CITIZEN

CITIZEN XML Print Service
JavaScript Label Print SDK
Programming Manual

For Ver. 1.03

CITIZEN SYSTEMS JAPAN CO., LTD.

Revision History

Date	Version	Description
12/26/2017	1.00	First issue.
9/18/2018	1.01	- Added a locale for Chinese model and Korean model to drawTextPtrFont. - Added the Chinese model and the Korean model to the definition of the locale.
6/26/2019		Added EConnectNotFound, EConnectOffline to the error code of JavaScript Label Print SDK response receive callback function.
7/18/2019		- Modified description of SendData tag and description of usage example. - Modified the setting contents of measurementUnit parameter of drawBitmap method.
6/2/2020		- Added CL-E300EX/303EX and CL-E321EX/331EX to support model.
7/2/2020	1.02	- Added CL-S700II/703II, CL-S621II/631II and CL-S521II/531II to support model. - Added URL description example for https communication. - Modified explanation and usage example of PrinterCheck method. - Added PrintHeadLowTemp, PrintHeadFailure, PrintHeadOverheat, MechanismOpen, AutoCutterError, FanMotorError, MiscError properties to the LabelPrinter class.
12/8/2020	1.03	- Modified explanation about HTTPS.
3/3/2021		- Modified the interface model numbers. - Modified explanation about HTTPS. - Added how to check the service version.
12/22/2023		- Added CL-S700III/703III to support model.
1/20/2026		- Added CL-S400DTII to support model.
7/23/2026		- Added CL-S403DTII to support model. (Page 8)

Permission Notice

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

Trademark

Microsoft, Windows are registered trademarks of Microsoft Corporation in the United States and/or other countries. (Official name for Windows is Microsoft Windows Operating System.)

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

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

Table of Contents

1. Introduction 7
1.1. System Overview 7
1.2. Scope of This Document 7
1.3. System Configuration Example 7
1.4. Supported Models 8
2. XML Print Service Messages 9
2.1. Request Messages 9
2.1.1. Transmission Method and Message Structure 9
2.1.2. POSPrinterRequest Tag 9
2.2. Response Messages 10
2.2.1. Message Structure 10
2.2.2. Acquiring the Request Result 10
2.2.3. Error Codes 11
2.3. Acquiring Device Status 12
3. Device Control Tags 14
3.1. Device Control Tag List 14
3.2. Details of Print Control Tags 15
3.2.1. Message ID (MessageID Tag) 15
3.2.2. Send command 16
4. XML Print Service Settings 17
4.1. Web Manager 17
4.1.1. Service Setting Screen / XML Print 17
4.1.2. Service Status Screen / XML Print 17
5. JavaScript Label Print SDK 18
5.1. Operating Environment 18
5.2. Programming Guide 18
5.2.1. Placement of SDK File 18
5.2.2. Program Configuration 18
5.2.3. Label design 19
5.2.4. Creating Device Control Object 19
5.2.5. Setting Response Receive Callback Function 19
5.2.6. Setting Send Error Callback Function 20
5.2.7. Executing Sending 20
5.3. Class summary 21
5.3.1. Device Control function(LabelPrint class) 21
5.3.2. Label Design function(LabelDesign class) 22
5.4. Return value 23
5.5. LabelPrint class 24
5.5.1. Constructor 24
5.5.2. messageID method 25
5.5.3. printerCheck method 26
5.5.4. print method 28
5.5.5. storeNVBitmap method 29
5.5.6. clearOutPut method 30
5.5.7. sendData method 31
5.5.8. HorizontalMagnification property 32
5.5.9. VerticalMagnification property 33
5.5.10. FormatAttribute property 34

5.5.11. ContinuousMediaLength property *35*
5.5.12. MeasurementUnit property *36*
5.5.13. PrintSpeed property *37*
5.5.14. FeedSpeed property *38*
5.5.15. SlewSpeed property *39*
5.5.16. BackupSpeed property *40*
5.5.17. PrintDarkness property *41*
5.5.18. DoubleHeat property *42*
5.5.19. VerticalOffset property *43*
5.5.20. HorizontalOffset property *44*
5.5.21. MediaHandling property *45*
5.5.22. StartOffset property *46*
5.5.23. StopOffset property *47*
5.5.24. LabelSensor property *48*
5.5.25. PrintMethod property *49*
5.5.26. SensorLocation property *50*
5.5.27. CommandInterpreterInAction property *51*
5.5.28. PaperError property *52*
5.5.29. RibbonEnd property *53*
5.5.30. BatchProcessing property *54*
5.5.31. Printing property *55*
5.5.32. Pause property *56*
5.5.33. WaitingForPeeling property *57*
5.5.34. PrintHeadLowTemp property *58*
5.5.35. PrintHeadFailure property *59*
5.5.36. PrintHeadOverheat property *60*
5.5.37. MechanismOpen property *61*
5.5.38. AutoCutterError property *62*
5.5.39. FanMotorError property *63*
5.5.40. MiscError property *64*
5.6. LabelDesign class 65
5.6.1. Constructor *65*
5.6.2. drawTextPtrFont method *66*
5.6.3. drawTextDLFont method *68*
5.6.4. drawNVBitmap method *70*
5.6.5. drawBitmap method *71*
5.6.6. drawBarCode method *73*
5.6.7. drawMaxiCode method *77*
5.6.8. drawPDF417 method *78*
5.6.9. drawDataMatrix method *79*
5.6.10. drawQRCode method *80*
5.6.11. drawAztec method *81*
5.6.12. drawGS1DataBar method *82*
5.6.13. drawLine method *84*
5.6.14. drawRect method *85*
5.6.15. fillRect method *86*
5.6.16. drawCircle method *87*
5.6.17. fillCircle method *88*
5.6.18. drawPolygon method *89*
5.6.19. fillPolygon method *90*
5.6.20. embedRawDesignCommand method *91*

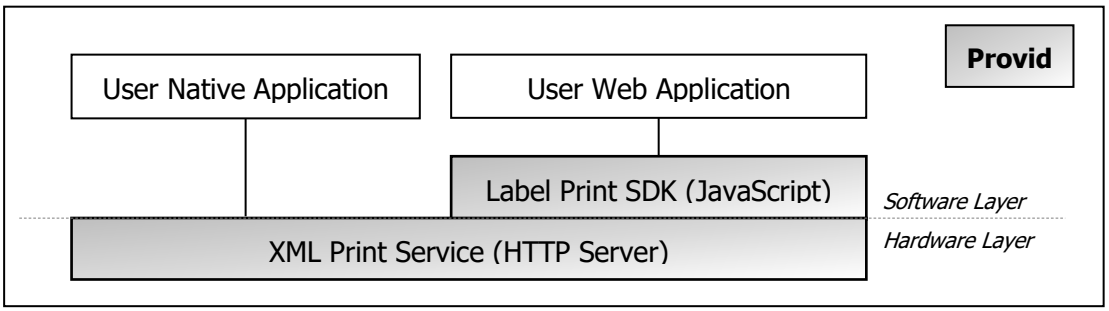
5.7. Appendix....92
5.7.1. Specifying object position....92
5.7.2. Predefined Constants....93

1. Introduction

This document is a manual for programmers for CITIZEN XML Print Service and JavaScript Label Print SDK.

