

Multi LABELIST Component Technical Manual

SATO Corporation

Introduction

Thank you for using the “Multi LABELIST Component” (hereinafter referred to as ML Component). ML Component is a .NET component developed to add label/tag issuance functionality to your applications by utilizing the assets of our general-purpose label/tag issuance software “Multi LABELIST V6” (hereinafter referred to as MLV6).

To enable you to build a highly flexible label/tag issuing system based on layout files created in MLV6, we have omitted some of MLV6’s functions. However, the design supports output devices such as USB, LAN, COM (serial port), and our printer drivers. It also supports status monitoring functionality, allowing you to retrieve the current status of the printer.

This manual explains various usage methods to help you understand ML Component. For detailed explanations of each property and method, please refer to [the “ML Component Reference Manual.”](#)

Important Notice

- Reproduction or duplication of any part or all of this manual without our permission is strictly prohibited, in any form.
- The contents of this manual may be revised or improved without prior notice.
- We shall not be liable for any consequences arising from the use of this manual.
- We have made every effort to ensure the accuracy of this manual. However, if you have any questions or concerns, please contact us.
- SATO and Multi LABELIST are registered trademarks or trademarks of SATO Corporation and its subsidiaries in Japan.
- Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States.
- Other company names and product names mentioned herein are registered trademarks or trademarks of their respective companies.

Table of Contents

Introduction.....	3
Important.....	3
1 : Basic Edition	6
1-1. Using Visual Studio	7
■ Procedure for Using Visual Studio 2022	7
■ Instance creation.....	9
1-2.Distributing the application.....	11
■ Creating an installer in Visual Studio 2022.....	11
1-3.Decide on the distribution method.....	14
■How to print.....	14
1-4.Connect to a printer	15
■Connection/Disconnection	15
■ Connect via USB.....	15
■ Connect via LAN.....	16
■ Connect via COM (serial port)	16
■ Connect via Bluetooth.....	16
1-5.Check printer status.....	17
■Communication protocol.....	17
■Status check.....	17
1-6.Use printer driver.....	19
■Connection/Disconnection	19
1-7.Issue labels and tags.....	20
■Label issuance	20
1-8.Enter data in bulk.....	21
■Input order	21
■Data format	21
■Enter multiple data	22
1-9.Enter data by specifying variable names.....	23
■Variable name.....	23
1-10.Check the version.....	25
■ Obtain from the Version property	25
■Check in file properties.....	25
1-11.Perform version upgrade.....	26
■Update MLComponent in the development environment.....	26
■Update MLComponent in the execution environment.....	26
1-12. Using with Microsoft Excel/Access.....	27
■Installation.....	27

■ Add references.....	27
■ Generate instance.....	30
2 : Advanced.....	31
2-1.Changing printer density and speed.....	32
■ Changing the density	32
■ Changing the speed.....	32
2-2.Adjusting the print position.....	33
■ Adjusting the print position.....	33
2-3.Set consumption tax.....	34
■ Tax editing	34
■ Consumption tax.....	35
2-4.Print serial numbers.....	36
■ Print serial numbers.....	36
■ Enter the initial value for serial numbers.....	37
2-5.Issue header and footer tags.....	39
■ Issue header and tail tags.....	39
■ Issue header and tail tags according to the layout settings.....	40
2-6.Use multi-panel labels	42
■ Enter multi-panel labels for the specified number of sheets (1) and issue them.....	42
■ Issue multi-panel labels by entering multiple sheets (printer driver output only)	43
2-7.Print sorting marks	45
■ Print sorting marks	45
2-8.Cut tags and labels	46
■ Cut.....	46
2-9.Cancel label issuance	48
■ Cancel issuance.....	48
2-10. Send printer command (SBPL)	49
■ Send command.....	49
■ Receive command	49
■ Send SBPL as a string.....	49
■ Send SBPL as a byte array.....	50
2-11.Use the operation setting file.....	51
■ Output log file.....	53
2-12.Improve the initial layout loading speed	55
■ Place MLComponent.XmlSerializers.dll.....	55
2-13.Improve publishing speed.....	58
■ Pre-generate printer commands for fixed objects	58

Chapter 1

Basic

1

Using Visual Studio

■ Adding References ■ Declaration

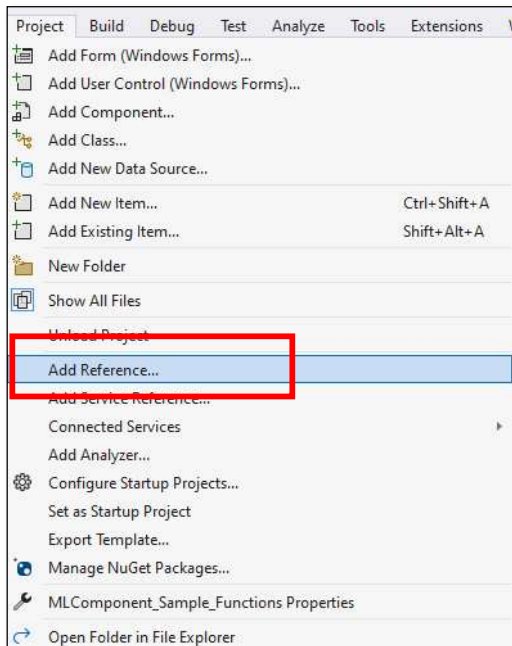
To use ML Component in Visual Studio, add a reference and create an instance in the code.

■ Usage steps in Visual Studio 2022

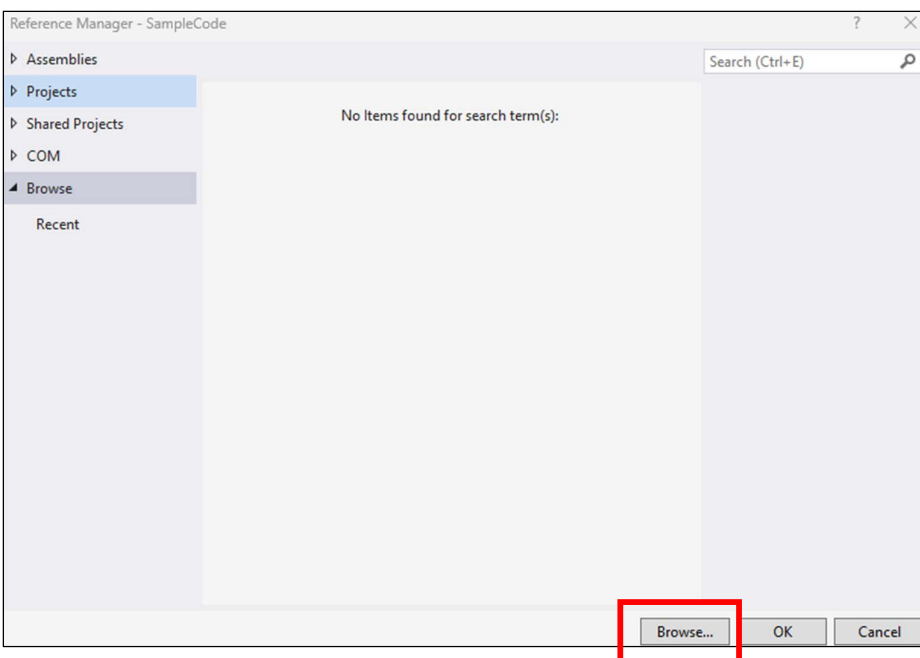
Create a project named "Windows Forms Application (.NET Framework)."

Select ".NET Framework 4.8" as the framework.

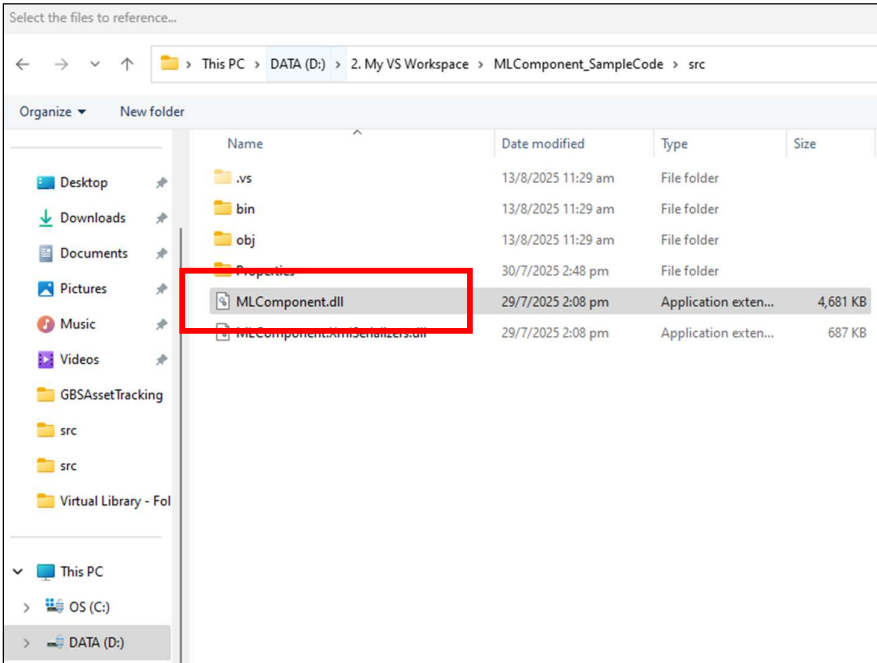
Select "Projects" > "Add Reference."



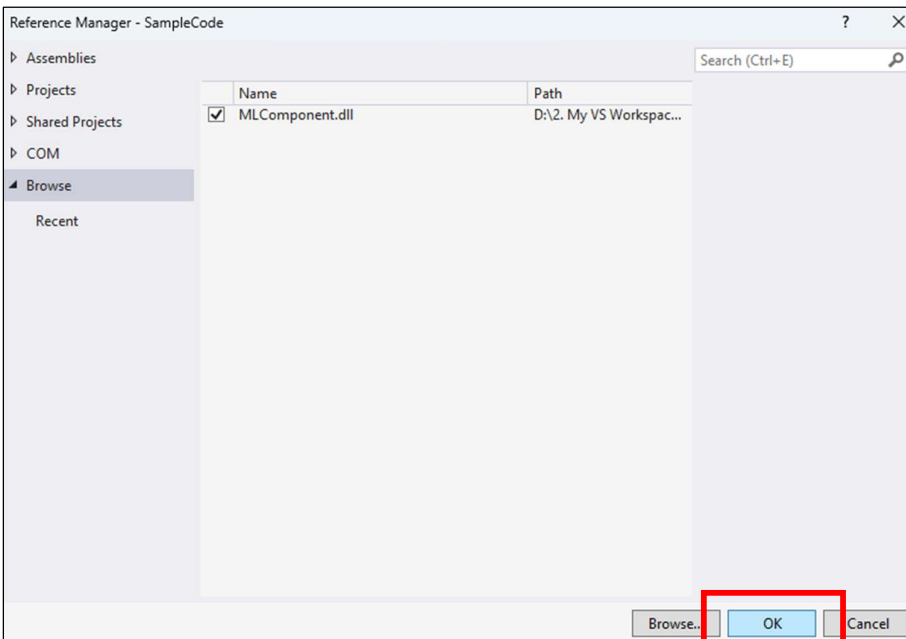
Click "References."



Select "MLComponent.dll" and click "Add."



Verify that "MLComponent.dll" has been added to the "References" list, then click "OK."

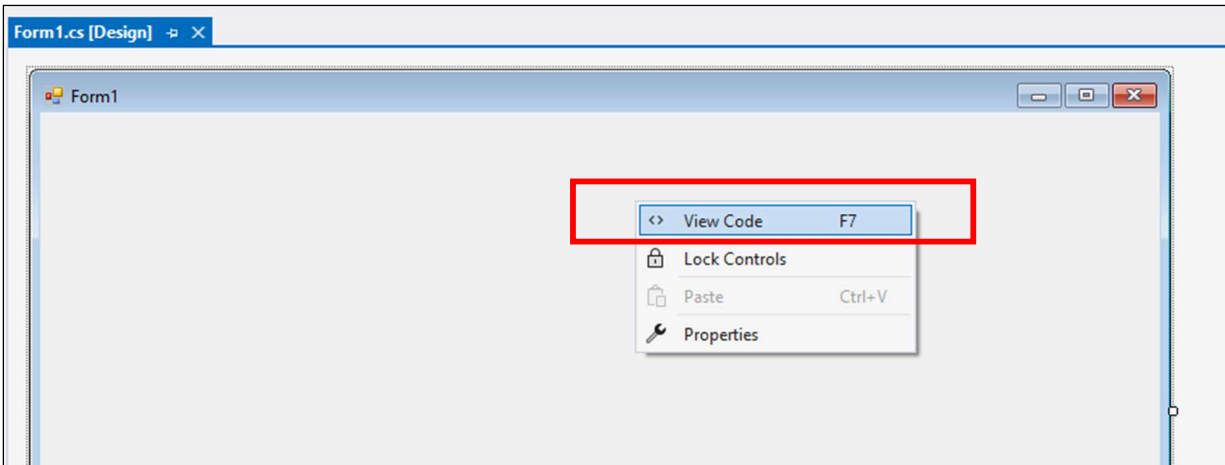


Explanation: Reference Settings

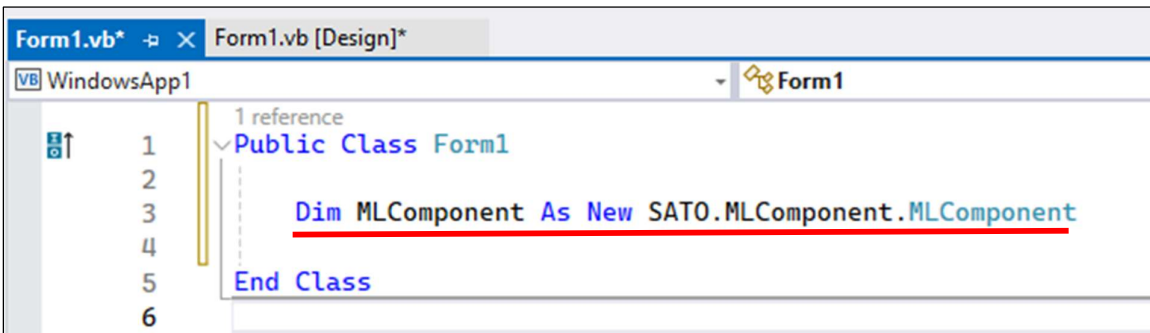
When you open the reference settings in Solution Explorer, the libraries referenced by the application are displayed. After adding the reference for ML Component, you can confirm that it has been added to the list as an assembly.

