align HRI characters above and below the barcode
int	BarcodeWidth	Specifying the bar width 0 to 3: 1 to 4 dots
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

Note:

If the TextSource is a serial number, use SetSerial to determine the format in advance.

3.27. GetBarcode2dData

Get the object data of the 2D barcode.

Declaration:

```
bool    LayoutDesigner.LibApi.GetBarcode2dData(string object, ref Rectangle rect, ref string TextData,  
        ref int BarcodeSource, ref int BarcodeType, ref int CellSize, ref int EccLevel, ref int SymbolSize,  
        ref int Colum, ref int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Obtaining object coordinates
string	TextData	Get text data
int	BarcodeSource	Get 2D barcode data sources
int	BarcodeType	Acquisition of 2D barcode type
int	CellSize	Get cell size for QR code
int	EccLevel	Obtaining ECC levels
int	SymbolSize	Get symbol size
int	Colum	Get column count (discard except Micro PDF417 selection)
int	angle	Get rotation angle

Return value:

true	Normal end
false	Error

3.28. SetBarcode2dData

Sets the object data of the 2D barcode.

Declaration:

```
bool LayoutDesigner.LibApi.SetBarcode2dData(ref string object, Rectangle rect, string TextData,
                                             int BarcodeSource, int BarcodeType, int CellSize, int EccLevel, int SymbolSize,
                                             int Colum, int angle)
```

Argument:

string	object	Name of the object
Rectangle	rect	Specifying object coordinates
string	TextData	Specification of text data
int	BarcodeSource	Specifying the 2D barcode data source 0: Text data 1: Serial number
int	BarcodeType	2D barcode type specification 1: QR CODE 2: Micro QR CODE 3: PDF417 4: Micro PDF417 5: DATA MATRIX 6: MAXI CODE
int	CellSize	To specify the cell size of the QR code. 1 to 8: Number of dots per 1 cell
int	EccLevel	Specifying the ECC Level 1: ECC level 7% (QR CODE /Micro QR CODE) 2: ECC level 15% (QR CODE /Micro QR CODE) 3: ECC level 25% (QR CODE /Micro QR CODE) 4: ECC level 30% (QR CODE)
int	SymbolSize	Designation of symbol size When QR CODE is selected 0: Auto size 1 to 40: Specified size When Micro QR CODE is selected 0: Auto size 1 to 4: Specified size When DATA MATRIX is selected Specified size of 10, 18, 22, 26, 32, 40, 48
int	Colum	Number of columns specified (Value enabled only when Micro PDF417 is selected) 0 to 3: Number of columns 1 to 4 is specified.
int	angle	Specifying the rotation angle 0: Rotation angle 0 degrees 1: Rotation angle 90 degrees 2: Rotation angle 180 degrees 3: Rotation angle 270 degrees

Return value:

true	Normal end
false	Error

Note:

If the TextSource is a serial number, use SetSerial to determine the format in advance.

3.29. GetMacroData

Get the document macro object data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetMacroData(string object, ref string[] start_cmd, ref string[] end_cmd)
```

Argument:

string	object	Name of the object
string[]	start_cmd	Get page start command
string[]	end_cmd	Get page end command

Return value:

true	Normal end
false	Error

3.30. SetMacroData

Sets the document macro object data.

Declaration:

```
bool    LayoutDesigner.LibApi.SetMacroData(ref string object, string[] start_cmd, string[] end_cmd)
```

Argument:

string	object	Name of the object
string[]	start_cmd	Page start command specification
string[]	end_cmd	Designation of page end command

Return value:

true	Normal end
false	Error

3.31. AlternativeFontNameSet

If the font name specified in the Text object does not exist, the font name specified in this API will be used instead.

Declaration:

```
bool    LayoutDesigner.LibApi.AlternativeFontNameSet(string FontName)
```

Argument:

string	FontName	Alternate font name
--------	----------	---------------------

Return value:

true	Normal end
false	Error

3.32. InitStandaloneMacro

Initialize a standalone macro.

Declaration:

```
void InitStandaloneMacro ()
```

Argument: None

Return value: None

3.33. GetStandaloneMacroData

Get the command data of a standalone macro.