1.1. System Overview

CITIZEN XML Print Service is provided to control a printer without a device driver in a multi-platform environment which is not operating system dependent.
Since the control method is HTTP (XML) based, a printer can be controlled easily from a Web service environment. Furthermore, CITIZEN XML-Label Print API is provided as a library for CITIZEN XML Print Service to print using JavaScript on the client side. The following shows a conceptual diagram of the provided service.

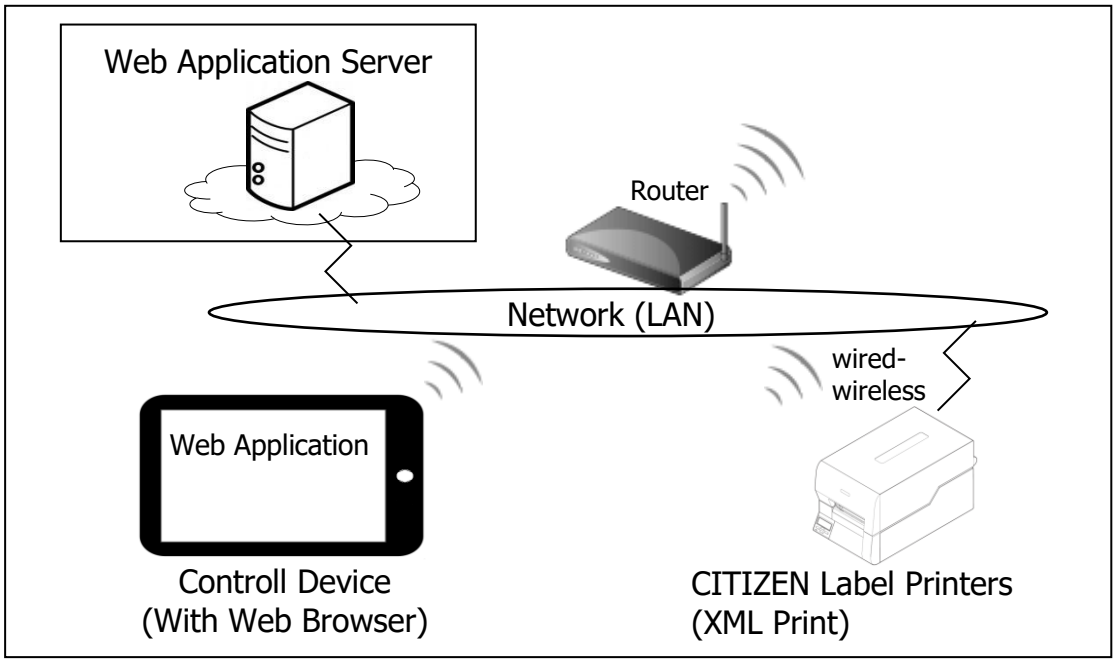


1.2. Scope of This Document

This document is intended to be a reference for developers of applications that use CITIZEN XML Print Service compatible printers.
For the control specifications for HTTP (XML) based printing, refer to "2. XML Print Service Messages", "3. Device Control Tags", and "4. XML Print Service Settings" in this document.
For the API specifications for printing from a Web service environment, refer to "5. JavaScript Label Print SDK".

1.3. System Configuration Example

The following shows a system configuration example for CITIZEN XML Print Service.



1.4. Supported Models

The applicable models of this service and the corresponding interfaces for those models are as follows. For details on the functions of each model, refer to the instruction manual of the corresponding printer.

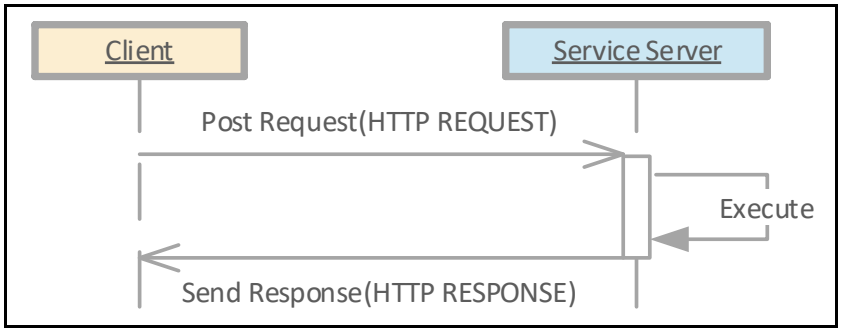
Applicable Model	Interface
CL-S400DT	IWired LAN (model number: IF1-EFX1*, IF1-EFX2) Wired LAN / Wireless LAN (model number: IF1-WFx4, IF1-WFx6)
CL-S400DTII/403DTII	
CL-S700III/703II	
CL-E720/730	
CL-E300EX/303EX	Wired LAN (model number: IF1-EFX1*, IF1-EFX2) Wired LAN / Wireless LAN (model number: IF1-WFx5, IF1-WFx6)
CL-E321EX/331EX	
CL-S700/703	Wired LAN (model number: IF5-EFX1)
CL-S700II/703II	
CL-S620/630	
CL-S620II/630II	
CL-S520/530	
CL-S520II/530II	

* Not available in HTTPS environment (only available in HTTP environment)

For the version number of this service, refer to "4.1.2. Service Status screen / XML Print". Usage and functions may differ depending on the version.

2. XML Print Service Messages

The service user issues a request message as an HTTP request and then receives a response message from the service as an HTTP response as shown in the figure below. Request messages and response messages are defined in the "CitizenXML-Print.xsd" XML schema file.



2.1. Request Messages

Control of the device (printer) is performed in accordance with the request messages issued from clients.

2.1.1. Transmission Method and Message Structure

Send a SOAP message using the following method for a request message.