■ Instance creation

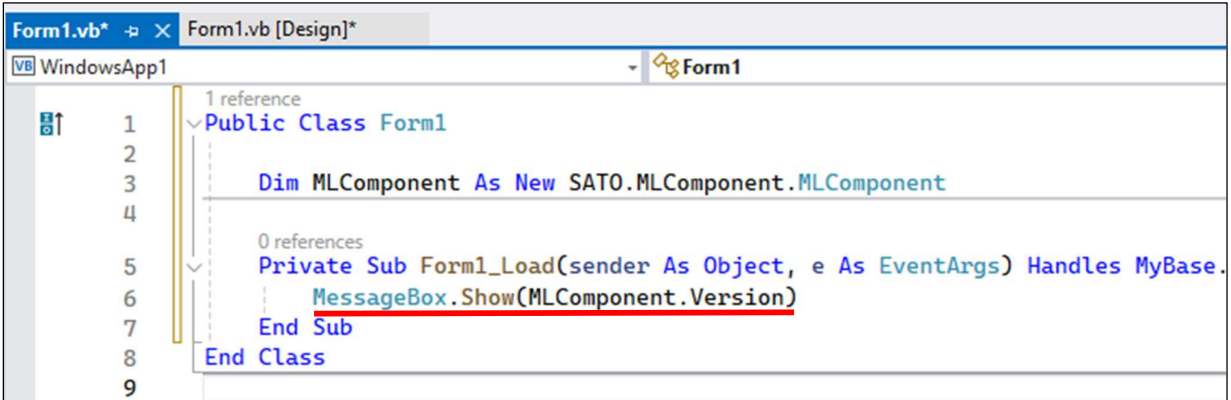
Right-click on the form and select "Show Code."



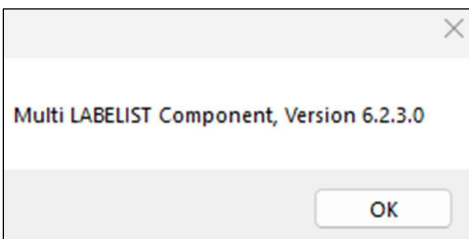
Create an instance of ML Component.



Check that the ML Component instance was created correctly. When the form loads, display the version information (the value of the Version property) in a dialog box.



Start debugging, and if the dialog appears as shown below, the process was successful.



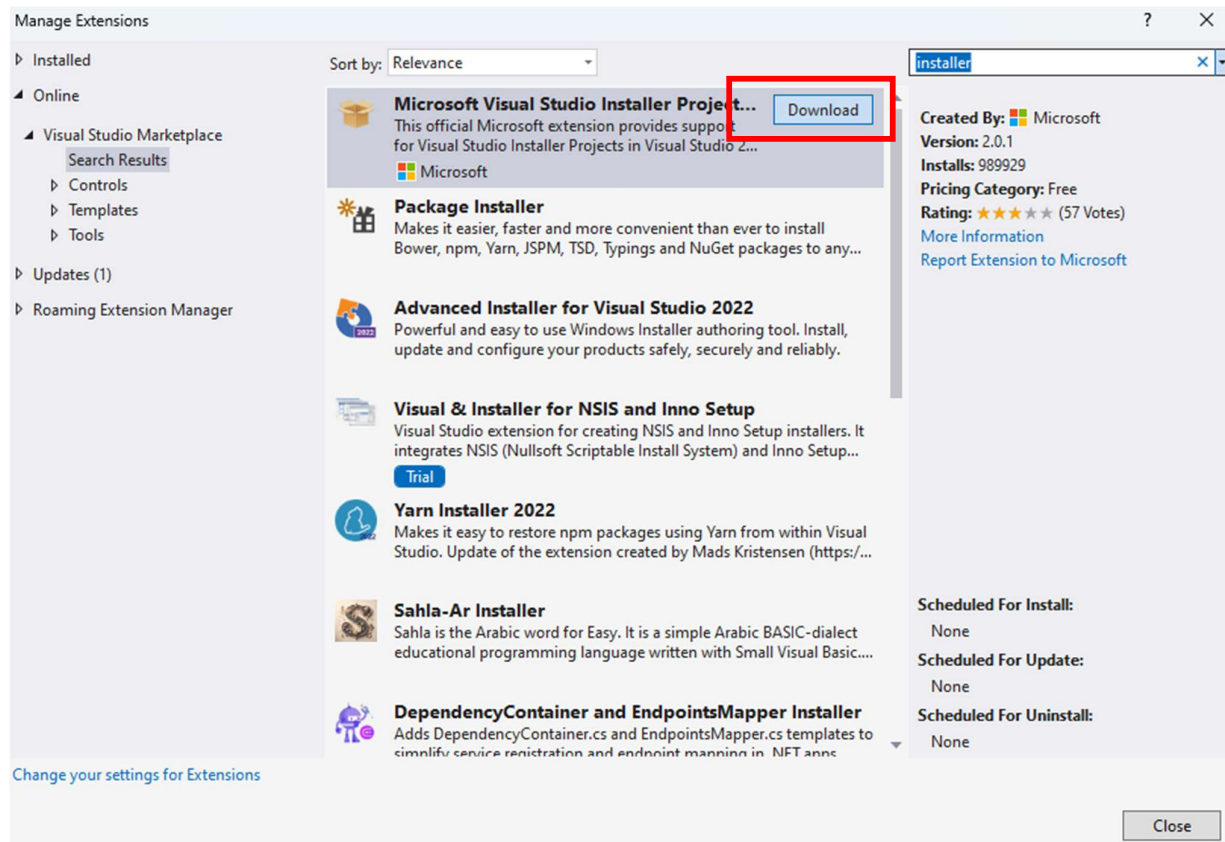
1-2**Distribute the application****■ Installer**

Distribute ML Component along with the application. Copy “MLComponent.dll” and “MLComponent.XmlSerializers.dll” to the same location as the executable file (*.exe) for it to function. This section explains how to create and distribute an installer using a Visual Studio 2022 setup project.

■ Creating an installer in Visual Studio 2022

Select “Extensions” > “Manage Extensions.”

Enter “Installer” in the search box, search for “Microsoft Visual Studio Installer Projects 2022,” and select “Download.”

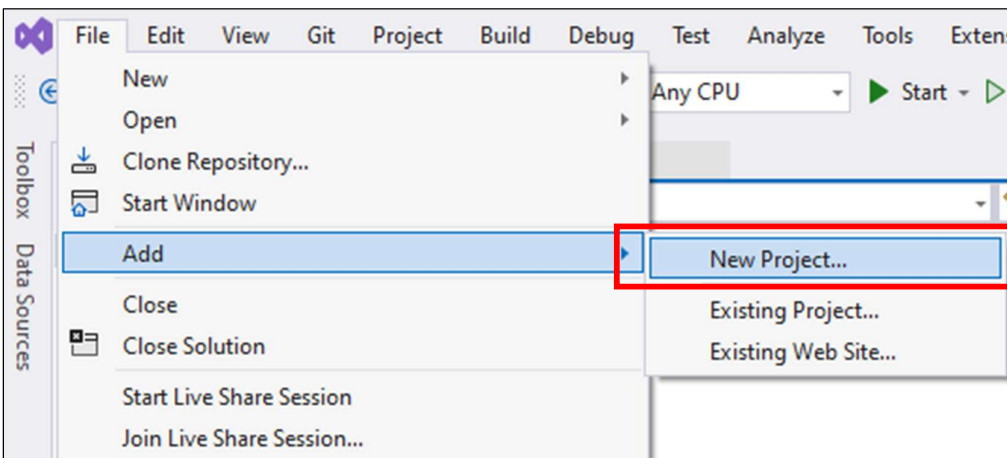


Close the window and exit Visual Studio.

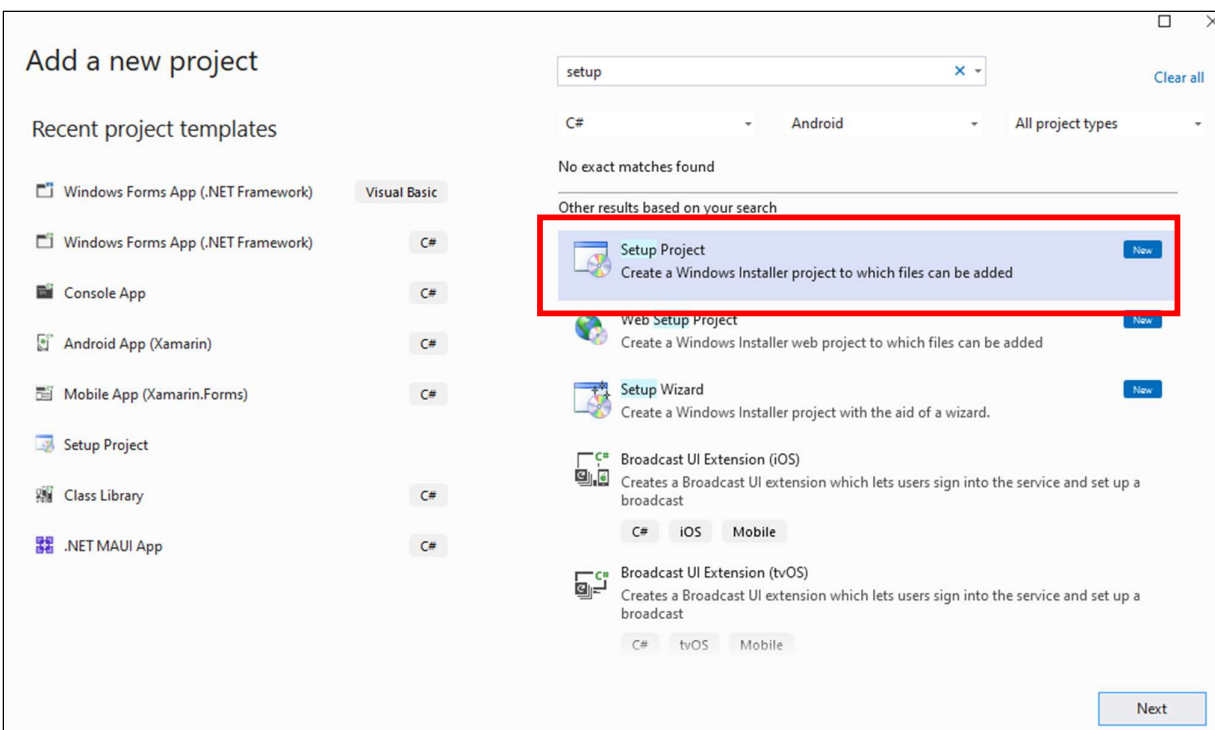
Follow the on-screen instructions to install the extension.

Once the installation is complete, reopen the project.

Select "File" > "Add" > "New Project."



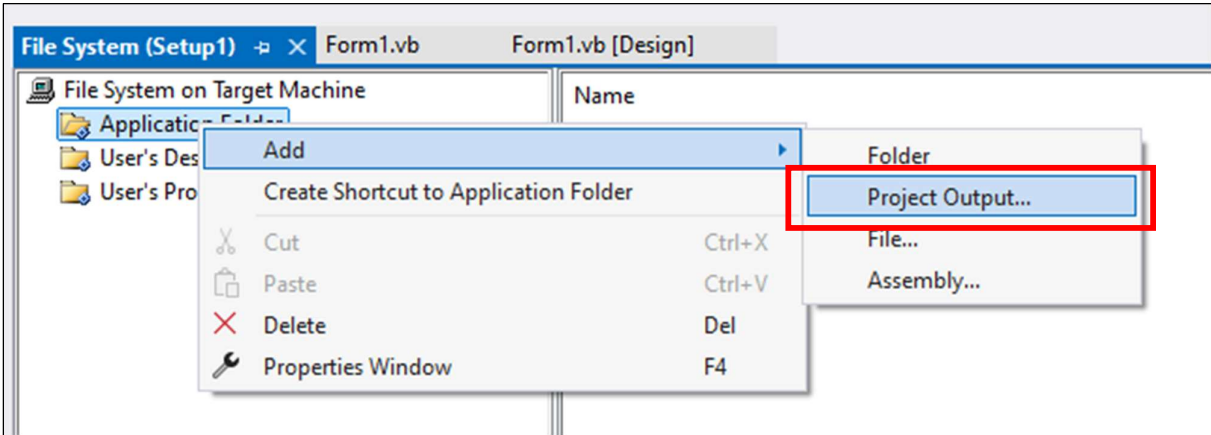
Enter "setup" in the search box, select "Setup Project," and then select "Next."



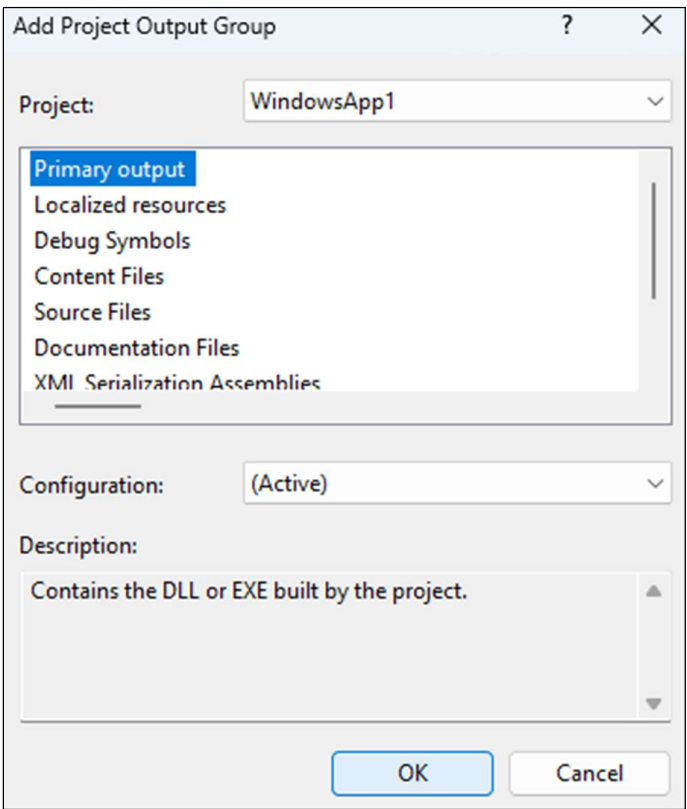
Enter a project name and select "Create."

The setup project will be created.

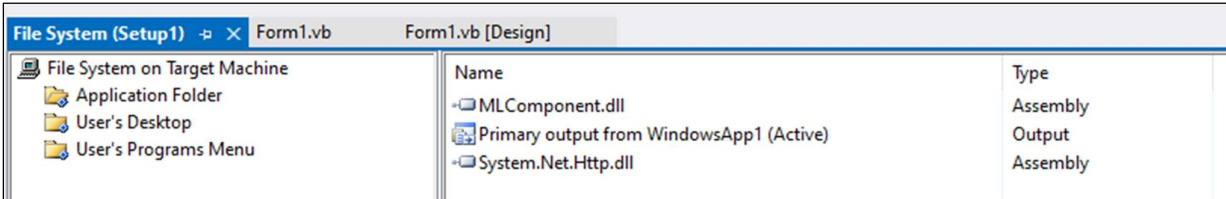
Right-click on "Application Folder," select "Add" > "Project Output."



Select "OK."



"MLComponent.dll" will be added.



1-3**Decide on the publishing method**

■ Issuing method

Label printing is performed by connecting the PC to the printer via a cable and sending the print data to the printer. There are two main methods of printing: “interface output” and “printer driver output.” Select the printing method that best suits your operational needs based on the advantages and disadvantages of each.