Declaration:

```
bool LayoutDesigner.LibApi.GetStandaloneMacroData(int macro_select, ref string[] stand_cmd)
```

Argument:

int	macro_select	Macro selection number
string[]	stand_cmd	Standalone macro command

Return value:

true	Normal end
false	Error

3.34. SetStandaloneMacroData

Sets the command data for a standalone macro.

Declaration:

```
bool LayoutDesigner.LibApi.SetStandaloneMacroData(int macro_select, string[] stand_cmd)
```

Argument:

int	macro_select	Macro selection number
string[]	stand_cmd	Standalone macro command

Return value:

true	Normal end
false	Error

3.35. GetStandaloneMacroTitle

Get the display name of a standalone macro.

Declaration:

```
bool    LayoutDesigner.LibApi.GetStandaloneMacroTitle(int macro_select, ref string title)
```

Argument:

int	macro_select	Macro selection number
string	title	Macro display name

Return value:

true	Normal end
false	Error

3.36. SetStandaloneMacroTitle

Sets the display name of the standalone macro.

Declaration:

```
bool    LayoutDesigner.LibApi.SetStandaloneMacroTitle(int macro_select, ref string title)
```

Argument:

int	macro_select	Macro selection number
string	title	Macro display name

Return value:

true	Normal end
false	Error

3.37. GetStandaloneMacroCount

Get the number of standalone macro data.

Declaration:

```
bool    LayoutDesigner.LibApi.GetStandaloneMacroCount (int macro_select, ref int count)
```

Argument:

int	macro_select	Macro selection number
string	count	Number of macro data

Return value:

true	Normal end
false	Error

3.38. LoadStandaloneMacro

Load a standalone macro file.

Declaration:

bool LayoutDesigner.LibApi.LoadStandaloneMacro (string file, ref int size)

Argument:

string	file	Standalone macro file name
int	size	File size

Return value:

true	Normal end
false	Error

3.39. SaveStandaloneMacro

Save a standalone macro file.

Declaration:

bool LayoutDesigner.LibApi.SaveStandaloneMacro (string file)

Argument:

string	file	Standalone macro file name
--------	------	----------------------------

Return value:

true	Normal end
false	Error

3.40. RegistStandaloneMacro

Register the standalone macro with the printer.

Declaration:

bool LayoutDesigner.LibApi.RegistStandaloneMacro (int macro_select, ref string title)

Argument:

int	macro_select	Macro selection number
string	title	Macro display name

Return value:

true	Normal end
false	Error

4. Constant definition

The list of error constant definitions for the LibApi class is as follows:

Name	Value	Item
Error type		
ERROR_CODE_OK	0	Normal end
ERROR_CODE_VER_GET	1	Failed to acquire version
ERROR_CODE_SDK_INIT	2	SDK initialization failed
ERROR_CODE_FILE_NEW	3	Failed to create new SOP file
ERROR_CODE_PAPER_SETTING	4	Paper setting failed
ERROR_CODE_FILE_READ	5	Failed to load SOP file
ERROR_CODE_FILE_WRITE	6	Failed to write SOP file
ERROR_CODE_FILE_PRINT	7	Failed to print SOP file
ERROR_CODE_FILE_PREVIEW	8	SOP file preview creation failed
ERROR_CODE_OBJECT_LIST_GET	9	Failed to acquire object list
ERROR_CODE_OBJECT_DATA_GET	10	Failed to acquire object data
ERROR_CODE_OBJECT_DATA_SET	11	Setting of object data failed
ERROR_CODE_OBJECT_DATA_DEL	12	Object data deletion failed
ERROR_CODE_SERIAL_INIT	13	Serial number initialization failed
ERROR_CODE_SERIAL_SET	14	Serial number setting failed
ERROR_CODE_DATE_TIME_SET	15	Failed to set date/time data
ERROR_CODE_TEXT_OBJECT_GET	16	Text data acquisition failed
ERROR_CODE_TEXT_OBJECT_SET	17	Setting of text data failed
ERROR_CODE_PTEXT_OBJECT_GET	18	Failed to acquire printer text data
ERROR_CODE_PTEXT_OBJECT_SET	19	Setting of printer text data failed
ERROR_CODE_IMAGE_OBJECT_GET	20	Failed to acquire image data
ERROR_CODE_IMAGE_OBJECT_SET	21	Failed to set image data
ERROR_CODE_LINE_OBJECT_GET	22	Failed to acquire linear data
ERROR_CODE_LINE_OBJECT_SET	23	Setting of linear data failed
ERROR_CODE_RECT_OBJECT_GET	24	Failed to acquire rectangular data
ERROR_CODE_RECT_OBJECT_SET	25	Setting of rectangular data failed
ERROR_CODE_FILL_OBJECT_GET	26	Failed to acquire fill data
ERROR_CODE_FILL_OBJECT_SET	27	Failed to set fill data
ERROR_CODE_BARCODE_1D_OBJECT_GET	28	Failed to acquire 1D barcode data
ERROR_CODE_BARCODE_1D_OBJECT_SET	29	Setting of 1D barcode data failed
ERROR_CODE_BARCODE_2D_OBJECT_GET	30	Failed to acquire 2D barcode data
ERROR_CODE_BARCODE_2D_OBJECT_SET	31	Setting of 2D barcode data failed
ERROR_CODE_MACRO_OBJECT_GET	32	Failed to acquire document macro data
ERROR_CODE_MACRO_OBJECT_SET	33	Document macro data setting failed
ERROR_CODE_STANDALONE_MACRO_GET	34	Failed to get standalone macro data
ERROR_CODE_STANDALONE_MACRO_SET	35	Failed to set standalone macro data
ERROR_CODE_STANDALONE_MACRO_LOAD	36	Failed to load standalone macro data
ERROR_CODE_STANDALONE_MACRO_SAVE	37	Failed to save standalone macro data
ERROR_CODE_STANDALONE_MACRO_REGIST	38	Failed to register standalone macro data