- HTTP URL: http(s)://[IP address of this service]/xmlprint
  * The https is supported by IF5-EFX1 and other service versions 2.1 or later.
  * In the service version 2.0 and earlier other than IF5-EFX1, http://[IP address of this service]:8080
- HTTP method: POST
- HTTP header: Content-Type:text/xml; charset=UTF-8

The structure of a request message is as follows.

```
<?xml version="1.0" encoding="utf-8"?>
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
 <s:Body>

  <POSPrinterRequest xmlns=http://www.citizen.co.jp/POSPrinter/
   MajorVersion="1">
    <!-- RequestID -->
   <MessageID>3ea2110f-8d31-24ef-e2a5-ca98d2af5f8d</MessageID>
   <!— Send printer command -->
   <SendData>
    <Data>ASMNCg==</Data>
   </SendData>
  </POSPrinterRequest>

 </s:Body>
</s:Envelope>
```

<POSPrinterRequest> Tag

2.1.2. POSPrinterRequest Tag

Insert the device control tag within the <POSPrinterRequest> tag in the request message for controlling the device. For details on the device control tags, refer to "3. Device Control Tags" in this document.

2.2. Response Messages

The result of a request can be checked from the response message from this service.

2.2.1. Message Structure

The structure of a response message is as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
 <s:Body>

  <POSPrinterResponse xmlns=http://www.citizen.co.jp/POSPrinter/
   MajorVersion="1">
   <MessageID>0011e507451e-648e8cd7</MessageID>
   <Response ResponseCode="OK">
    <RequestID>26fc854c-e6f2-9a86-9060-6d76069cf4e2</RequestID>
   </Response>
  </POSPrinterResponse>

 </s:Body>
</s:Envelope>
```

<POSPrinterResponse> Tag

2.2.2. Acquiring the Request Result

The result of a request can be checked in the information within the <POSPrinterResponse> tag.

Item	Description
ResponseCode attribute	Stores the process result.
MessageID element	Stores the ID for identifying the response message.
RequestID element	Stores the message ID specified when the message was sent.
BusinessError element	Stores the error information when an error occurred.

Checking for Error Occurrence

Whether or not an error occurred can be checked by checking the value of the ResponseCode attribute of the Response element.

Code	Description
OK	Ended normally.
Rejected	Error occurred.

The following is an example of the Response element when the process ended normally.

```
<Response ResponseCode="OK">
  <RequestID>26fc854c-e6f2-9a86-9060-6d76069cf4e2</RequestID>
</Response>
```

Checking Error Information

The cause of a result can be checked from the contents of the Code and Description elements in the BusinessError element within the Response element when an error occurs. For details, refer to "2.2.3. Error Codes" in this document.

The following is an example of the Response element when an error occurred.

```
<Response ResponseCode="Rejected">
  <RequestID></RequestID>
  <BusinessError Severity="Error">
    <Code>ETimeout</Code>
   <Description>The device is not respond.</Description>
  </BusinessError>
</Response>
```

2.2.3. Error Codes

The error code, error description, and other detailed information are set in the BusinessError element in the Response element within a response message. The error codes used with this service are shown below.

Code	Description
RequestInvalid	Request information is invalid
EConnectNotFound	Unsupported model
EOffline	Device is offline
EIllegal	Unsupported or invalid parameter
ETimeout	Processing timeout

2.3. Acquiring Device Status

To acquire the device status, use the GetDeviceInfo tag to specify a device information acquisition request.

Parameters

Attribute	Description	Setting range
BarcodePrinterStatus	BC printer information flag	Add BC printer information when "true" is specified.
BarcodePrinterErrorInfo	BC printer error information flag	Add BC printer error information when "true" is specified.

An example of a request message is shown below.

```
<?xml version="1.0" encoding="utf-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
 <s:Body>
  <POSPrinterRequest xmlns=http://www.citizen.co.jp/POSPrinter/
   MajorVersion="1">
   <!-- Device information acquisition request -->
   <GetDeviceInfo BarcodePrinterStatus='true'
                  BarcodePrinterErrorInfo='true' />
  </POSPrinterRequest>
 </s:Body>
</s:Envelope>
```

An example of a request message is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
 <s:Body>
  <POSPrinterResponse xmlns=http://www.citizen.co.jp/POSPrinter/
   MajorVersion="1">
   <MessageID>0011e507451e-648e8cd8</MessageID>
   <Response ResponseCode="OK">
    <RequestID></RequestID>
   </Response>
   <GetDeviceInfo>
    <Device BarcodePrinterStatus="NNNNNYNN"
            BarcodePrinterErrorInfo="60404040" />
   </GetDeviceInfo>
  </POSPrinterResponse>
 </s:Body>
</s:Envelope>
```

<GetDeviceInfo> tag

For the contents of the device information, the status is stored in the BarcodePrinterStatus attribute and BarcodePrinterErrorInfo attribute of the Device element of the response message.
*In the above example of the response message, it indicates that the printer is pausing.

BarcodePrinterStatus attribute

The printer status is stored as Yes / No.

Index	Description
[0]	Shows the printer status if the interpreter is processing commands.
[1]	Shows the printer status if there is a paper error.
[2]	Shows the printer status if there is a ribbon end error.
[3]	Shows the status if the printer is processing batch print jobs.
[4]	Shows the status if the printer is currently printing.
[5]	Shows the status if the printer is in the pause mode.
[6]	Shows the status if the printer is in the wait for peeling status.
[7]	Reserved. *Always "N"

BarcodePrinterErrorInfo attribute

The internal status of the printer is stored as 4-byte data. (Supported by IF5-EFX1 of wired LAN I/F)

Byte	Bit	Description	Error status
1	0	Battery exhaustion (Unsupported)	Yes:1 No:0
	1	Head at low temperature (Unsupported)	
	2	Main PCB at low temperature (Unsupported)	
	3	Wear and tear on a head	
	4	Spare	Always 0
	5	Pause	Yes:1 No:0
	6	Fixed	Always 1
	7		Always 0
2	0	Spare	Always 0
	1	Head overheat	Yes:1 No:0
	2	Spare	Always 0
	3		
	4	Mechanism is exposed	Yes:1 No:0
	5	Paper end	
	6	Fixed	Always 1
	7		Always 0
3	0	Paper out	Yes:1 No:0
	1	Ribbon end	
	2	Overheating of Main PCB (Unsupported)	
	3	Spare	Always 0
	4	Abnormality in option boards (Unsupported)	Yes:1 No:0
	5	Abnormality in auto cutter	
	6	Fixed	Always 1
	7		Always 0
4	0	Fan motor stop (Unsupported)	Yes:1 No:0
	1	Spare	Always 0
	2		
	3		
	4		
	5	Error is occurring	Yes:1 No:0
	6	Fixed	Always 1
	7		Always 0

3. Device Control Tags

3.1. Device Control Tag List

The device control tags that can be used with this service are shown below.