For instructions on how to use interface output, refer to “[Connecting to a Printer.](#)” and for Printer Driver Output, refer to “[Using the Printer Driver.](#)”

■ How to print

	Printer driver output	Interface output
Available LAN	LAN (wireless LAN) USB RS-232C	LAN (wireless LAN) USB RS-232C Bluetooth
Development level	Beginner	Intermediate to Advanced
Printer control	Not required (Handled by printer driver)	Required
Advantages	Communication control is performed by the printer driver, making application development easier.	You can check the printer status. No need to install printer drivers.
Disadvantages	Printer drivers must be installed on each PC. The printer status cannot be checked from the application.	Implementation of communication control with the printer is required. It is necessary to understand the communication-related technology and printer specifications, and develop processing to handle various situations such as printer errors and communication failures.

1-4**Connecting to the printer**

■Connection/Disconnection ■USB ■LAN ■COM ■Bluetooth

To issue labels, first connect to the printer. After issuing, always disconnect at the end. Failing to disconnect may result in issues such as being unable to reconnect to the printer or issuing from other PCs when attempting to issue again.

If you want to issue labels easily using the printer driver, please refer to [“Using the Printer Driver.”](#)

■Connection/Disconnection

- Sample code

```
'Create an instance of MLComponent
Dim MLComponent As New SATO.MLComponent.MLComponent

'Processing result
Dim Result As Integer

'Communication settings (LAN connection)
MLComponent.Setting = "LAN:192.168.1.100,1024"

'Connect to printer
Result = MLComponent.OpenPort(1)
If Result < 0 Then
    MessageBox.Show("OpenPortError No." & Result.ToString)
End If

'Disconnect from printer
MLComponent.ClosePort()
```

■ Connect via USB

If there is only one printer to connect, specify "USB:". The program will automatically connect to the first Sato USB printer found. If there are two or more USB printers connected to the PC, specify the USB serial ID of the printer after "USB:". For details on how to confirm the USB serial ID, refer to the "Reference Manual."

Sample code

```
'Connect to the printer connected to the PC
MLComponent.Setting = "USB:"

'Connect to the serial number 0000T123 connected to the PC"
MLComponent.Setting = "USB:0000T123"
```

■Connect via LAN

On a LAN, specify the printer's [IP address] and [port number].

The port number is typically set to "1024" or "9100."

Sample code

```
Connect to the printer using IP address 192.168.1.10 and port number 1024
```

```
MLComponent.Setting = "LAN:192.168.1.10,1024"
```

```
'Connect to the printer using IP address 192.168.1.10 and port number 9100
```

```
MLComponent.Setting = "LAN:192.168.1.10,9100"
```

■Connect via COM (serial port)

For COM, specify the printer's [COM port number], [baud rate], [parity bit], [data bits], and [stop bits]. Match these settings to the printer's configuration.

Sample code

```
Connect to the printer connected to COM19
```

```
MLComponent.Setting = "COM19:115200,n,8,1"
```

■Connect via Bluetooth

When connecting via Bluetooth, specify the printer's [BD Address].

Connectable devices are printers that are paired or Bluetooth Ver.3.0-compatible with the authentication level set to "No Authentication." Bluetooth pairing can be performed using the AuthenticateBluetoothDevice method or the Windows standard settings screen.

Sample code

```
Connect to the printer with BD address 000b5db4aebb
```

```
MLComponent.Setting = "BT:000b5db4aebb"
```


1-5**Check the printer status**

■Communication protocol ■Status check

When issuing via interface output without using a printer driver, be sure to check the printer status before and after issuance. Failure to do so may result in errors occurring on the printer, causing print data to be sent even if there are errors, and labels may not be issued, resulting in data loss.

■Communication Protocol

To communicate with the printer, set the [Communication Protocol]. Ensure it matches the printer's settings.

Communication protocol	Interface			
	USB	LAN	Bluetooth	COM
Status 3	×	○	○	○
Status 4	○	○	○	○

O: Available X: Not available

■Status Check

- Sample code

```

'Create an instance of MLComponent
Dim MLComponent As New SATO.MLComponent.MLComponent

'Processing result
Dim Result As Integer

'Status string
Dim Status As String = ""

'Communication settings (LAN connection)
MLComponent.Setting = "LAN:192.168.1.100,1024"

'Communication protocol settings
MLComponent.Protocol = SATO.MLComponent.Protocols.Status3

'Connect to printer
Result = MLComponent.OpenPort(1)
If Result < 0 Then
    MessageBox.Show("OpenPortError No." & Result.ToString)
    Exit
End If

'Check printer status
Result = MLComponent.GetStatus(Status)
If Result = 0 Then

```

```
        MessageBox.Show("PrinterStatus = " & Status.Substring(2, 1))  
Else  
        MessageBox.Show("OpenPortError No." & Result.ToString)  
End  
  
'Disconnect from printer  
MLComponent.ClosePort()
```

The printer status can be determined by the third digit of the status string obtained. For details on the status and whether transmission is possible, refer to the ["Reference Manual"](#) for the status specifications.

For example, if "A" is obtained, the printer is online and no errors have occurred, so printing can proceed without any problems. If "I" is obtained, the printer is online and printing a label, but the receive buffer capacity is low, so it is recommended to wait a few seconds and then check again. If "g" is obtained, the printer head is disconnected and cannot print normally, so you can take measures such as displaying a warning message prompting the user to replace the printer without printing.

1-6**Using the printer driver****■Connection/Disconnection**

If you want to issue labels easily without needing to monitor the printer's status in detail or check each label individually, using the printer driver is recommended. Simply output data to the printer driver, and no complicated communication control is required.

For more detailed control, please refer to ["Connecting to the Printer."](#)

■Connection/Disconnection

Specify the printer driver in the Setting property as "DRV:". Optionally, specify the **[Printer Driver Name]**. To issue labels, first connect to the printer driver (start output). After issuance is complete, be sure to disconnect (end output).

- Sample Code

```
'Create an instance of MLComponent
Dim MLComponent As New SATO.MLComponent.MLComponent

'Processing result
Dim Result As Integer

'Communication settings (printer driver)
MLComponent.Setting = "DRV:SATO CL4NX-J 305dpi"

'Start output to printer driver
Result = MLComponent.OpenPort(1)
If Result < 0 Then
    MessageBox.Show("OpenPortError No." & Result.ToString)
End If

'Output to printer driver complete
MLComponent.ClosePort()
```

The printer driver creates a "job" container to store data for printing when printing begins (connection), and accumulates the data one piece at a time. The timing for sending data to the printer during data output varies depending on the OS. To send data to the printer immediately, execute ClosePort immediately after the Output method.

1-7**Issuing labels and tags**

■ Label Issuance

Label issuance combines the layout file created in Multi LABELIST V6 with the input data. [Number of copies] is a required input data field; if [Number of copies] is not entered, an error will occur during issuance.

■ Label Issuance

Before issuing, specify the [Layout File Path] and [Input Data]. Input data can be specified in various ways. For details, refer to “[Enter Data.](#)”

- Sample code

```
'Prerequisite: OpenPort has been successfully executed

'Processing results
Dim Result As Integer

'Specify layout file
MLComponent.LayoutFile = "C:\¥sato¥label.mllayx"

'Specify input data (when the input variable is only [Number of copies])
MLComponent.PrnData = "10"

'Issue label
Result = MLComponent.Output()
If Result < 0 Then
    MessageBox.Show("Output Error No." & Result.ToString)
End If
```

One-Point Technique

When issuing multiple labels for the same data, instead of sending “Issue Count = 1” multiple times,

send “Issue quantity = N” (e.g., “Issue quantity = 10” for 10 copies) once. This method allows labels to be issued more quickly.

1-8**Batch input data**

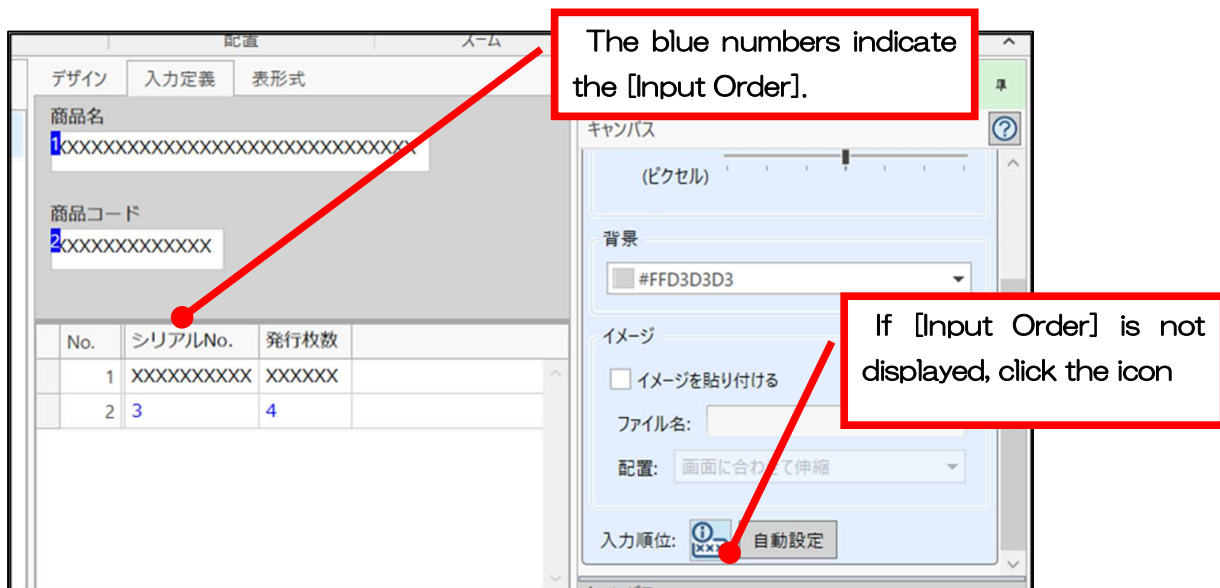
■Input order ■Data format ■Multiple data

Enter data in bulk according to the specified data format. When using a printer driver, you can enter not only a single data set but also multiple data sets in bulk.

If you want to enter data using variable names without considering the order, please refer to [“Enter data using variable names.”](#)

■Input order

First, confirm the order in which data will be entered by displaying [Input Order] in the ML design input definition.



• Sample code

```
'Specify input data
'Enter the product name, product code, serial number, and number of copies in order
MLComponent.PrnData = "Label Printer" & vbTab & "490310999999" & vbTab &
"RX00007802" & vbTab & "3"
```

■Data Format

By default, data is specified in tab-delimited (TSV format), but you can also specify data in comma-delimited CSV format or space-delimited PRN format.

• Sample code (CSV format)

```
'Data format specification (CSV format)
MLComponent.PrnDataType = SATO.MLComponent.PrnDataTypes.Csv

'Specify input data
MLComponent.PrnData = "Label Printer,490310999999,RX00007802,3"
```

'Use double quotes (") to include line break codes.

```
MLComponent.PrnData = ""This printer is a 4-inch rugged printer capable of high-speed printing." & vbCrLf & "", 1"
```

- Sample code (PRN format)

'Data format specification (PRN format)

```
MLComponent.PrnDataType = SATO.MLComponent.PrnDataTypes.Prn
```

'Specify input data

```
MLComponent.PrnData = "Label Printer 490310999999 RX00007802 3"
```

■ Entering multiple data

When using a printer driver, you can input multiple data in bulk.

- Sample code

'Prerequisite: Use a printer driver

'Create an array of strings

```
Dim inputData(0 To 2) As String
```

```
inputData(0) = "Label Printer A,490310999999,RX00007802,1"
```

```
inputData(1) = "Label Printer B,490310123456,MS00000619,1"
```

```
inputData(2) = "Label Printer C,490310000005,FX00000550,1"
```

'Specify multiple input data

```
MLComponent.SetPrnDataArray(inputData)
```

1

Specifying and inputting data by variable name

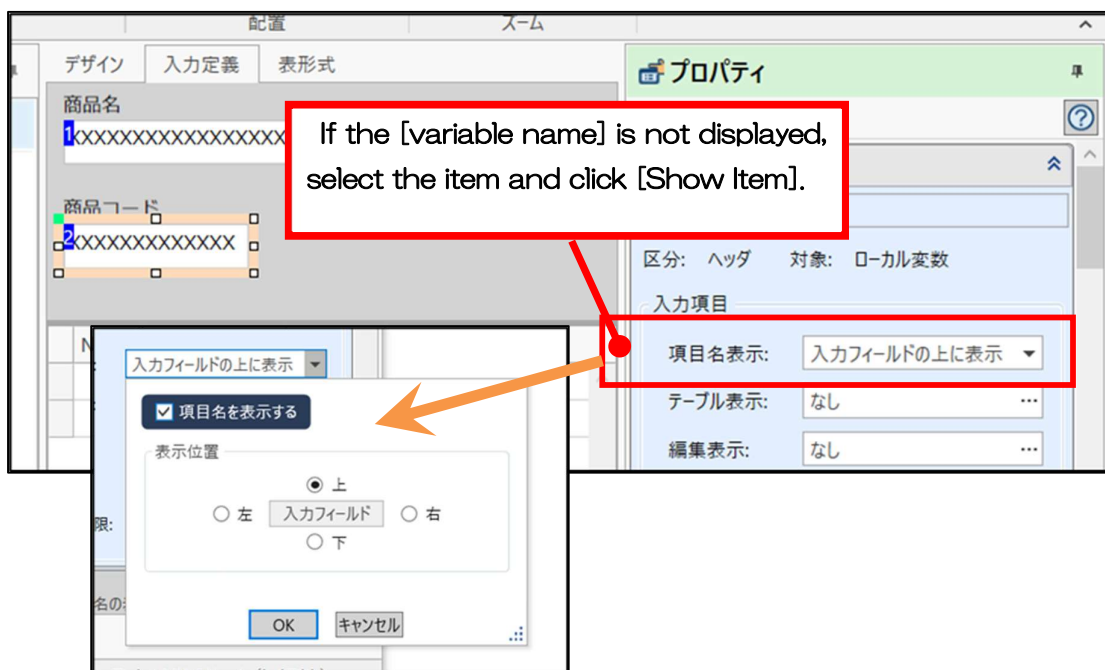
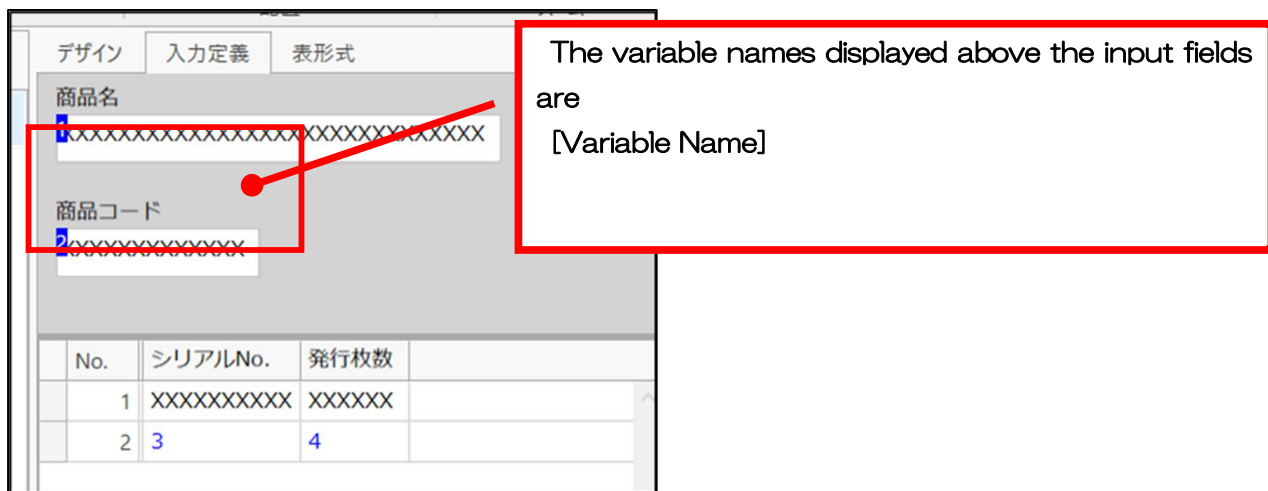
■Variable name

Specify variable names to input data. This allows you to input data without considering the input order of layout files, which is useful when using multiple layout files with common variable names or when the input order may change.