The list of constant definitions that can be specified in the API of the LibApi class is as follows.

Name	Value	Item
Printer model number		
PRINTER_SK1_21	1	SK1-21、SK1-211
PRINTER_SK1_31	2	SK1-31、SK1-311
PRINTER_SK1_41	3	SK1-41
PRINTER_SK4_21	4	SK4-21
PRINTER_SK4_31	5	SK4-31
PRINTER_BL2_58	6	BL2-58
PRINTER_SD3_21	7	SD3-21
PRINTER_SD3_22	8	SD3-22
PRINTER_SM1_21	9	SM1-21
PRINTER_BLM_80	10	BLM-80
PRINTER_SM2_41	11	SM2-41
PRINTER_SM3_21	12	SM3-21
PRINTER_SM4_21	13	SM4-21
PRINTER_SM4_31	14	SM4-31
Object type		
OBJECT_TYPE_NONE	0	Invalid object
OBJECT_TYPE_TEXT	1	Text object
OBJECT_TYPE_PTEXT	2	Printer text object
OBJECT_TYPE_RECT	3	Rectangular object
OBJECT_TYPE_FILL	4	Fill object
OBJECT_TYPE_LINE	5	Linear object
OBJECT_TYPE_IMAGE	6	Image object
OBJECT_TYPE_BAR1	7	1D barcode object
OBJECT_TYPE_BAR2	8	2D barcode object
OBJECT_TYPE_MACRO	9	Document macro object
Rotation angle		
OBJECT_ANGLE0	0	Rotation angle 0 degrees
OBJECT_ANGLE90	1	Rotation angle 90 degrees
OBJECT_ANGLE180	2	Rotation angle 180 degrees
OBJECT_ANGLE270	3	Rotation angle 270 degrees
Data source for text or printer text		
TEXT_SOURCE_TEXT	0	Text data
TEXT_SOURCE_DATE	1	Date/Time
TEXT_SOURCE_SERIAL	2	Serial number
Enhancement modification of text or printer text		
TEXT_BOLD_ENABLE	0	Enable the enhancement modification
TEXT_BOLD_DISABLE	1	Disable the enhancement modification
Italic modification of text or printer text		
TEXT_ITALIC_ENABLE	0	Enable italic modification
TEXT_ITALIC_DISABLE	1	Disable italic modification
Underline modification of text or printer text		
TEXT_UNDERLINE_ENABLE	0	Enable underline modification
TEXT_UNDERLINE_DISABLE	1	Disable the underline modification
Text Font Undo Line Qualification		
TEXT_STRIKE_ENABLE	0	Enable undo line modification
TEXT_STRIKE_DISABLE	1	Disable undo line modification
Character set of a Text character		
CHARACTER_SET_JAPAN	128	Character set Japanese
CHARACTER_SET_EURO	0	Character set European sentence