Function	Tag	Description
Message ID	<MessageID>	To specify for the sender to identify the message.
Command send	<SendData>	To specify printer commands to send.
Device information	<GetDeviceInfo>	Refer to "2.3. Acquiring Device Status".

3.2. Details of Print Control Tags

3.2.1. Message ID (MessageID Tag)

Value

Specify the request message ID.

Description

This tag is for the sender to identify the message.
The specified request message ID is added to the <RequestID> tag of the response message. For details on response messages, refer to "2.2. Response Messages" in this document.

Usage example

```
<MessageID>12345678</MessageID>
```

3.2.2. Send command

Parameters

Element	Description	Acceptable values
Data	Send data	Base64 encoded data.

Description

This tag is to send printer command to the printer.
Printer command needs to base64 encode bainary data.

Usage example

The following is a description example of SendData tag when sending a printer command.

Printer command

[02][1B]G0	Set the command set to "DMI/DMW"
[02]L	Starts the label format mode
D11	Sets the pixel size
1X1100000100000b0100010000050005	Prints the rectangle
Q0001	Sets the number of print
E	Ends the label format mode and prints

```
<?xml version="1.0" encoding="utf-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <POSPrinterRequest xmlns="http://www.citizen.co.jp/POSPrinter/"
     MajorVersion="1">
      <MessageID>00000001</MessageID>
      <SendData>
        <Data>AhtHMA0KAkwNCkQxMQ0KMVgxMTAwMDAwMTAwMDAwYjAxMDAwMTAwMDAw
         NTAwMDUNClEwMDAxDQpFDQo=</Data>
      </SendData>
    </POSPrinterRequest>
  </s:Body>
</s:Envelope>
```

4. XML Print Service Settings

This chapter describes how to set CITIZEN XML Print Service.

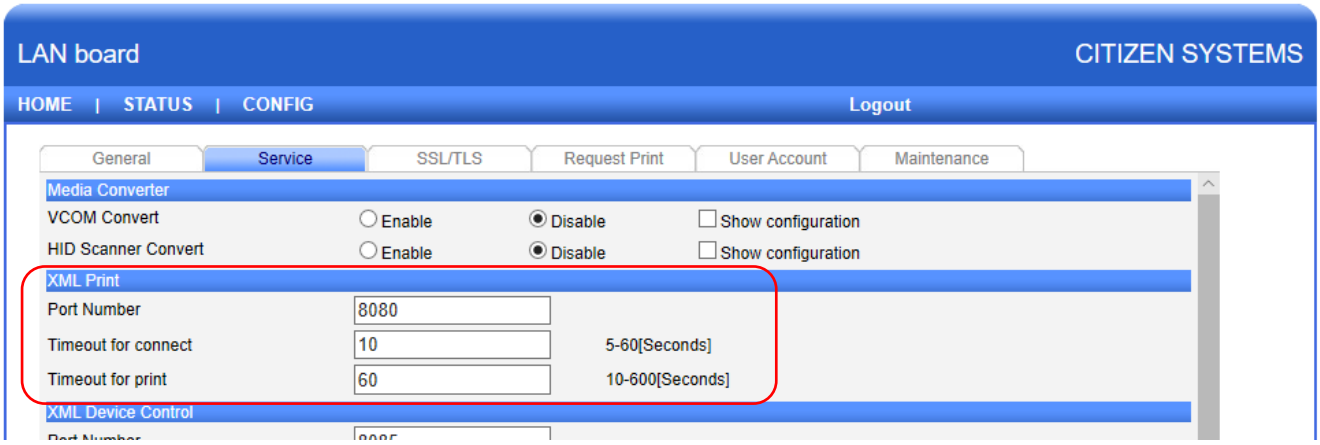
4.1. Web Manager

The settings for the printers can be changed by connecting from a Web browser to each printer. For details on the basic operations, refer to the interface board instruction manual of the printer.

This document describes the setting items of XML Print Service.

4.1.1. Service Setting Screen / XML Print

The settings of the service provided by the printer can be set in the Service screen.

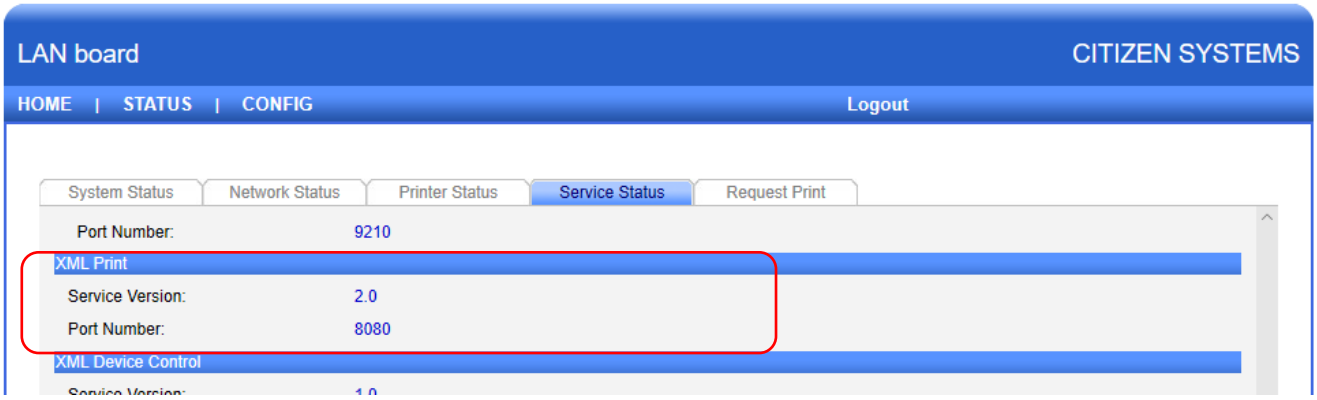


Item	Initial value	Acceptable range	Description
Port Number	8080	1025 to 65535	Port number to connect
Timeout for connect	10	5 to 60	Timeout value for starting printing
Timeout for print	60	10 to 600	(unused)

*This page won't be displayed for non-supported printer models.
*Each set value will be persistent through a power cycle, will be reset to the default value by the factory reset.

4.1.2. Service Status Screen / XML Print

The service version and setting information can be check in the Service Status screen.



5. JavaScript Label Print SDK

Label Print SDK is a JavaScript library to control the CITIZEN XML Print service from client side. This allows you to print labels from web application by using JavaScript.

5.1. Operating Environment

The web browser must support HTML5.

5.2. Programming Guide

5.2.1. Placement of SDK File

Label Print SDK is provided as JavaScript. To use the SDK, place "cxmlp-label-api.js" to the web server. Do not modify the source code of the SDK. It may not work properly otherwise.