If you want to enter data easily without considering the input order, please refer to [“Enter data in bulk.”](#)

■Variable names

First, confirm the [Variable Name] required for data specification in the ML design input definition.



- Sample code

'Enter the product name

```
MLComponent.SetPrnDataField("Product Name", "Label Printer")
```

'Enter the product code

```
MLComponent.SetPrnDataField("Product Code", "490310999999")
```

'Enter the serial number

```
MLComponent.SetPrnDataField("Serial No.", "RX00007802")
```

'Enter the number of copies

```
MLComponent.SetPrnDataField("Number of Copies", "3")
```


1

Check the version

■ Properties ■ File version

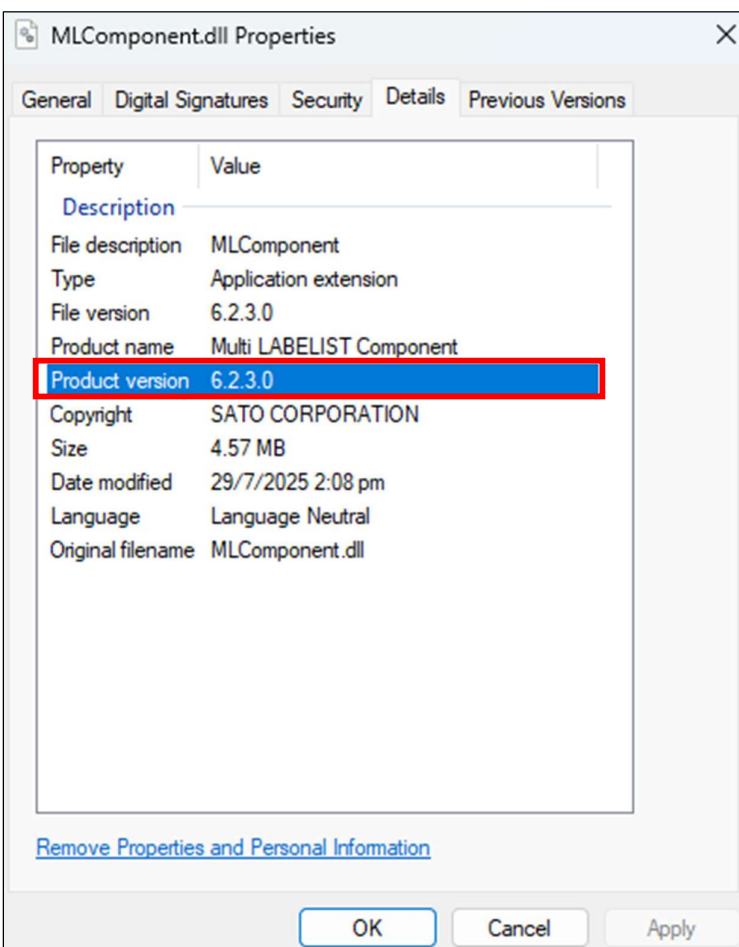
■ Obtaining the Version property

You can confirm the version number from the information obtained from the Version property.

Multi LABELIST Component, Version 6.x.x.x

■ Check using file properties

You can confirm the version number in the file properties of "MLComponent.dll," which is the main file of MLComponent.



1-11**Performing a version upgrade**

■Development Environment ■Execution Environment

■Updating ML Component in the Development Environment

Select the new version of "MLComponent.dll" in "1-1. Use in Visual Studio" to update it.

■Update ML Component in the runtime environment

Simply replace the "MLComponent.dll" file distributed with the executable application with the new version. No rebuild or recompilation is required in the development environment.

If "Symbol Drawing Error" is enabled in the layout file's runtime settings, place "BCD32.dll" and "BCD64.dll" in the same folder as MLComponent.dll.

1-

12

Using with Microsoft Excel/Access (VBA)

■Add reference ■Declaration

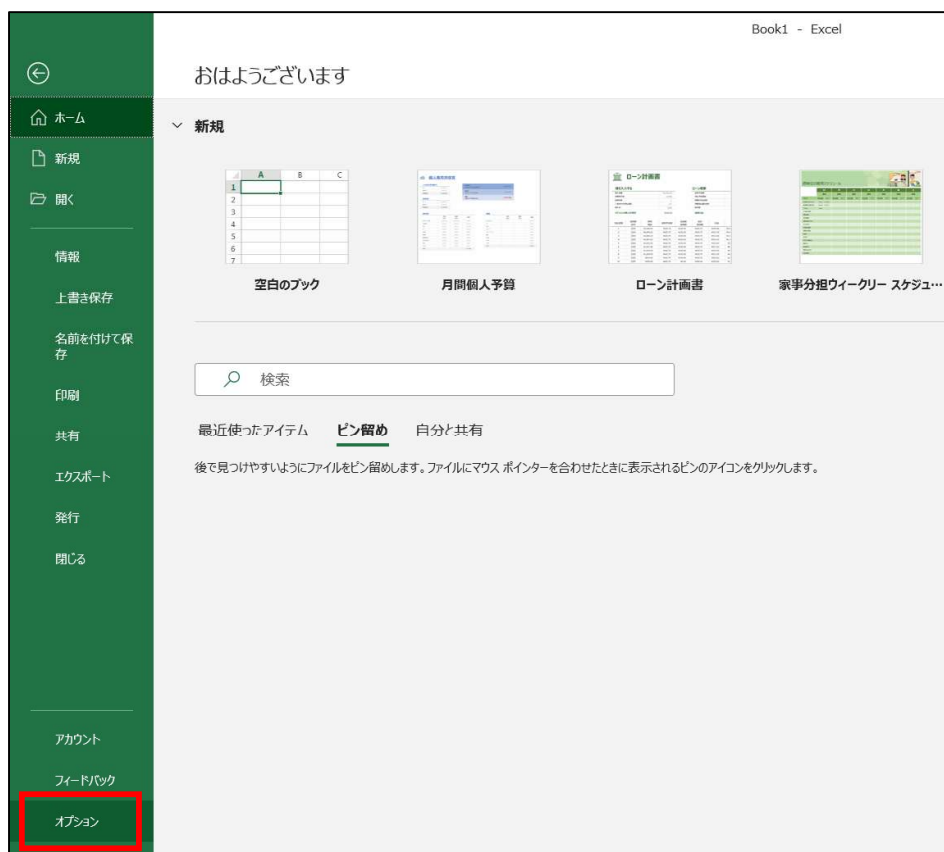
■Installation

To use with Microsoft Excel/Access (VBA), install ML Component in the development environment using the installer. Download the installer from the following page.

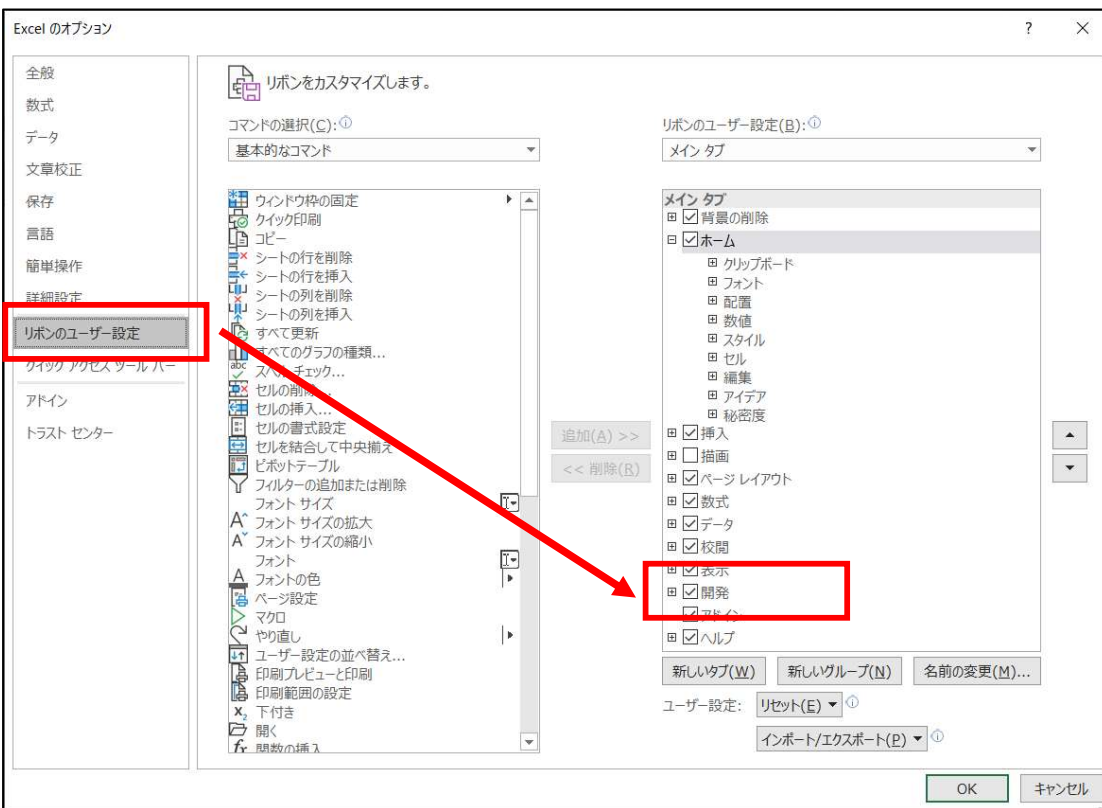
<https://www.sato.co.jp/support/printrtool/tool/multi-labelist-component/>

■Adding References

For Excel, display the “Developer” tab on the ribbon to add references. Select “Options” from the “File” menu.

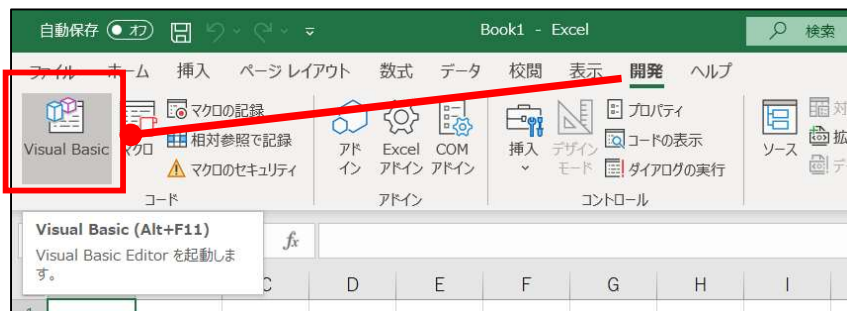


Select “Customize the Ribbon,” check “Developer,” and then click “OK.”

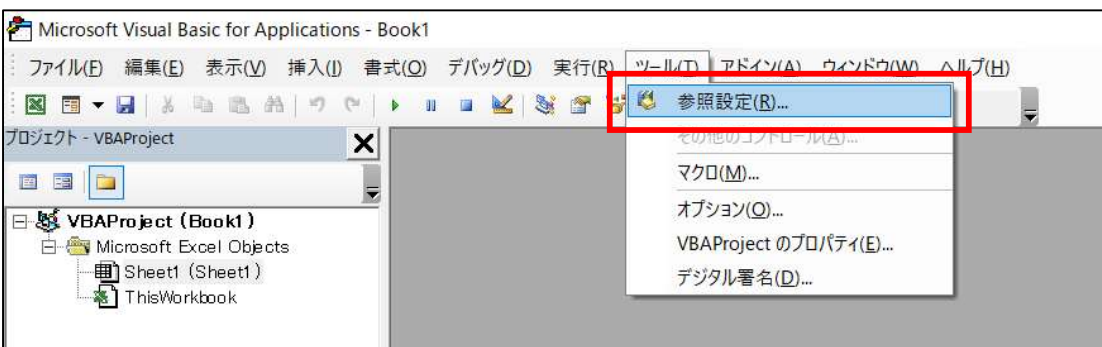


When the “Development” tab appears, configure the reference settings.

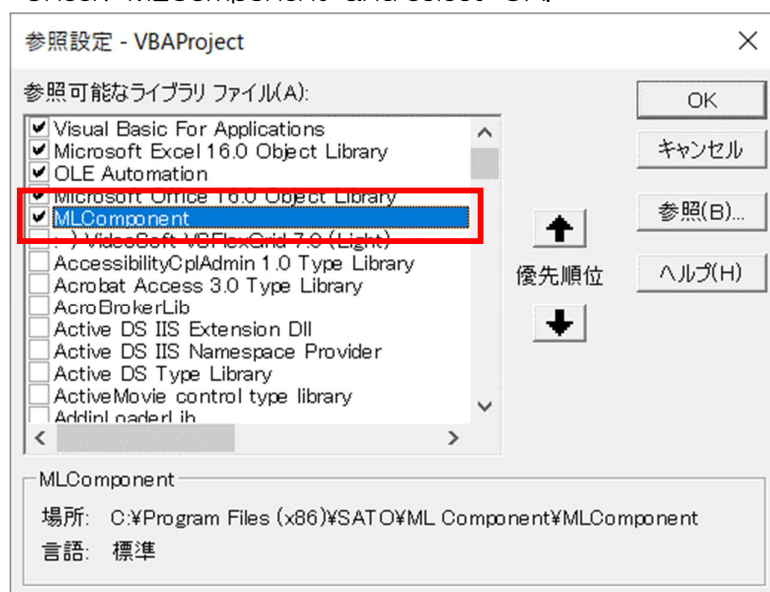
Select “Development” tab > “Visual Basic.” (In Access, select “Database Tools” tab > “Visual Basic.”)



Select “Tools” > “References.”



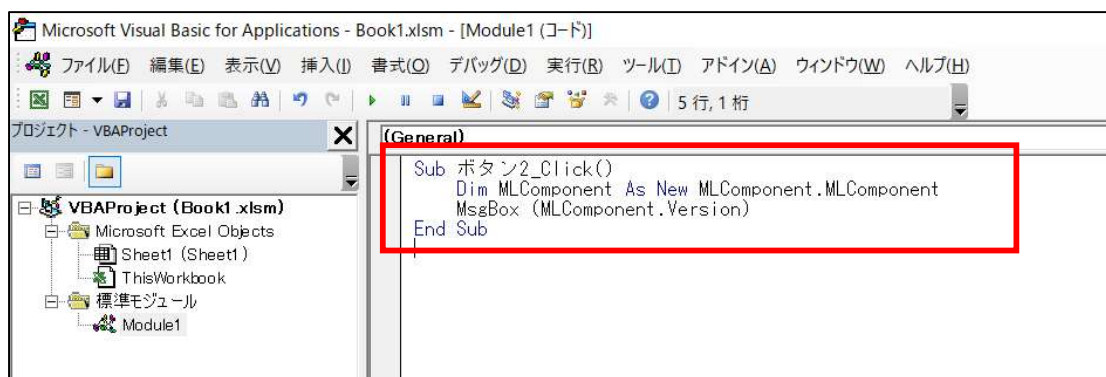
Check "MLComponent" and select "OK."