Text Font Alignment		
TEXT_ALIGN_LEFT	0	Left justified
TEXT_ALIGN_CENTER	1	Centering
TEXT_ALIGN_RIGHT	2	Right justified
Auto-Size Text Fonts		
TEXT_AUTO_SIZE_DISABLE	false	Disable automatic sizing
TEXT_AUTO_SIZE_ENABLE	true	Enable auto size adjustment
Printer Text Font Type		
PTEXT_FONTA	0	Font A (24 dots)
PTEXT_FONTB	1	Font B (16 dots)
Double-angle size of the printer text		
PTEXT_ZOOM_SIZE1	1	1x
PTEXT_ZOOM_SIZE2	2	2x
PTEXT_ZOOM_SIZE3	3	3x
PTEXT_ZOOM_SIZE4	4	4x
PTEXT_ZOOM_SIZE5	5	5x
PTEXT_ZOOM_SIZE6	6	6x
PTEXT_ZOOM_SIZE7	7	7x
PTEXT_ZOOM_SIZE8	8	8x
Binary conversion type		
IMAGE_CONVERT_SIMPLE	0	Simplified binarization
IMAGE_CONVERT_DITHER	1	Dither transformation
IMAGE_CONVERT_ERROR_DIFFUSION	2	Error diffusion
Binarization threshold		
IMAGE_THRESHOLD_MIN	0	Threshold minimum
IMAGE_THRESHOLD_MAX	256	Threshold maximum
Aspect Ratio		
ASPECT_RATIO_NON_KEEP	false	Do not maintain aspect ratio
ASPECT_RATIO_KEEP	true	Maintain the aspect ratio
Line type		
LINE_TYPE_SOLID	0	Line type solid line
LINE_TYPE_DASH	1	Type dashed line
LINE_TYPE_DOT	2	Line type dotted line
LINE_TYPE_1DOT_CHAIN	3	Line type 1-dot chain line
LINE_TYPE_2DOT_CHAIN	4	Line type 2-dot chain line
Line width		
LINE_WIDTH_1DOT	1	Line width 1 dot
LINE_WIDTH_2DOT	2	Line width 2 dots
LINE_WIDTH_3DOT	3	Line width 3 dots
LINE_WIDTH_4DOT	4	Line width 4 dots
LINE_WIDTH_5DOT	5	Line width 5 dots
LINE_WIDTH_6DOT	6	Line width 6 dots
LINE_WIDTH_7DOT	7	Line width 7 dots
LINE_WIDTH_8DOT	8	Line width 8 dots
LINE_WIDTH_9DOT	9	Line width 9 dots
LINE_WIDTH_10DOT	10	Line width 10 dots
Fill color		
FILL_COLOR_BLACK	0	Fill color black
FILL_COLOR_WHITE	1	Fill color white
FILL_COLOR_REVERSE	2	Fill color invert

1D barcode type		
BARCODE1D_TYPE_UPCA	1	UPC-A
BARCODE1D_TYPE_UPCE	2	UPC-E
BARCODE1D_TYPE_JAN13	3	JAN13
BARCODE1D_TYPE_JAN8	4	JAN8
BARCODE1D_TYPE_CODE39	5	CODE39
BARCODE1D_TYPE_ITF	6	ITF
BARCODE1D_TYPE_CODABAR	7	CODABAR
BARCODE1D_TYPE_CODE128	8	CODE128
BARCODE1D_TYPE_CODE93	9	CODE93
2D barcode types		
BARCODE2D_TYPE_QRCODE	1	QR CODE
BARCODE2D_TYPE_MICRO_QRCODE	2	Micro QR CODE
BARCODE2D_TYPE_PDF417	3	PDF417
BARCODE2D_TYPE_MICRO_PDF417	4	Micro PDF417
BARCODE2D_TYPE_DATA_MATRIX	5	DATA MATRIX
BARCODE2D_TYPE_MAXI_CODE	6	MAXI CODE
1D barcode and 2D barcode data sources		
BARCODE_SOURCE_TEXT	0	Text data
BARCODE_SOURCE_SERIAL	1	Serial number
Printing position of HRI letters		
HRI_LOCATE_NONE	0	Do not use HRI characters
HRI_LOCATE_UPPER	1	Align HRI characters on the upper side of the barcode
HRI_LOCATE_BOTTOM	2	Align HRI characters to the bottom of the barcode
HRI_LOCATE_BOTH	3	Align HRI characters above and below the barcode
Bar width		
BAR_WIDTH1	0	Bar width 1 dot
BAR_WIDTH2	1	Bar width 2 dots
BAR_WIDTH3	2	Bar width 3 dots
BAR_WIDTH4	3	Bar width 4 dots
Cell size of the QR code		
QRCODE_CELL_SIZE1	1	Cell size 1 of QR CODE
QRCODE_CELL_SIZE2	2	Cell size 2 of QR CODE
QRCODE_CELL_SIZE3	3	Cell size 3 of QR CODE
QRCODE_CELL_SIZE4	4	Cell size 4 of QR CODE
QRCODE_CELL_SIZE5	5	Cell size 5 of QR CODE
QRCODE_CELL_SIZE6	6	Cell size 6 of QR CODE
QRCODE_CELL_SIZE7	7	Cell size 7 of QR CODE
QRCODE_CELL_SIZE8	8	Cell size 8 of QR CODE