5.2.2. Program Configuration

To control a device, insert your program in the HTML <script> tag placed on the web server.
The program structure looks as below.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Label Print SDK Sample</title>
  <script type="text/javascript" src=" cxmlp-label-api.js"></script>
  <script type="text/javascript">
    //Create an LabelDesign object
    var design = new citizen.LabelDesign();
    //Register design of label
    design.drawCircle(50, 50, 15);
    design.drawRect(20, 30, 180, 280, 10);
    //Create an Device Control object
    var print = new citizen.LabelPrint();
    //Set response receive callback function
    print.OnReceive = function (res, xml) {
      alert(res.ResponseCode);
    };
    //Set send error callback function
    print.OnError = function (res) {
      alert(res.status);
    };
    //Set RequestMessageID
    print.MessageID('12345678');
    //Execute sending
    print.print('http://192.168.129.178/xmlprint', design, 1);
  </script>
</head>

<body>
      .
      .
</body>
</html>
```

Integrating SDK

Program text

5.2.3. Label design

Label designing is done by the LabelDesign object. A label design is defined by creating a "citizen.LabelDesign" instance and using design methods. For details on the design methods, refer to the "5.6. LabelDesign class".

An example of label design is shown bellow.

```
//Create an LabelDesign object
design = new citizen.LabelDesign();
//Register design of label
design.drawCircle(50, 50, 15);
design.drawRect(20, 30, 180, 280, 10);
```

5.2.4. Creating Device Control Object

Device control is done by the LabelPrint object. Create a "citizen.LabelPrint" instance first.

5.2.5. Setting Response Receive Callback Function

The control result can be checked in the argument information of the function by setting a callback function in the OnReceive properties of Device Control Object.

Item	Description
ResponseCode	Stores the process result.
MessageID	Stores the ID for identifying the response message.
RequestID	Stores the message ID specified when the message was sent.
ErrorCode	Stores the error code when an error occurred.
Description	Stores the explanation when an error occurred.

An example of setting the response receive callback function is shown below.

```
//Set response receive callback function
print.OnReceive = function (res, xml) {
  var msg;
  if(res.ResponseCode == 'OK'){
    msg = 'Print success!\n\n';
  }
  else{
    msg = 'Print failure!\n\n';
    msg += ' Code: ' + res.ErrorCode + '\n';
    msg += ' Description: ' + res.Description + '\n\n';
  }
  msg += ' RequestID: ' + res. RequestID + '\n';
  alert(msg);
};
```

Checking for Error Occurrence

Errors can be confirmed by checking the ResponseCode value.

Code	Description
OK	Done successfully.
Rejected	Error occurred.

Checking Error Information

The cause of a result can be checked from the contents of the ErrorCode and Description elements stored when an error occurs.
The error codes used with this service are shown below.

Code	Description
EConnectNotFound	Unsupported model
EOffline	Device is offline
ETimeout	Processing timeout

5.2.6. Setting Send Error Callback Function

The error details can be checked in the argument information of the function by setting a callback function in the OnError properties of Device Control Object.
The status when an error occurs is stored in status, and the response details are stored in responseText.

An example of setting the send error callback function is shown below.

```
//Set send error callback function
print.OnError = function (res) {
  var msg = 'Send failure!\n\n';
  msg += ' status: ' + res. status + '\n';
  msg += ' responseText: ' + res. responseText + '\n';
  alert(msg);
};
```

5.2.7. Executing Sending

The sending process is started by calling the method of the LabelPrint object by specifying the URL of the XML Print Service in the Device Control Object. When the process ends, the set response receive callback function is called and the control result can be acquired. For details on acquiring control results, refer to "5.2.5. Setting Response Receive Callback Function" in this document.

The format of the specified URL is shown below.
http(s)://[IP address of this service]/xmlprint
* The https is supported by IF5-EFX1 and other service versions 2.1 or later.
* In the service version 2.0 and earlier other than IF5-EFX1, http://[IP address of this service]:8080

An example of specifying the executing of sending is shown below.

```
//Execute sending (for https)
print.print("https://192.168.129.178/xmlprint/", design, 1);

//Execute sending (for http)
print.print("http://192.168.129.178/xmlprint/", design, 1);

//Execute sending (for http of the service version 2.0 and earlier)
print.print("http://192.168.129.178:8080/", design, 1);
```

5.3. Class summary

5.3.1. Device Control function(LabelPrint class)

Methods (LabelPrint class)

No	Method	Description
1	LabelPrint	Constructor.
2	messageID	Set the message ID.
3	printerCheck	Gets status of a printer.
4	print	Prints labels.
5	storeNVBitmap	Stores a bitmap image into the flash memory.
6	clearOutput	Clears all data in the printer buffer.
7	sendData	Sends binary data to a printer.

Properties (LabelPrint class)

No	Property	Attribute	Description
1	HorizontalMagnification	R/W	Horizontal magnification of the label.
2	VerticalMagnification	R/W	Vertical magnification of the label.
3	FormatAttribute	R/W	Specifies how to print overlapped objects.
4	ContinuousMediaLength	R/W	Label length for continuous paper.
5	MeasurementUnit	R/W	Measurement unit. Inches or millimeters.
6	PrintSpeed	R/W	Print speed.
7	FeedSpeed	R/W	Feed speed.
8	SlewSpeed	R/W	Slew speed.
9	BackupSpeed	R/W	Backup speed.
10	PrintDarkness	R/W	Print darkness.
11	DoubleHeat	R/W	Double heat.
12	VerticalOffset	R/W	Vertical offset of the printing position.
13	HorizontalOffset	R/W	Horizontal offset of the printing position.
14	MediaHandling	R/W	Media handling after print. (Pause, Cut or Peel)
15	StartOffset	R/W	Vertical offset of the paper position when start printing.
16	StopOffset	R/W	Vertical offset of the paper position when stop printing.
17	LabelSensor	R/W	Label sensor type. (See through, Reflect or None)
18	PrintMethod	R/W	Thermal transfer or Direct thermal.
19	SensorLocation	R/W	Front or Rear. (for the models which have multiple sensors)
20	CommandInterpreterInAction	R	Status: CommandInterpreterInAction
21	PaperError	R	Status: PaperError
22	RibbonEnd	R	Status: RibbonEnd
23	BatchProcessing	R	Status: BatchProcessing
24	Printing	R	Status: Printing
25	Pause	R	Status: Pause
26	WaitingForPeeling	R	Status: WaitingForPeeling
27	PrintHeadLowTemp	R	Status: PrintHeadLowTemp
28	PrintHeadFailure	R	Status: PrintHeadFailure
29	PrintHeadOverheat	R	Status: PrintHeadOverheat
30	MechanismOpen	R	Status: MechanismOpen
31	AutoCutterError	R	Status: AutoCutterError
32	FanMotorError	R	Status: FanMotorError
33	MiscError	R	Status: MiscError

5.3.2. Label Design function(LabelDesign class)

Methods (LabelDesign class)

No	Method	Description
1	LabelDesign	Constructor.
2	drawTextPtrFont	Draws text by using a printer device font.
3	drawTextDLFont	Draws text by using a downloaded font.
4	drawNVBitmap	Draws a bitmap image stored in the flash memory of the printer.
5	drawBitmap	Draws a bitmap file in the computer.
6	drawBarCode	Draws a barcode.
7	drawMaxiCode	
8	drawPDF417	
9	drawDataMatrix	
10	drawQRCode	
11	drawAztec	
12	drawGS1DataBar	
13	drawLine	Draws a line.
14	drawRect	Draws a rectangular.
15	fillRect	Draws a rectangular filled with a specified pattern.
16	drawCircle	Draws a circle.
17	fillCircle	Draws a circle filled with a specified pattern.
18	drawPolygon	Draws a polygon.
19	fillPolygon	Draws a polygon filled with a specified pattern.
20	embedRawDesignCommand	Embeds raw printer commands.

5.4. Return value

Return value	Description
CLS_SUCCESS (0)	The operation has been done successfully.
CLS_E_ILLEGAL (1101)	The operation is not supported by the printer or an invalid parameter was used.

5.5. LabelPrint class

5.5.1. Constructor

Syntax
citizen.LabelPrint ()

Parameters
none

Description
Creates an instance of the LabelPrint class.

Return value
none

Example
```
var print = new citizen.LabelPrint();
```

5.5.2. messageID method

Syntax

messageID (messageID)

Parameters

Parameter	[IN/OUT]	Description	Setting range
messageID	[IN]	Request message ID	

Description

This method is used to enable the sender to identify the message.
The specified request message ID is added to the RequestID parameter of the control result. For details on control results, refer to "5.2.5. Setting Response Receive Callback Function" in this document.

Return value

none

Example