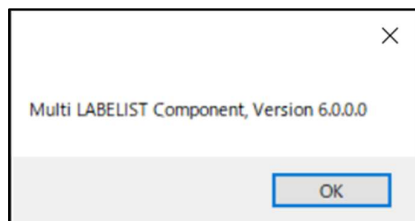
■Instance creation



Click the button on the code to display the MLComponent version in a dialog box.



When executed, the following dialog box will be displayed.



Chapter 2

Advanced

2-1**Changing printer density and speed**

■Density ■Speed

Printer density and speed are typically adjusted on the printer itself. However, in operations where multiple types of labels are used interchangeably, and common settings cannot accommodate all requirements, adjust the settings for each layout using ML Component.

Both density and speed can be specified in three ways.

- ② Use the printer's default density and speed
- ② Use the values specified in MLComponent
- ③ Use the values set in the layout file

■Changing the density

- Sample code

'Use the printer settings
MLComponent.Darkness = ""

'Use the value specified in MLComponent
MLComponent.Darkness = 3

'Use layout settings
MLComponent.Darkness = "S"

■Change speed

- Sample code

'Use printer settings
MLComponent.Speed= ""

'Use the value specified in MLComponent
MLComponent.Speed = 3

'Use layout settings
MLComponent.Speed= "S"

2-2**Adjust print position**

■ Print position

Fine-tune the overall position where printing appears on the label. If the position differs significantly, change the paper size or object position in ML Design.

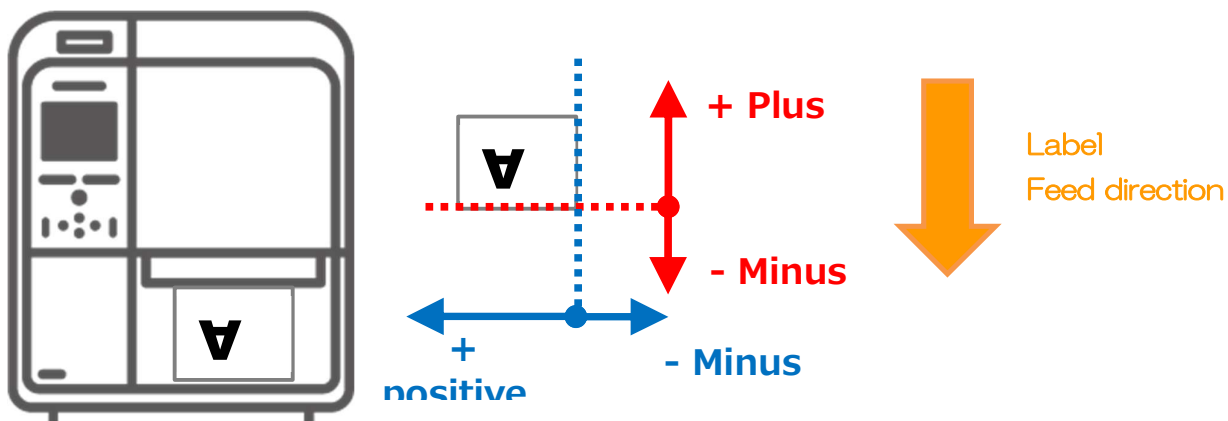
There are two ways to specify this.

- ① Use the values specified in MLComponent
- ② Use the values set in the layout file

■ Adjusting the print position

The print position is negative in the vertical direction of the label feed direction, and positive in the horizontal direction of the label's inner side.

Values can be adjusted in millimeters up to the second decimal place.



• Sample code

- Adjust using the value specified in MLComponent

```
MLComponent.Offset = "1.5,3.25"
```

- Use layout settings

```
MLComponent.Offset = "S,S"
```

2-3

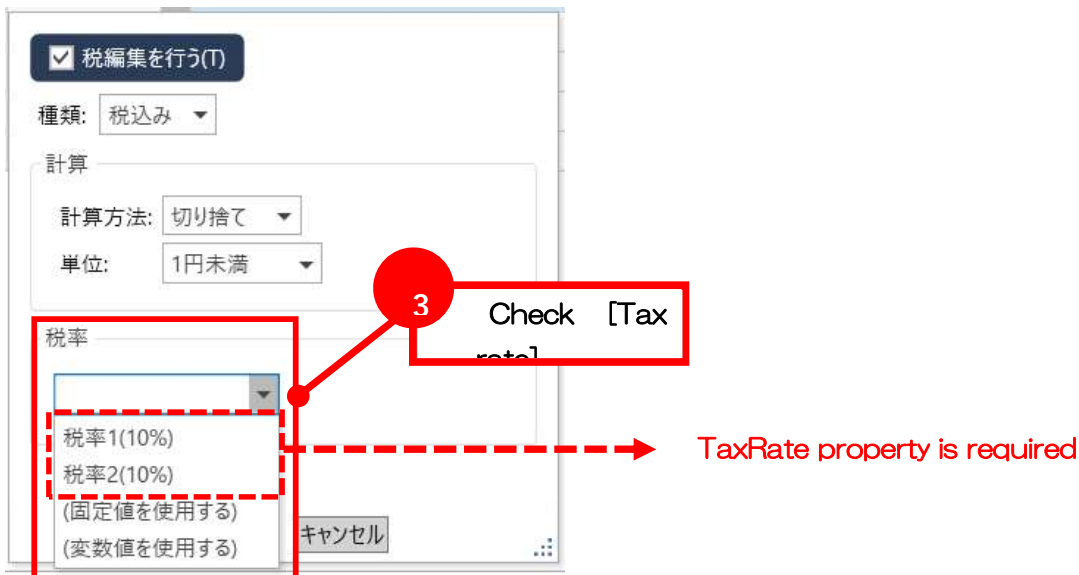
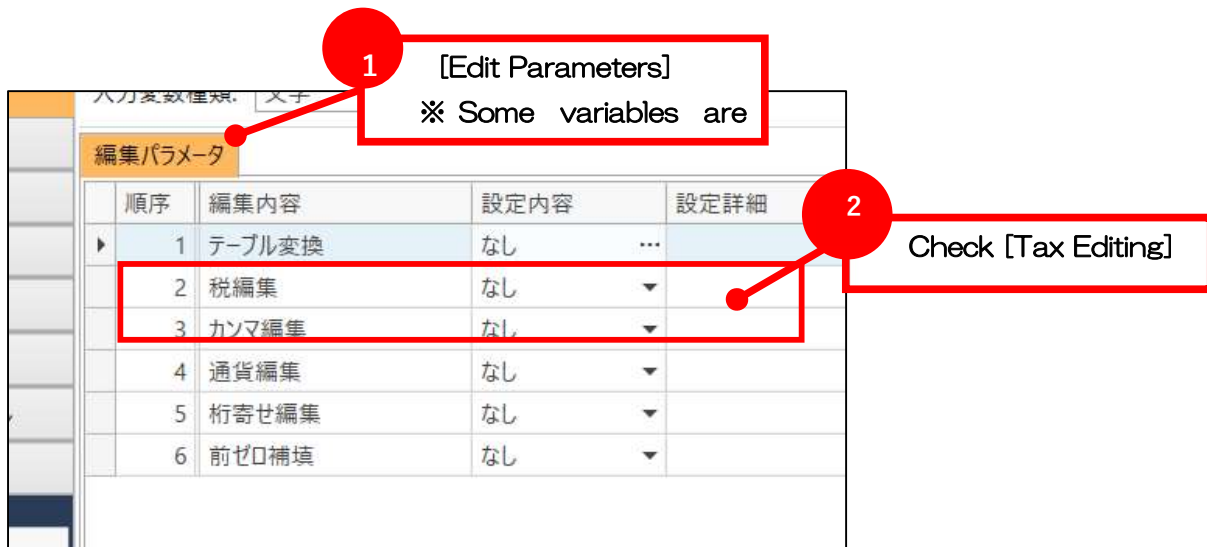
Set sales tax

■Tax Editing ■Sales Tax

When using the [Tax Edit] option in the [Edit Parameters] section of MLV6 variable settings, you must set the consumption tax using the TaxRate property. If consumption tax is not set, an error 413 (An error occurred during tax editing.) will be returned by the Output method at issuance.

■Tax Edit

If [Tax Edit] in [Edit Parameters] is set to [None], and [Tax Edit] is set to [Yes] with [Tax Rate] set to [Use Fixed Value] or [Use Variable Value], there is no need to set the TaxRate property.



■Consumption tax

• Sample code

```
'Prerequisite: OpenPort has been successful

'Processing result
Dim Result As Integer

'Specify layout file
MLComponent.LayoutFile = "C:\¥sato¥price,mllayx"

'Specify input data (for example, entering "100" for price)
MLComponent.PrnData = "Green tea" & vbTab & "100" & vbTab & "2"

'Specify consumption tax
MLComponent.TaxRate = "10"

'Label issuance
Result = MLComponent.Output()
If Result < 0 Then
    MessageBox.Show("OutputError No." & Result.ToString)
End If
```

**Green
tea
¥110**

**Green tea
¥110**

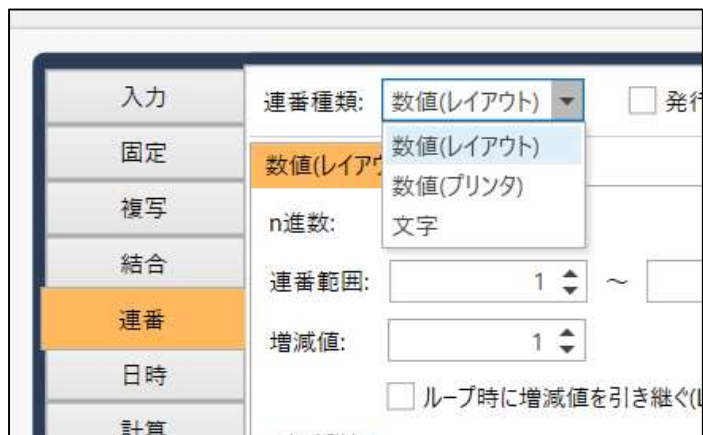
2-4

Print serial numbers

■ Sequential number ■ Initial value

You can print sequential numbers that can be used for product serial numbers or label identification.

Sequential numbers use the sequential number variable in MLV6. Typically, the “Numeric (Layout)” option, which uses layout information, is used. If sequential number values do not need to be saved and you want to print using a printer font, select “Numeric (Printer).” If you want to print sequential numbers using special patterns or characters, you can also select “Character.”



■ Print serial numbers

No special operations are required in MLComponent to print sequential numbers.

- Sample code

```
'Prerequisite: OpenPort has been successful

'Processing result
Dim Result As Integer

'Specify the layout file using the sequential number variable
= "C:¥sato¥count.mllayx"

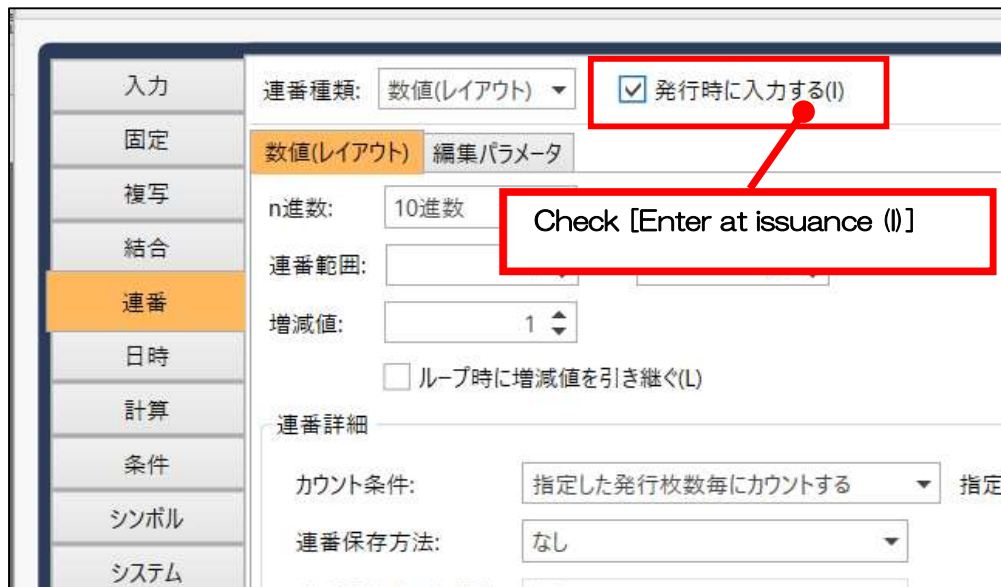
'Specify input data (when the input variable is only [Number of copies])
MLComponent.PrnData = "10"

'Label issuance
Result = MLComponent.Output()
If Result < 0 Then
    MessageBox.Show("Output Error No." & Result.ToString)
End If
```

■Enter the initial value for sequential numbers

If the start value or end value of the sequential number is managed on the application side, enable the input of the initial value of the sequential number as data each time it is issued.

Check the [Enter at issuance (I)] box in the sequential number settings screen.



- Sample code

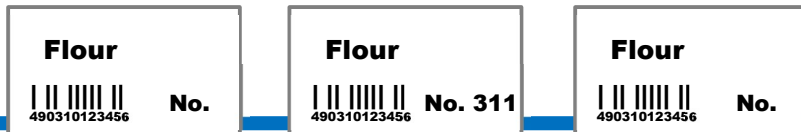
```
'Prerequisite: OpenPort has been successful

'Processing result
Dim Result As Integer

'Specify the layout file using a sequential variable
MLComponent.LayoutFile = "C:¥sato¥count.mllayx"

'Specify input data (issue 3 sheets starting from the initial value "310")
MLComponent.PrnData = "Flour" & vbTab & "490310123456" & vbTab & "310" &
vbTab & "3"

'Label issuance
Result = MLComponent.Output()
If Result <> 0 Then
    MessageBox.Show("Output Error No." & Result.ToString)
End If
```



2

Issue header/tail tags

■ Header/Tail Tags

Issues header tags and tail tags set in the layout file. System variables commonly used with header and tail tags, such as [Total Number of Issues] and [Layout Name], can also be set from MLComponent.