Symbol size		
QRCODE_SYMBOL_MIN_SIZE	0	QR CODE auto size
QRCODE_SYMBOL_MAX_SIZE	40	QR CODE maximum size
MICRO_QRCODE_SYMBOL_MIN_SIZE	0	MICRO QR CODE auto size
MICRO_QRCODE_SYMBOL_MAX_SIZE	4	MICRO QR CODE maximum size
DATAMATRIX_SYMBOL_SIZE10	10	DATA MATRIX size 10
DATAMATRIX_SYMBOL_SIZE18	18	DATA MATRIX size 18
DATAMATRIX_SYMBOL_SIZE22	22	DATA MATRIX size 22
DATAMATRIX_SYMBOL_SIZE26	26	DATA MATRIX size 26
DATAMATRIX_SYMBOL_SIZE32	32	DATA MATRIX size 32
DATAMATRIX_SYMBOL_SIZE40	40	DATA MATRIX size 40
DATAMATRIX_SYMBOL_SIZE48	48	DATA MATRIX size 48
ECC level		
QRCODE_ECC_L	1	QR CODE ECC Level 7%
QRCODE_ECC_M	2	QR CODE ECC level 15%
QRCODE_ECC_Q	3	QR CODE ECC level 25%
QRCODE_ECC_H	4	QR CODE ECC level 30%
Number of rows		
MICRO_PDF417_COLUMNS_SIZE1	0	Column number 1 of Micro PDF417
MICRO_PDF417_COLUMNS_SIZE2	1	Column number 2 of Micro PDF417
MICRO_PDF417_COLUMNS_SIZE3	2	Column number 3 of Micro PDF417
MICRO_PDF417_COLUMNS_SIZE4	3	Column number 4 of Micro PDF417
Object type with serial number		
OBJECT_TYPE_TEX	0	Text
OBJECT_TYPE_BARCODE	1	Barcode
Serial number type		
SERIAL_TYPE_DECIMAL	0	Decimal number
Increasing/decreasing serial number		
SERIAL_NUMBER_INC	0	Sequential number increase
SERIAL_NUMBER_DEC	1	Sequential decrease
Selection of date or time		
SELECT_TYPE_DATE	0	Date selection
SELECT_TYPE_TIME	1	Time selection
Date offset type		
OFFSET_TYPE_DAY	0	Day
OFFSET_TYPE_MONTH	1	Month
OFFSET_TYPE_YEAR	2	Year
Date offset range		
OFFSET_DAY_MIN	0	Daily offset minimum
OFFSET_DAY_MAX	30	Maximum daily offset
OFFSET_MONTH_MIN	0	Minimum offset value for the month
OFFSET_MONTH_MAX	11	Maximum offset value for the month
OFFSET_YEAR_MIN	0	Year's minimum offset
OFFSET_YEAR_MAX	98	Year's maximum offset