```
print.messageID("00000001");
```

5.5.3. printerCheck method

Syntax

printerCheck (address)

Parameters

Parameter	[IN/OUT]	Description	Setting range
address	[IN]	HTTP URL	

Description

Gets the status of the printer and stores information into the properties, CommandInterpreterInAction, PaperError, RibbonEnd, BatchProcessing, Printing, Pause and WaitingForPeeling as well as PrintHeadLowTemp, PrintHeadFailure, PrintHeadOverheat, MechanismOpen, AutoCutterError, FanMotorError and MiscError. (PrintHeadLowTemp, PrintHeadFailure, PrintHeadOverheat, MechanismOpen, AutoCutterError, FanMotorError and MiscError properties ware supported by IF5-EFX1 and other service versions 2.1 or later.)
When this method returns an error, a communication failure or a device error might occur. In this case, make sure that the contents of the specified address are correct, or that the printer is connected to the network.

For printers of CL-S5xx/6xx/70x, change the setting of "Parallel Error Output" to OFF beforehand. This can be done in the "Advanced" tab of the Label Printer Utility.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
//Below, the processing in the receive callback function
function myOnReceive(res, xml)
{
    if (res.ResponseCode == 'OK')
    {
        // CommandInterpreterInAction Property
        if (print.getCommandInterpreterInAction() == 1) {
            // Command interpreter in action
        }
        // PaperError Property
        if (print.getPaperError() == 1) {
            // Paper error
        }
        // RibbonEnd Property
        if (print.getRibbonEnd() == 1) {
            // Ribbon end
        }
        // BatchProcessing Property
        if (print.getBatchProcessing == 1) {
            // Batch processing
        }
        // Printing Property
        if (print.getPrinting()== 1) {
            // Printing
        }
        // Pause Property
        if (print.getPause() == 1) {
            // Pause
        }
```

```
        // WaitingForPeeling Property
        if (print.getWaitingForPeeling() == 1) {
            // Waitiong for peeling
        }

        // PrintHeadLowTemp Property
        if (print.getPrintHeadLowTemp() == 1) {
            // Print head low temp
        }
        // PrintHeadFailure Property
        if (print.getPrintHeadFailure() == 1) {
            // Print head failure
        }
        // PrintHeadOverheat Property
        if (print.getPrintHeadOverheat() == 1) {
            // Print head overheat
        }
        // MechanismOpen Property
        if (print.getMechanismOpen() == 1) {
            // Mechanism open
        }
        // AutoCutterError Property
        if (print.getAutoCutterError() == 1) {
            // Auto cutter error
        }
        // FanMotorError Property
        if (print.getFanMotorError() == 1) {
            // Fan motor error
        }
        // MiscError Property
        if (print.getMiscError() == 1) {
            // Misc error
        }
    }
    else
    {
        // Fail
    }
}


//Set response receive callback function
print.OnReceive = myOnReceive;


//Send Status Acquisition Command
print.messageID("00000001");
print.printerCheck("http://192.168.129.178/xmlprint");

//When a response to the status acquisition command is received from
//the printer, the reception callback function myOnReceive () is called
```

5.5.4. print method

Syntax

print (address, design, quantity)

Parameters

Parameter	[IN/OUT]	Description	Setting range
address	[IN]	HTTP URL	
design	[IN]	Instance of LabelDesignclass	
quantity	[IN]	Number of labels to print	1 - 9999

Description

Prints labels by sending a label design created in the LabelDesign class, followed by printer configuration commands if any of properties are set.

Return value

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Example

```
design.fillCircle(50, 50, 50, citizen.LabelConst.CLS_SHADED_PTN_11);
print.messageID("00000001");
print.print("http://192.168.129.178/xmlprint", design, 1);
```

5.5.5. storeNVBitmap method

Syntax

storeNVBitmap (image, address, name, rotation, width, height)

Parameters

Parameter	[IN/OUT]	Description	Setting range
image	[IN]	Image object	
address	[IN]	HTTP URL	
name	[IN]	File name to store	ASCII characters except below: -Underscore cannot be the first character. -The following symbols. □/ : * ? " < > |
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotate CLS_RT_RIGHT90: Rotate right 90 CLS_RT_ROTATE180: Rotate 180 CLS_RT_LEFT90: Rotate left 90
width	[IN]	Horizontal size (pixels)	
height	[IN]	Vertical size (pixels)	

Description

Stores a bitmap image into the flash memory with specified settings such as file name, rotation, width and height. The stored bitmap image can be used by the drawNVBitmap method.
Supported graphic file formats are BMP/GIF/EXIF/JPG/PNG/TIFF.