■ Issuing Header and Tail Tags

- Sample code

```
'Prerequisite: OpenPort has been successful

'Processing result
Dim Result As Integer

'Specify layout file
MLComponent.LayoutFile = "C:¥sato¥price.mllayx"

'Specify input data (including data for header and tail tags)
MLComponent.PrnData = ""
    "Jacket" & vbTab & "19000" & vbTab & "490310041310" & vbTab & "Tokyo
Main Store" & vbTab & "3"

'Consumption tax specification (when using tax editing)
MLComponent.TaxRate = "10"

'System variable [Total number of issues] specification
MLComponent.TotalQtyCaption = "3"

'System variable [Layout name] specification
MLComponent.LayoutNameCaption = "Price Tag"

'Issue header tag
Result = MLComponent.OutputHeader
If Result < 0 Then
    MessageBox.Show("OutputHeaderError No." & Result.ToString)
    Exit
End If

'Issue body tag
Result = MLComponent.Output()
If Result &lt;&gt; 0 Then
    MessageBox.Show("OutputError No." & Result.ToString)
    Exit Sub
End If

'Issue tail tag
Result = MLComponent.OutputTail()
If Result &lt;&gt; 0 Then
    MessageBox.Show("OutputTailError No." & Result.ToString)
    Exit Sub
End If
```



■ Issue header and tail tags according to layout settings

When using printer driver output, you can combine this feature with the function to issue multiple data in batches to automatically issue header and tail tags according to the layout settings.

- Sample code

```
'Prerequisite: OpenPort has been successful

'Process result
Dim Result As Integer

'Specify layout file
MLComponent.LayoutFile = "C:¥sato¥price.mllayx"

'Create an array of strings (including data for header and tail tags)
Dim inputData(0 To 2) As String
inputData(0) = "Jacket A" & vbTab & "19000" & vbTab & "490310041310" & vbTab &
"Tokyo Main Store" & vbTab & "2"
inputData(1) = "Jacket B" & vbTab & "15000" & vbTab & "490310841310" & vbTab &
"Tokyo Main Store" & vbTab & "2"
inputData(2) = "Jacket C" & vbTab & "9500" & vbTab &
"490310413108" & vbTab & "Tokyo Main Store" & vbTab & "2"

'Specify multiple input data
MLComponent.SetPrnDataArray(inputData)

'Specify consumption tax (when using tax editing)
MLComponent.TaxRate = "10"

'Specify system variable [Total number of issued items]
MLComponent.TotalQtyCaption = "6"

'Specify system variable [Layout Name]
MLComponent.LayoutNameCaption = "Price Tag"

'Specify automatic issuance of header and tail tags
MLComponent.HeaderTailSetting = True
```



```
'Tag issuance
Result = MLComponent.Output()
If Result < 0 Then
    MessageBox.Show("OutputError No." & Result.ToString)
    Exit
End If
```



2-6**Using multi-faceted labels**

■ Multi-panel

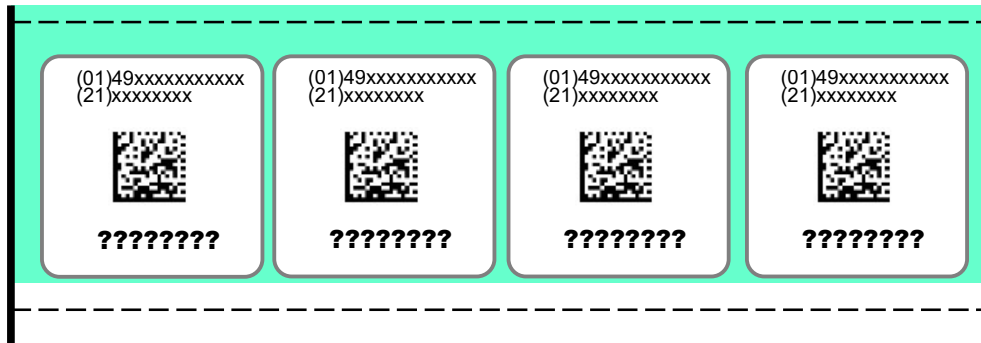
Use multi-panel labels, which consist of multiple labels arranged on a single sheet (1 sheet). Please note that the number of copies that can be entered varies depending on the output method.

In the case of interface output, only the number of sheets can be specified. For example, if you enter "6" as the number of sheets for a 4-panel label and issue the command, an error 801 will be returned via the Output method.

For printer driver output, there is no limit to the number of copies that can be issued. By combining this with the specification of multiple data sets, you can issue a large number of labels with a single command.

■ Issuing multiple-sided labels by entering one sheet's worth

Example: For a label with 4 panels per sheet



• Sample code

```
'Prerequisite: OpenPort has been successful
```

```
'Processing result
```

```
Dim Result As Integer
```

```
'Specify layout file
```

```
MLComponent.LayoutFile = "C:¥sato¥sheet.mllayx"
```

```
'Create an array of strings (total of 4 for the number of issues)
```

```
Dim inputData(0 To 2) As String
```

```
inputData(0) = "PRINTER-A" & vbTab & "490310999999" & vbTab & "31007802" &  
vbTab & "2"
```

```
inputData(1) = "PRINTER-B" & vbTab & "490310123456" & vbTab & "31000619" &  
vbTab & "1"
```

```
inputData(2) = "PRINTER-C" & vbTab & "490310000005" & vbTab & "31000550" &  
vbTab & "1"
```

'Specify multiple input data

MLComponent.SetPrnDataArray(inputData)

'Issue label

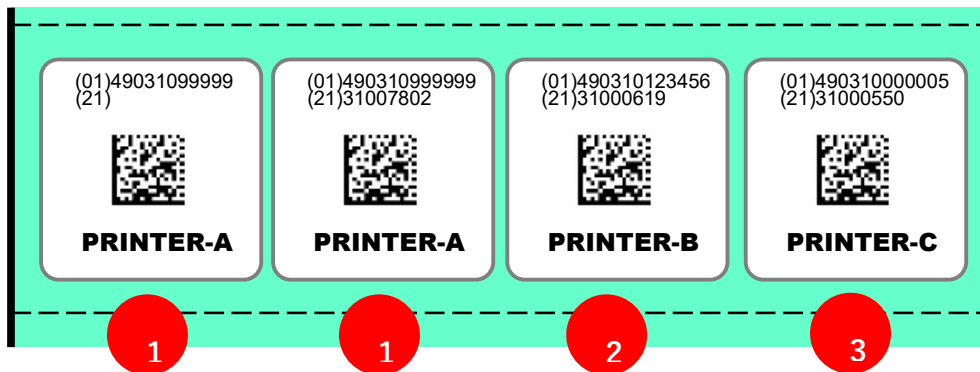
Result = MLComponent.Output()

If Result < 0 Then

 MessageBox.Show("OutputError No." & Result.ToString)

 Exit

End If



①inputData(0) ②inputData(1) ③inputData(2)

■Enter multiple sheets of multi-faceted data and issue (printer driver output only)

- Sample code

'Prerequisite: OpenPort has been successful

'Processing result

Dim Result As Integer

'Specify layout file

MLComponent.LayoutFile = "C:¥sato¥sheet.mllayx"

'Create an array of strings (total of 6 for the number of issues)

Dim inputData(0 To 3) As String

inputData(0) = "PRINTER-A" & vbTab & "49031099999" & vbTab & "31007802" & vbTab & "1"

inputData(1) = "PRINTER-B" & vbTab & "490310123456" & vbTab & "31000619" & vbTab & "2"

inputData(2) = "PRINTER-C" & vbTab & "490310000005" & vbTab & "31000550" & vbTab & "3"

inputData(3) = "PRINTER-D" & vbTab & "490310041310" & vbTab & "31000100" & vbTab & "1"

'Specify multiple input data

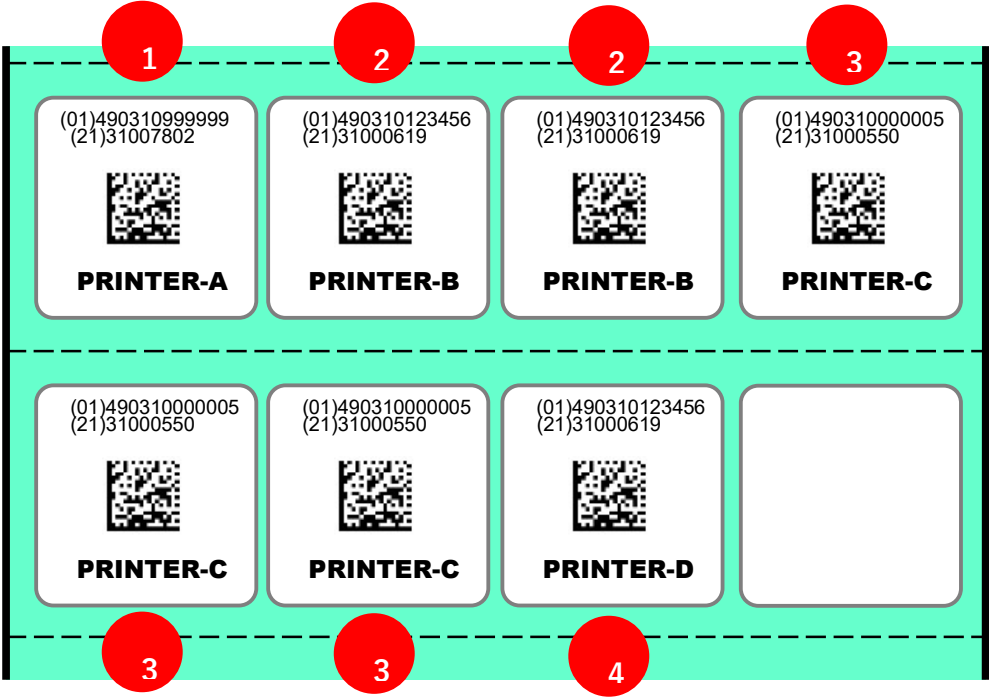
MLComponent.SetPrnDataArray(inputData)

'Issue label

Result = MLComponent.Output()

If Result < 0 Then

```
        MessageBox.Show("OutputError No." & Result.ToString)
    Exit
End If
```



①inputData(0) ②inputData(1) ③inputData(2) ④inputData(3)

2

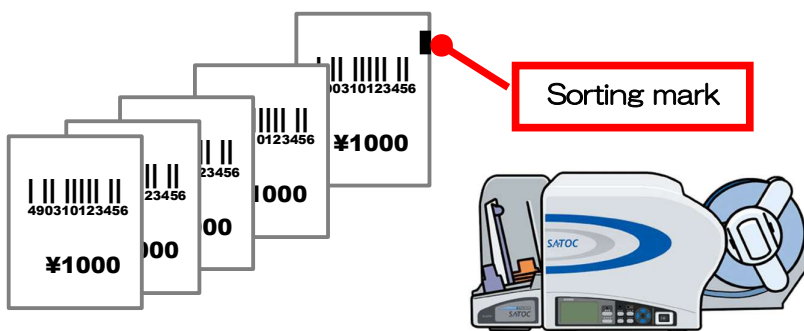
Print sorting marks

■Sorting marks

When issuing multiple types of tags, print sorting marks to clearly indicate the type. Please check the supported printers in [the Reference Manual](#).

■Printing sorting marks

Printed on the side of the leading tag. Setting the stacker allows for more effective operation.



• Sample code

```
'Processing result
Dim Result As Integer

'Use sorting mark
MLComponent.SortMark = True

'Issue label
Result = MLComponent.Output()
If Result < 0 Then
    MessageBox.Show("OutputError No." & Result.ToString)
End If
```

2

Cut tags and labels

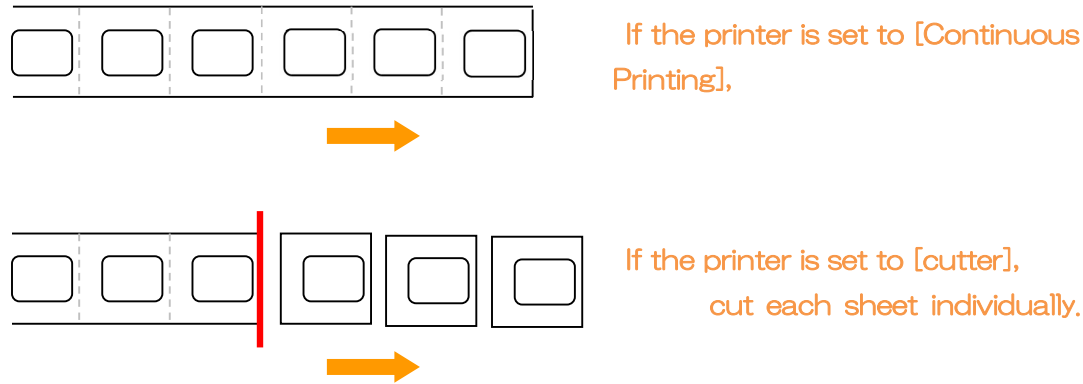
■Cut

You can cut tags and labels at any time. There are various cutting methods available, so please select the most suitable method according to your operational needs. Please refer to the “[Reference Manual](#)” for supported printer models.

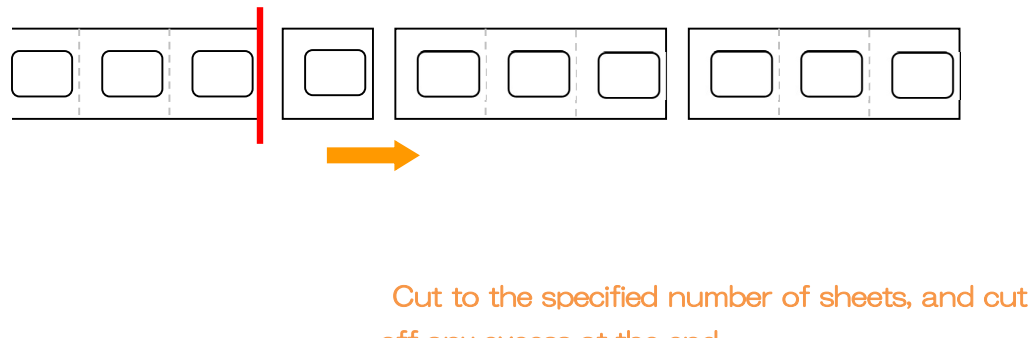
■Performing the cut

Cutting method	Setting Method	
	MultiCut Properties	EjectCut property
Do not cut (default value)	0	False
Follow printer operation mode	-	Not required
Cut to the specified number of sheets	Specify number of sheets	Not required
Cut for each issue instruction		True
Follow layout settings	-	Not required

Follow the printer’s operation mode
Example: When the number of copies is 3

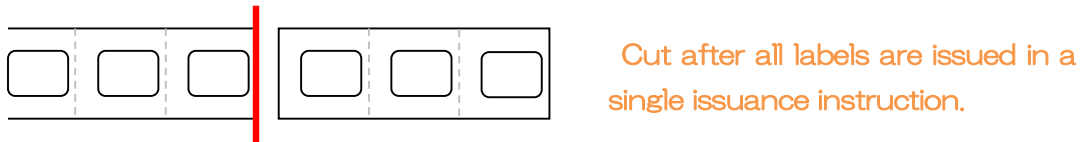


- Cut to the specified number of sheets
Example: If the number of sheets to be issued is 7 and the specified number is 3



Cut for each issue instruction.