Date format		
DATE_FORMAT_TYPE01	0	2021/04/13(Year/Month/Day)
DATE_FORMAT_TYPE02	1	21/04/13(Year/Month/Day)
DATE_FORMAT_TYPE03	2	2021-04-13(Year-Month-Day)
DATE_FORMAT_TYPE04	3	21-04-13(Year-Month-Day)
DATE_FORMAT_TYPE05	4	2021,04,13(Year,Month,Day)
DATE_FORMAT_TYPE06	5	21,04,13(Year,Month,Day)
DATE_FORMAT_TYPE07	6	2021.04.13(Year.Month.Day)
DATE_FORMAT_TYPE08	7	21.04.13(Year,Month.Day)
DATE_FORMAT_TYPE09	8	April 13, 2021
DATE_FORMAT_TYPE10	9	April 13, 210
DATE_FORMAT_TYPE11	10	April 13, 2021 Tuesday
DATE_FORMAT_TYPE12	11	April 13, 210 Tuesday
DATE_FORMAT_TYPE13	12	April 13, 2003
DATE_FORMAT_TYPE14	13	April 13, R03
DATE_FORMAT_TYPE15	14	R03/04/13(Year/Month/Day)
DATE_FORMAT_TYPE16	15	R03-04-13(Year-Month-Day)
DATE_FORMAT_TYPE17	16	R03,04,13(Year,Month,Day)
DATE_FORMAT_TYPE18	17	R03.04.13(Year.Month.Day)
DATE_FORMAT_TYPE19	18	210413(YearMonthDay)
DATE_FORMAT_TYPE20	19	20210413(YearMonthDay)
DATE_FORMAT_TYPE21	20	Fire (DayOfWeek)
DATE_FORMAT_TYPE22	21	21(Year)
DATE_FORMAT_TYPE23	22	2021(Year)
DATE_FORMAT_TYPE24	23	04(Month)
DATE_FORMAT_TYPE25	24	13(Day)
Time format		
TIME_FORMAT_TYPE01	0	02:22 PM(12 Hour Format)
TIME_FORMAT_TYPE02	1	02:22(12 Hour Format)
TIME_FORMAT_TYPE03	2	02:22:19(12 Hour Format)
TIME_FORMAT_TYPE04	3	02:22:19(12 Hour Format:2Digits)
TIME_FORMAT_TYPE05	4	02:22:19 PM(12 Hour Format)
TIME_FORMAT_TYPE06	5	14:22 PM(24 Hour Format)
TIME_FORMAT_TYPE07	6	14:22(24 Hour Format)
TIME_FORMAT_TYPE08	7	14:22:19(24 Hour Format)
TIME_FORMAT_TYPE09	8	14:22:19(24 Hour Format:2Digits)
TIME_FORMAT_TYPE10	9	02:22 (12 Hour Format)
TIME_FORMAT_TYPE11	10	14:22 (24 Hour Format)
TIME_FORMAT_TYPE12	11	02:22:19 (12 Hour Format)
TIME_FORMAT_TYPE13	12	14:22:19 (24 Hour Format)
TIME_FORMAT_TYPE14	13	02:22 p.m. (12 Hour Format)
TIME_FORMAT_TYPE15	14	14:22 p.m. (24 Hour Format)
TIME_FORMAT_TYPE16	15	PM
TIME_FORMAT_TYPE17	16	02:00 (12 Hour Format)
TIME_FORMAT_TYPE18	17	14:00 (24 Hour Format)
TIME_FORMAT_TYPE19	18	02:00(12 Hour Format)
TIME_FORMAT_TYPE20	19	14:00(24 Hour Format)

5. Commands that can be used in document macros

The commands that can be used in the document macro are as follows.

Name	Command string	Argument
Feed	FEED	Paper feed amount
Back feed	BACK_FEED	Paper feed amount
Full cut	FULL_CUT	None
Partial cut	PARTIAL_CUT	None
Printer initialization	PRT_INIT	None
Print density setting	PRT_DENSITY	Print density
Marking position detection	MARK_DETECTION	None
Marking position re-detection	RE-MARK_DETECTION	None
Registration to non-volatile memory begins	REG_BEGINNING	Memory number
Registration to non-volatile memory is completed	REG_END	Memory number

To actually set with the document macro, describe as follows.

```
string[] start = { @"PRT_INIT", @" PRT_DENSITY,120"}; //Page start macro.
string[] end = { @"FEED,120"}; //End of page macro
string item = @""; //Object name
```

```
SetMacroData(ref item, start, end); //Macro settings
```

6. Commands that can be used in standalone macros

The commands that can be used in standalone macros are as follows.