The image is resized by the specified width and height with keeping the original aspect ratio.
Example: The image size will be 200 x 50 pixels under the following conditions.
The original image size: 400 x 100 pixels
Width: 200
Height: 200

If 0 is set to either width or height, the image size will be calculated by another parameter.
Example: The image size will be 800 x 200" pixels under the following conditions.
The original image size: 400 x 100 pixels
Width: 0
Height: 200

Return value

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Example

```
var image = new Image();

//Omit the image reading process

print.messageID("00000001");
print.storeNVBitmap(image, "http://192.168.129.178/xmlprint",
          "Dice_1.bmp", citizen.LabelConst.CLS_RT_NORMAL, 0, 0);
```

5.5.6. clearOutPut method

Syntax

clearOutput ()

Parameters

Parameter	[IN/OUT]	Description	Setting range
address	[IN]	HTTP URL	

Description

Clears data in the print buffer. This triggers the initialization process which is equivalent to the boot sequence.

Return value

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Example

```
print.messageID("00000001");
print.clearOutput();
```

5.5.7. sendData method

Syntax

sendData (address, data)

Parameters

Parameter	[IN/OUT]	Description	Setting range
address	[IN]	HTTP URL	
data	[IN]	Send data	

Description

Sends binary data to a printer.
This can be used only when sending raw printer commands to the printer.

Return value

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Example

```
var data = new Array(0x01, 0x23, 0x0D, 0x0A); // [01]# (Reset)

print.messageID("00000001");
print.sendData("http://192.168.129.178/xmlprint", data);
```

5.5.8. HorizontalMagnification property

Syntax
HorizontalMagnification

Attribute
Read/Write

Description
Horizontal dot size. "1" or "2".

Setter method
setHorizontalMagnification (horizontalMagnification)

"1" or "2" must be set as a parameter.
Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getHorizontalMagnification ()

Example

```
var pixelsize;

pixelsize = print.getHorizontalMagnification();
print.setHorizontalMagnification(2);
```

5.5.9. VerticalMagnification property

Syntax
VerticalMagnification

Attribute
Read/Write

Description
Vertical dot size. "1", "2" or "3".

Setter method
setVerticalMagnification (verticalMagnification)

"1", "2" or "3" must be set as a parameter.
Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getVerticalMagnification ()

Example

```
var pixelsize;

pixelsize = print.getVerticalMagnification();
print.setVerticalMagnification(3);
```

5.5.10. FormatAttribute property

Syntax
formatAttribute

Attribute
Read/Write

Description
Specifies how to print the area where multiple objects are overlapped.

0: XOR mode. Overlapped area will not be printed.
1: OR mode. Overlapped area will be printed.

Setter method
setFormatAttribute (formatAttribute)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getFormatAttribute ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var attr;

print.setFormatAttribute(1);
attr = print.getFormatAttribute();
```

Printing Result

XOR 

OR 

5.5.11. ContinuousMediaLength property

Syntax
ContinuousMediaLength

Attribute
Read/Write

Description
Sets a paper length for continuous paper.

Inch system	0001 – 9999 (0.01 - 99.99 inches)
Metric system	0001 – 9999 (0.1 - 999.9 mm)

Setter method
setContinuousMediaLength (continuousMediaLength)

The default value in the printer is "0000."
Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getContinuousMediaLength ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var medialength;

print.setContinuousMediaLength(2000);
medialength = print.getContinuousMediaLength();
```

5.5.12. MeasurementUnit property

Syntax
MeasurementUnit

Attribute
Read/Write

Description
Sets the measurement unit.

Value	Description
CLS_UNIT_MILLI (0)	Metric system
CLS_UNIT_INCH (1)	Inch system

Setter method
setMeasurementUnit (measurementUnit)

The default value in the printer is "1."
Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getMeasurementUnit ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var unit;

print.setMeasurementUnit(citizen.LabelConst.CLS_UNIT_MILLI);
unit = print.getMeasurementUnit();
```

5.5.13. PrintSpeed property

Syntax
PrintSpeed

Attribute
Read/Write

Description
Sets the print speed.
See "5.7.2. Predefined Constants", No.18 for available value.
*Initial and maximum value vary depending on the printer model.

Setter method
setPrintSpeed (printSpeed)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getPrintSpeed ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var printspeed;

print.setPrintSpeed(citizen.LabelConst.CLS_SPEEDSETTING_X);
printspeed = print.getPrintSpeed();
```

5.5.14. FeedSpeed property

Syntax
FeedSpeed

Attribute
Read/Write

Description
Sets the feed speed.
See "5.7.2. Predefined Constants", No.18 for available value.
*Initial and maximum value vary depending on the printer model.

Setter method
setFeedSpeed (feedSpeed)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getFeedSpeed ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example
```
var feedspeed;

print.setFeedSpeed(citizen.LabelConst.CLS_SPEEDSETTING_W);
feedspeed = print.getFeedSpeed();
```

5.5.15. SlewSpeed property

Syntax
SlewSpeed

Attribute
Read/Write

Description
Sets the slew speed.
See "5.7.2. Predefined Constants", No.18 for available value.
*Initial and maximum value vary depending on the printer model.

Setter method
setSlewSpeed (slewSpeed)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getSlewSpeed ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example
```
var slewspeed;

print.setSlewSpeed(citizen.LabelConst.CLS_SPEEDSETTING_V);
slewspeed = print.getSlewSpeed();
```

5.5.16. BackupSpeed property

Syntax
BackupSpeed

Attribute
Read/Write

Description
Sets the backup speed. See below for the acceptable values.

Value	Description
CLS_SPEEDSETTING_A or CLS_SPEEDSETTING_B	1.0 inch(25.4mm) / sec
CLS_SPEEDSETTING_C or CLS_SPEEDSETTING_D	2.0 inch(50.8mm) / sec
CLS_SPEEDSETTING_E or CLS_SPEEDSETTING_F	3.0 inch(76.2mm) / sec
CLS_SPEEDSETTING_G or CLS_SPEEDSETTING_H	4.0 inch(101.6mm) / sec
CLS_SPEEDSETTING_I or CLS_SPEEDSETTING_J	5.0 inch(127.0mm) / sec
CLS_SPEEDSETTING_K or CLS_SPEEDSETTING_L	6.0 inch(152.4mm) / sec
CLS_SPEEDSETTING_M or CLS_SPEEDSETTING_N	7.0 inch(177.8mm) / sec
CLS_SPEEDSETTING_O	8.0 inch(203.2mm) / sec
CLS_SPEEDSETTING_1 - CLS_SPEEDSETTING_8	1.0 inch(25.4mm) / sec - 8.0 inch(203.2mm) / sec

*Initial and maximum value vary depending on the printer model.