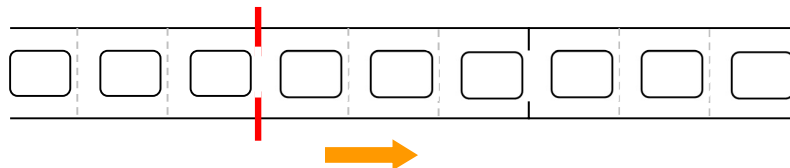
Example: When the number of labels to be issued is 3



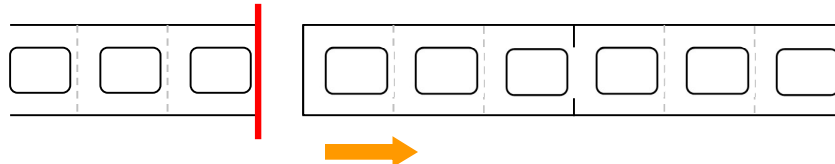
- Cut the last label completely with a partial cut (only for models that support partial cutting)

Set to partial cut mode, and if you want to cut the last label completely, set the EjectCut property to "True."

Example) When using only MultiCut (partial cut)



Example: When using EjectCut (partial cut, full cut)



- Sample code

```
'Do not cut
MLComponent.MultiCut = 0
MLComponent.EjectCut = False

'Follow printer operation mode
MLComponent.MultiCut = -1

'Cut every 5 sheets
MLComponent.MultiCut = 5

'Cut after each print command
MLComponent.MultiCut = 0
MLComponent.EjectCut = True

'Follow layout settings
MLComponent.MultiCut = -2
```

2-9**Cancel label issuance**

■ Cancel issuance

Clear all data sent to the printer and stop label printing.

■ Cancel issuance

- Sample code

```
'Prerequisite: OpenPort has been successful
```

```
'Process result
```

```
Dim Result As Integer
```

```
'Cancel issuance
```

```
Result = MLComponent.SendCancel()
```

```
If Result < 0 Then
```

```
    MessageBox.Show("SendCancelError No." & Result.ToString)
```

```
Exit Sub
```

```
End If
```


2-
10**Send printer command (SBPL)**

■Command transmission ■Command reception

Sends printer commands (SBPL: SATO Barcode Printer Language) to the printer. Please ensure you fully understand the SBPL specifications before use. Sending incorrect SBPL commands may cause command errors on the printer, interrupt label printing, or result in unexpected issues. For details on SBPL, please consult your sales representative.

■Command transmission

SBPL can be sent as either a string type or a byte array type.

Use the SendStringData method for string type and the SendRawData method for byte array type. The SendRawData method is used to send binary data that cannot be represented as a string, such as NULL (hexadecimal code: 00).

■Command Reception

SBPL includes commands that return data such as printer version information and operation settings.

The returned data can be retrieved as a string, byte array, or hexadecimal code. If the data contains binary data that cannot be represented as a string (e.g., NULL [hexadecimal code: 00]), retrieve it as a byte array or hexadecimal code.

Example: If the returned data is "13.00.03.00"

String : "13.00.03.00"

Byte array : {31h, 33h, 2Eh, 30h, 30h, 30h, 2Eh, 30h, 33h, 2Eh, 30h, 30h}

Hexadecimal character code : "31332E30302E30332E3030"

■Send SBPL as a string

- Sample code

```
'Prerequisite: OpenPort has been successful

'Return data
Dim Result As String

'Create SBPL as a string (DC2+PG: printer status information acquisition command)
Dim printerCommand As String
printerCommand = Chr(&H12) & "PG"

'Send SBPL (receive end character: ETX, receive as string)
Try
    Result = MLComponent.SendStringData(0, printerCommand, 0, Chr(&H3))
Catch ex As SATO.MLComponent.MLComponentException
    MessageBox.Show("SendStringDataError No." & ex.Number.ToString)
Exit
```

```
End Try
MessageBox.Show("SendStringData Result=" & Result)
```

■ Send SBPL as a byte array

- Sample code

'Prerequisite: OpenPort must be successful.

'Return data

```
Dim Result As String
```

'Create SBPL as a byte array (DC2+ , PG: printer status information acquisition command)

```
Dim printerCommand(0 To 2) As Byte
```

```
printerCommand(0) = &H12
```

```
printerCommand(1) = &H50
```

```
printerCommand(2) = &H47
```

'Send SBPL (receive end character: ETX, receive in hexadecimal code)

```
Try
```

```
    Result = MLComponent.SendRawData(2, printerCommand, 0, Chr(&H3))
```

```
Catch ex As SATO.MLComponent.MLComponentException
```

```
    MessageBox.Show("SendRawDataError No." & ex.Number.ToString)
```

```
    Exit
```

```
End Try
```

```
MessageBox.Show("SendRawData Result=" & Result)
```

2-11

Using the operation settings file

■ Operation Settings File

By placing the operation settings file in the same folder as MLComponent, you can use advanced settings that cannot be configured via properties. If no setting value (XML tag) is specified, the default value is used.

■ File Name

MLComponentSettings.xml

■ Storage location

Same folder as MLComponent.dll

■ Character encoding

Unicode (UTF-8)

■ Format (example)

```
<?xml version="1.0"?>
< MLComponentSettings xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance"
                        xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  < IsLog>false</IsLog>
  <LogFolder>C:\Sato\Logs</LogFolder>
  < IsSheetCountError > false </IsSheetCountError>
  < TaxRate > 8.0,8.0 </TaxRate>
  < AlternativeFont > false </AlternativeFont>
  < DesignDefaultWindowsFontName> MS Gothic</DesignDefaultWindowsFontName>
  < DesignDefaultWindowsFontSize > 9 </DesignDefaultWindowsFontSize>
  <IsFileForwardCompatibleError>true</IsFileForwardCompatibleError>
  < IsEnableGcCollectForced>false</IsEnableGcCollectForced>
  < GcCollectForcedThreshold>0</GcCollectForcedThreshold>
  < IsWindowsFontTraceLog>false</IsWindowsFontTraceLog>
  <WindowsFontDrawingNumberOfTimes>0</WindowsFontDrawingNumberOfTimes>
  < WindowsFontTraceLogFolder></WindowsFontTraceLogFolder>
</MLComponentSettings>
```

Sample operation settings files are available on the software download site.

<https://www.sato.co.jp/support/printrtool/tool/multi-labelist-component/>

IsLog (Enable/Disable log output)

Sets whether to output log files.

true	Outputs log files.
false (default)	Does not output log files.

LogFolder (log output destination)

Specifies the destination folder for log files in full path format.
Used when IsLog is enabled.

IsSheetCountError (Enable/Disable sheet count error)

Specifies whether to treat an error when the number of copies is not specified. **For MLOCX compatibility settings, do not change this setting and specify the number of copies in the print data.**

true (default)	If the number of copies is not specified in the PrnData property or PrnDataArray method, the Output method will return error No. 802.
false	Even if the number of copies is not specified, no print error is generated, and the printer command is sent with the number of copies set to zero. This may cause issues such as the cut operation not functioning. Do not use this setting normally.

TaxRate (Tax rate setting)

Sets the tax rate used for tax editing. Refer to the TaxRate property for the setting value. This is a default value provided for emergency use. **Typically, specify the value using the TaxRate property.**

AlternativeFont (Enable/Disable Alternative Font Use)

Sets whether to use an alternative font when the Windows font used in the layout is not available.

true	Alternative fonts will be used. Printing will use a different font than the one specified during design, resulting in differences in character shape, size, and automatic line break positions. Typically, install the corresponding font in the production environment or make other adjustments such as changing the font.
false (default)	The Output method will return error No. 600.

DesignDefaultWindowsFontName (default font settings)

Sets the font to be used as the default font for alternative fonts in Windows font names. If this setting is not specified, the OS default font will be used.

DesignDefaultWindowsFontSize (Default font size settings)

Sets the font size of the default font used for alternative fonts in points. If this setting is not specified, the OS default font size will be used.

IsFileForwardCompatibleError (Enable/Disable file version check)

Specifies whether to issue an error when the layout file version is newer than MLComponent.	
true (default)	An issuance error (No. 61-66) will be generated.
false	No error occurs. Depending on the layout, there is a risk that printing may not be performed correctly. If you disable file version checking, please exercise caution when adding or editing layouts, and ensure that printing is performed correctly before putting the system into operation.

IsEnableGcCollectForced (Memory automatic release enabled/disabled)	
When the application's memory usage reaches the threshold, garbage collection (GC.Collect) is executed to free memory. The threshold is specified by GcCollectForcedThreshold.	
true	Enables automatic memory release.
false (default)	Does not perform automatic memory release.

GcCollectForcedThreshold (Memory automatic release threshold setting)	
Sets the threshold (application memory usage) for memory release.	
0 (default)	Executes memory release each time it is issued. Since this may cause delays in issuance, adjust the value appropriately based on the application's memory usage.
1-2048 (MB)	Executes memory release when the memory usage (in MB) of the application incorporating MLComponent exceeds the specified value.

■Output log files

By enabling log output in the operation settings file, logs can be output each time communication, issuance, or printer control methods are executed.

Log files are generated with the filename "MLComponent_*.log" (*GUID) when MLComponent is loaded, and logs are output when the method is called. When MLComponent is unloaded, any open files are closed.

No checks are performed for folder existence, permissions, or disk space when logging. Additionally, logged files are not automatically deleted, so please delete them manually or via your application.

• Format

20XX/11/25

10:00:00:123<tab>OpenPort<tab>O<tab>COM1:9600,n,8,1...

①

②

③

④

① Method execution date and time

YYYY/MM/DD HH:MM:SS:MMM

② Method name

Communication OpenPort, ClosePort

Issue Output, OutputHeader, OutputTail, SendStringData,

SendRawData

Printer control GetStatus, Cut, SendCancel

③ Method return values

Return value indicating the result of method execution (0: normal termination, anything other than 0 indicates an error)

SendStringData and SendRawData return exception error numbers

④ Additional information

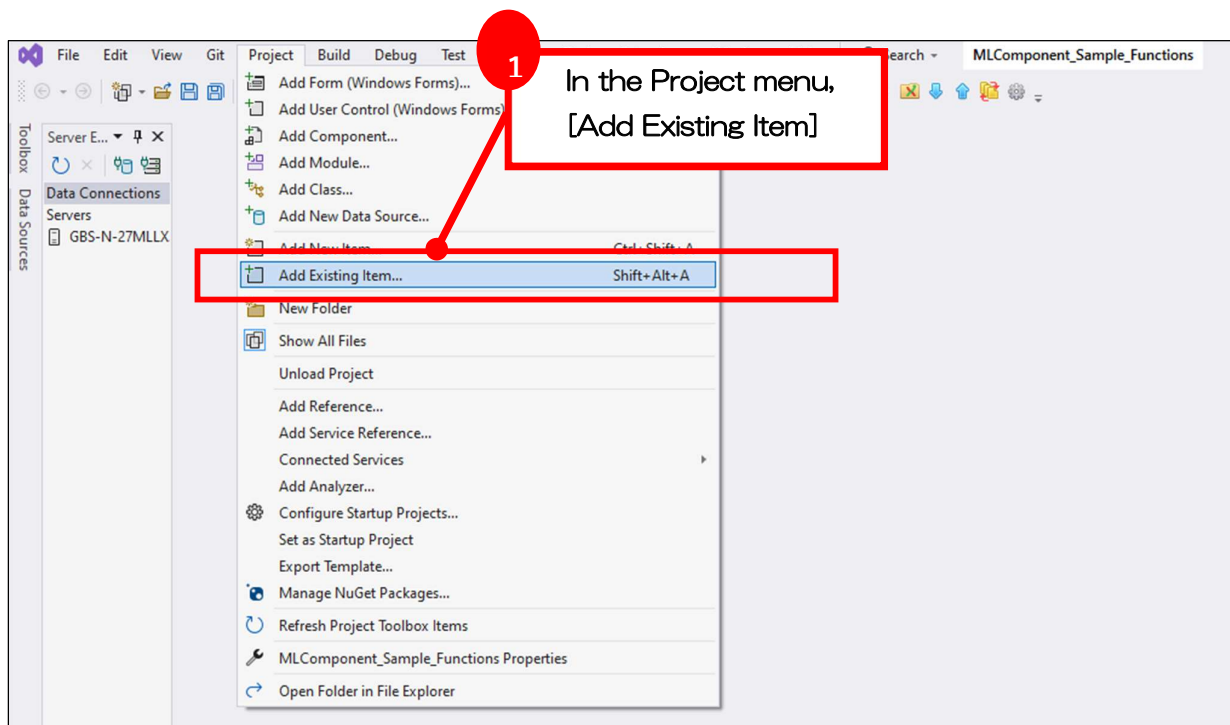
The values of properties and statuses output for each method are appended in tab-delimited format.