Name	Command string	Argument	Others
Feed	FEED	Paper feed amount	
Back feed	BACK_FEED	Paper feed amount	
Printer initialization	PRT_INIT	None	
Print density setting	PRT_DENSITY	Print density	
New line	CR	None	
Space	SPACE	None	
Placement	ALIGNMENT	Specified position	
Printer font size	FONT_SIZE	None	
Time stamp	TIME_PRINT	Form number	
Time loading	TIME_READ	None	
Time stamp with offset	OFFSET_TIME_PRINT	Offset type, Offset value, Format	
Counter printing	COUNTER_PRINT	None	
Zero suppression setting	ZERO_SUPPRESS	Number of digits, Zero suppression setting	
Counter inc/dec setting	INCDEC_SETTING	Counter inc/dec timing, Number of inc/dec, Number of times until inc/dec execution, Inc or dec	
Kanji code selection	KANJI_SELECT	Kanji code system	
QR code printing	QR_CODE	Size, ECC level	Use QR string
Full cut	FULL_CUT	None	
Partial cut	PARTIAL_CUT	None	
Marking position detection	MARK_DETECTION	None	

Registration image print	BATCH_PRINT	None	
String variable setting	STRVAL_SET	Variable number	Use variable string
Cell size setting	CELL_SET	Cell size	

The format of the command string specified in the standalone macro is as follows.

“Command, Arg1, Arg2, Arg3, Arg4, Arg5, Arg6, QR string, Variable string, Number of commands”

Command : A command string such as FEED or CR.

Arg : Specify a number from 0 to 255 in the command argument.
Specify 0 even if no argument is specified.

QR string : Used to set QR code data.
You don't specify anything other than setting the QR string.

Variable string : Used to set string variables.
You don't specify anything other than setting the string variable.

Number of commands : Specifies the number of commands.
The range is from 1 to 30.

The commands that can be used in the QR string are as follows.

Name	Command string	Argument	Function
Time set 0	\$TIM0	None	Set the time in the QR data in the format of “YYYYMMDDhhmm”
Time set 1	\$TIM1	None	Set the time in the QR data in the format of “YYMMDDhhmm”
Time set 2	\$TIM2	None	Set the time in the QR data in the format of “YYYYMMDDhhmm”
Time set 3	\$TIM3	None	Set the time in the QR data in the format of “YYMMDDhhmm”
Time set 4	\$TIM4	None	Set the time in the QR data in the format of “YYYYMMDD”
Time set 5	\$TIM5	None	Set time in QR data in EPOCH format
Set 0 for time with offset	\$TIMd	Month offset, Form number	Format numbers are 0–4, corresponding to time sets 0–4
Set 1 for time with offset	\$TIME	Day offset, Form number	Format numbers are 0–4, corresponding to time sets 0–4
Set 2 for time with offset	\$TIMf	Hour offset, Form number	Format numbers are 0–4, corresponding to time sets 0–4
Counter increment	\$INC	None	Update the counter
Set 0 of string variables	\$STR0	None	Set the contents of string variable 0
Set 1 of string variables	\$STR1	None	Set the contents of string variable 1
Set 2 of string variables	\$STR2	None	Set the contents of string variable 2
Set 3 of string variables	\$STR3	None	Set the contents of string variable 3
Set 4 of string variables	\$STR4	None	Set the contents of string variable 4
Set 5 of string variables	\$STR5	None	Set the contents of string variable 5
Set 6 of string variables	\$STR6	None	Set the contents of string variable 6
Set 7 of string variables	\$STR7	None	Set the contents of string variable 7
Checksum	\$CHK	None	Add checksum to end of data

To actually register the standalone macro, describe as follows.

```
string stand_cmd1 = { @" QR_CODE,8,2,0,0,0,0,SANEI ELEC,,1"}; //Print QR code
string stand_cmd1 = { @" CR,0,0,0,0,0,0,,10"}; //10 line breaks

InitStandaloneMacro(); //Initializing a standalone macro
SetStandaloneMacroTitle(0, @"1 QR TEST"); //Set display name
RegistStandaloneMacro(@"SANEI SM4-31-UNI-JP", 0); //Register standalone macros on the printer
```

The SDK comes with a sample program that creates the label image shown in the figure below. Includes how to use each API.

See Chapter 2 for sample configuration and processes.