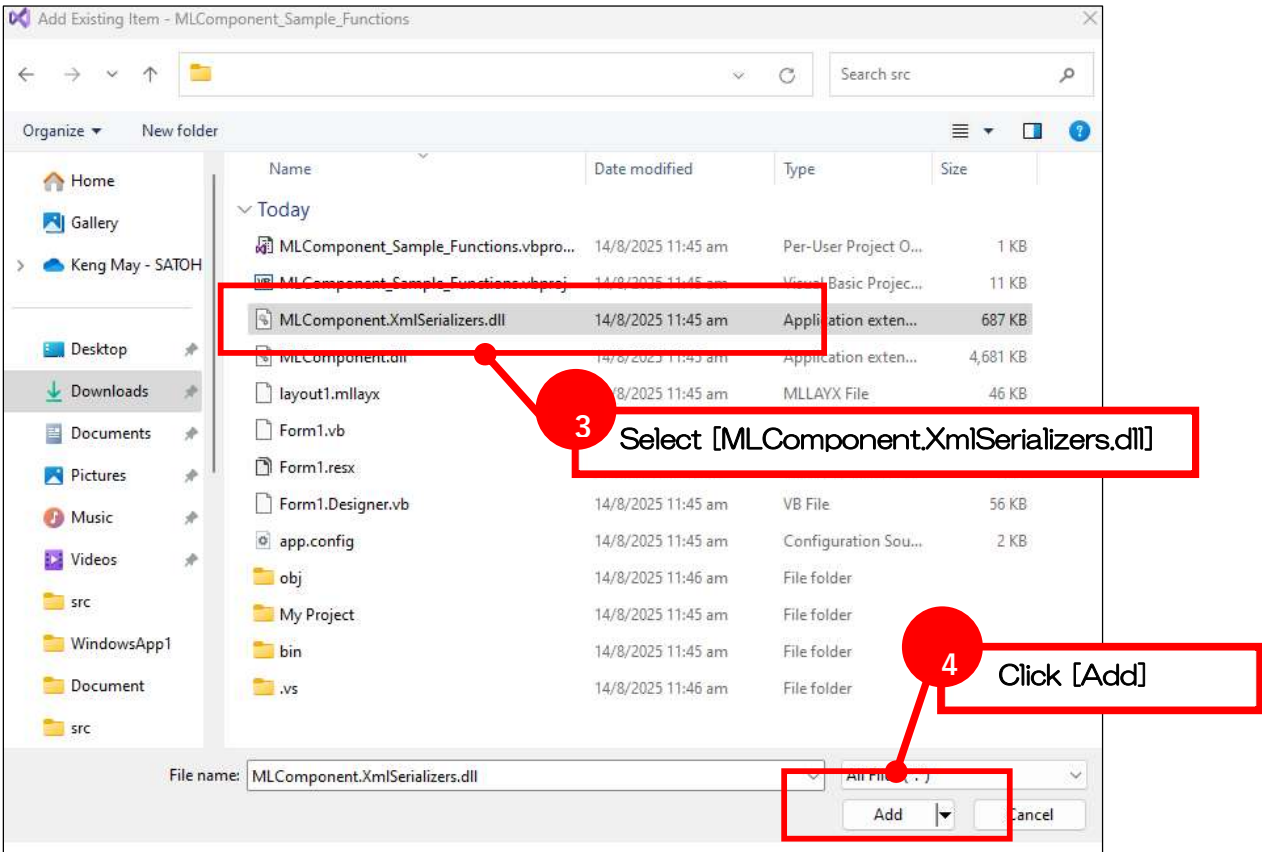
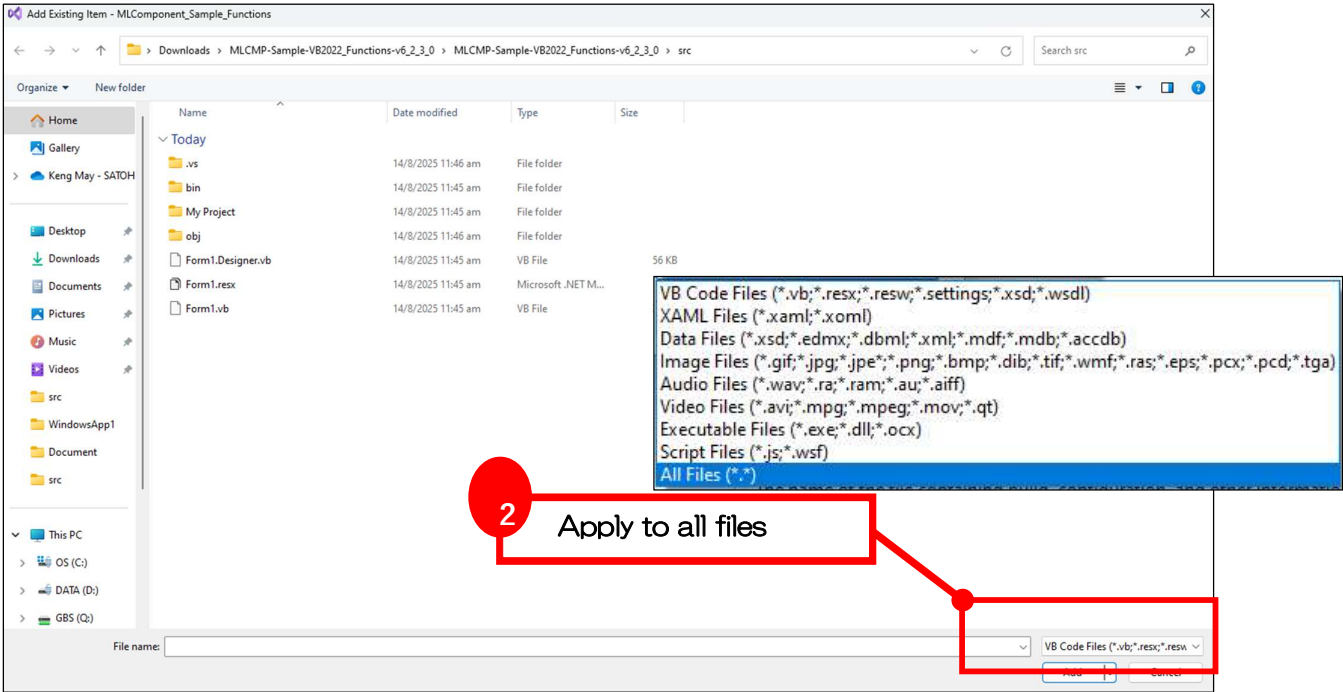
Method	Content
OpenPort	Setting, Protocol, Timeout
ClosePort	None
Output	LayoutFile, Darkness, Speed, Offset, MultiCut, SortMark,
Output Header	HeaderTailSetting, Formoverlay, TaxRate, PrnDataType, PrnData
OutputTail	(PrnDataArray)
SendStringData	None
SendRawData	None
GetStatus	Status string received from the printer
Cut	None
SendCancel	None

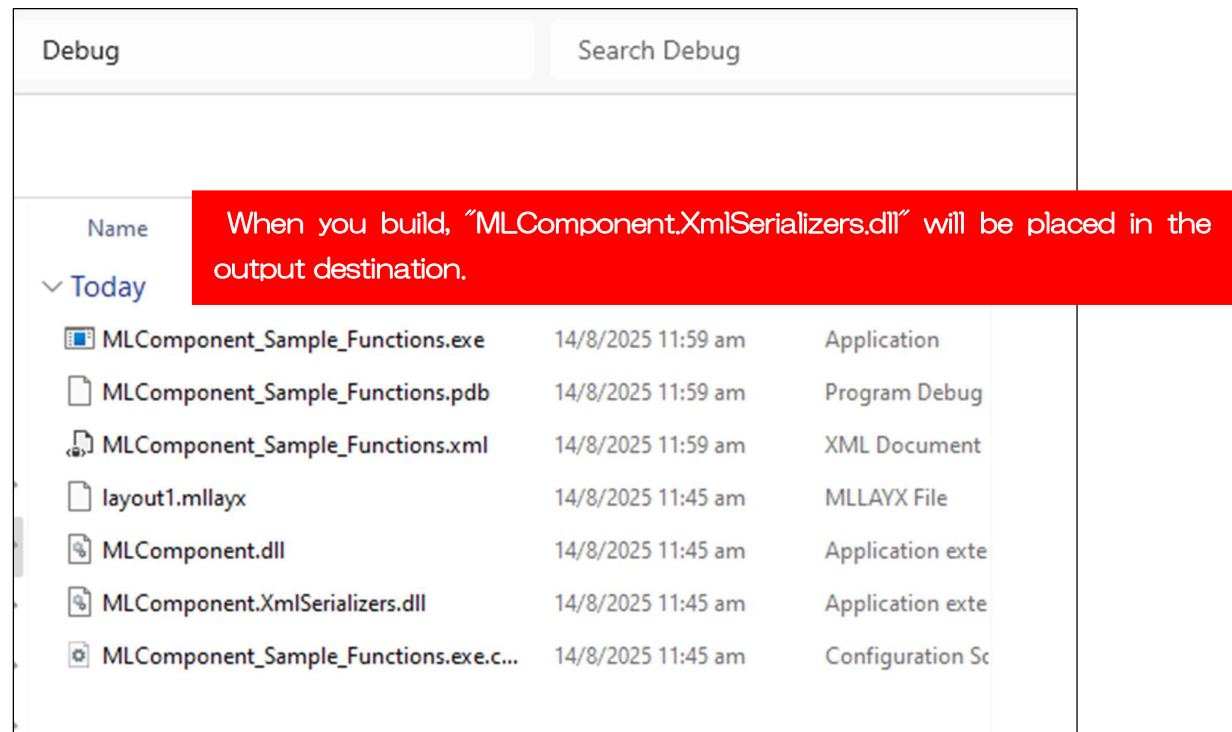
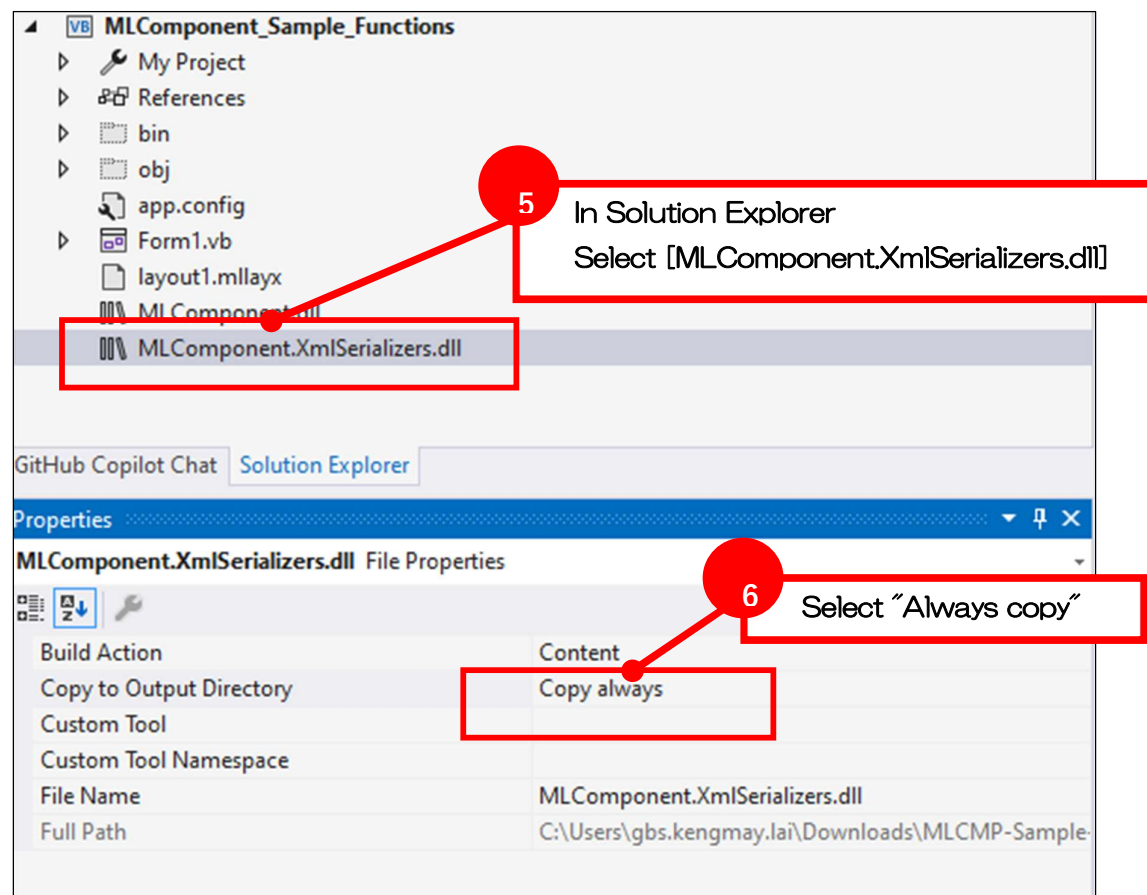
**2-
12****Improve the initial load speed of the layout**■ `MLComponent.XmlSerializers.dll`■ **Place `MLComponent.XmlSerializers.dll`**

When the .NET Framework libraries used by `MLComponent` are loaded, there may be a delay in processing when the layout information is loaded for the first time after application startup (e.g., in the `Output` method or `GetPrinter` method).

To improve library loading time, place the XML serializer "`MLComponent.XmlSerializers.dll`" in the same folder as `MLComponent.dll`. As an example, we will explain how to automatically place `MLComponent.XmlSerializers.dll` when building in Visual Studio.







2-13

Improve publishing speed

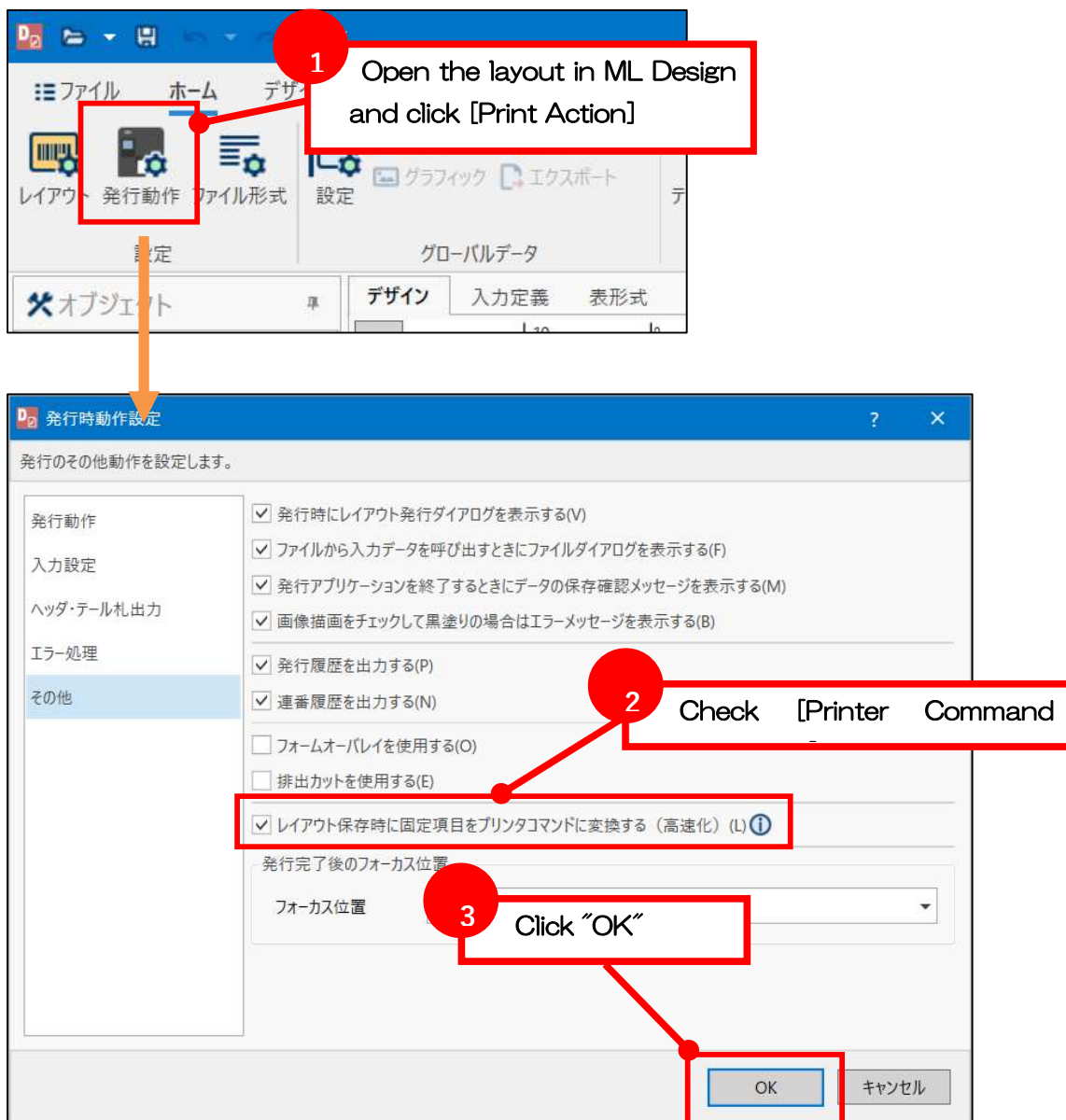
■Generating printer commands for fixed objects

■Pre-generate printer commands for fixed objects

Available in MLV5 Ver.5.7.5.0 and later, and MLV6 Ver.6.0.0.0 and later.

By converting fixed objects such as pasted text and grid lines—whose content does not change based on data—into printer commands during layout saving, you can improve label issuance speed.

This feature is particularly effective when there are a large number of fixed objects or when automatic line breaks are frequently used with pasted text in Windows fonts.



There are no settings on the MLComponent side.

If you are using a version older than Ver.5.7.5.0, please upgrade to Ver.5.7.5.0 or later.