Setter method
setBackupSpeed (backupSpeed)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getBackupSpeed ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var backupspeed;

print.setBackupSpeed(citizen.LabelConst.CLS_SPEEDSETTING_O);
backupspeed = print.getBackupSpeed();
```

5.5.17. PrintDarkness property

Syntax
PrintDarkness

Attribute
Read/Write

Description
Sets the print density. The acceptable values are below:

Setting range	0 - 30
Default value	10

Setter method
setPrintDarkness (printDarkness)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getPrintDarkness ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var printdarkness;

print.setPrintDarkness(30);
printdarkness = print.getPrintDarkness();
```

5.5.18. DoubleHeat property

Syntax
DoubleHeat

Attribute
Read/Write

Description
Sets the double heat option. The acceptable values are below:

0: OFF (Default)
1: ON

Setter method
setDoubleHeat (doubleHeat)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getDoubleHeat ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var doubleheat;

print.setDoubleHeat(1);
doubleheat = print.getDoubleHeat();
```

5.5.19. VerticalOffset property

Syntax
VerticalOffset

Attribute
Read/Write

Description
Sets the vertical offset to adjust the vertical printing position on the paper.

Inch system	0000 – 9999 (0.00 inch - 99.99 inches)
Metric system	0000 – 9999 (0.0 mm - 999.9 mm)
Default value	0000

Setter method
setVerticalOffset (verticalOffset)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getVerticalOffset ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var verticaloffset;

print.setVerticalOffset(10);
verticaloffset = print.getVerticalOffset();
```

5.5.20. HorizontalOffset property

Syntax
HorizontalOffset

Attribute
Read/Write

Description
Sets the horizontal offset to adjust the horizontal printing position on the paper.

Inch system	0000 – 9999 (0.00 inch - 99.99 inches)
Metric system	0000 – 9999 (0.0 mm - 999.9 mm)
Default value	0000

Setter method
setHorizontalOffset (horizontalOffset)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getHorizontalOffset ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var horizontaloffset;

print.setHorizontalOffset(20);
horizontaloffset = print.getHorizontalOffset();
```

5.5.21. MediaHandling property

Syntax
MediaHandling

Attribute
Read/Write

Description
Sets the way how to handle media after printing.

Value	Description
CLS_MEDIAHANDLING_NONE (0)	To print without any special actions.
CLS_MEDIAHANDLING_TEAROFF (1)	To feed labels to the tear bar.
CLS_MEDIAHANDLING_DISPENSES (2)	To feed labels to the tear bar, and wait for removal.
CLS_MEDIAHANDLING_PAUSE (3)	To pause after printing.
CLS_MEDIAHANDLING_CUT (4)	To cut after printing.
CLS_MEDIAHANDLING_CUTANDPAUSE (5)	To cut and pause after printing.
CLS_MEDIAHANDLING_PEELOFF (6)	To peel labels off the backing and wait for removal.
CLS_MEDIAHANDLING_REWIND (7)	To use a rewinder.

Setter method
setMediaHandling (mediaHandling)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getMediaHandling ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var mediahandling;

print.setMediaHandling(citizen.LabelConst.CLS_MEDIAHANDLING_PAUSE);
mediahandling = print.getMediaHandling();
```

5.5.22. StartOffset property

Syntax
StartOffset

Attribute
Read/Write

Description
Specifies the distance between paper sensor and print head to change the print starting position.

Inch system			Metric system		
Initial value	Minimum value	Max value	Initial value	Minimum value	Max value
0220	0120	0320	0559	0305	0813

*0.01 inches or 0.1 mm

Setter method
setStartOffset(startOffset)

Make sure to set a valid value since the consistency with the selected measurement unit will never be checked.
Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getStartOffset ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example
```
var startoffset;

print.setStartOffset(220);
startoffset = print.getStartOffset();
```

5.5.23. StopOffset property

Syntax
StopOffset

Attribute
Read/Write

Description
Specifies the distance between paper sensor and cutter or peeler to change the print stop position.

Note that the initial values are calculated to stop paper at an appropriate position. An insufficient value truncates the printed label, an exceeded value truncates the next label.

Media handling	Inch system			Metric system		
	Initial value	Minimum value	Max value	Initial value	Minimum value	Max value
None	0000	0000	9999	0000	0000	9999
Cutter	0100			0254		
Peel Off	0050			0127		
Tear Off	0070			0178		

*0.01 inches or 0.1 mm

Setter method
setStopOffset (stopOffset)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getStopOffset ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example
```
var stopoffset;

print.setStopOffset(240);
stopoffset = print.getStopOffset();
```

5.5.24. LabelSensor property

Syntax
LabelSensor

Attribute
Read/Write

Description
Sets the label sensor type.

Value	Description
CLS_SELSENSOR_NONE (0)	None. This assumes that a continuous paper roll is used. Therefore a valid length must be set in the ContinuousMediaLength property, returns CLS_E_ILLEGAL (1101) otherwise.
CLS_SELSENSOR_SEETHROUGH (1)	Gap sensor. To detect a gap between labels or a notch on tags.
CLS_SELSENSOR_REFLECT (2)	Reflect sensor. Detect a black mark on the back side of the label.

Setter method
setLabelSensor (labelSensor)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getLabelSensor ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
//Other than "CLS_SELSENSOR_NONE"
var labelsensor;

print.setLabelSensor(citizen.LabelConst.CLS_SELSENSOR_SEETHROUGH);
labelsensor = print.getLabelSensor();


//"CLS_SELSENSOR_NONE"
var labelsensor;

print.setContinuousMediaLength(100);
print.setLabelSensor(citizen.LabelConst.CLS_SELSENSOR_NONE);
labelsensor = print.getLabelSensor();
```

5.5.25. PrintMethod property

Syntax
PrintMethod

Attribute
Read/Write

Description
Sets the print method to either thermal transfer or direct thermal.

Value	Description
CLS_PRTMETHOD_TT (0)	Thermal transfer
CLS_PRTMETHOD_DT (1)	Direct thermal

Setter method
setPrintMethod (printMethod)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getPrintMethod ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example

```
var printmethod;

print.setPrintMethod(citizen.LabelConst.CLS_PRTMETHOD_DT);
printmethod = print.getPrintMethod();
```

5.5.26. SensorLocation property

Syntax
SensorLocation

Attribute
Read/Write

Description
Selects the sensor to use for the models which have multiple paper sensors, front and rear.
The choice will be stored in the non-volatile flash memory and will be kept until overwritten.

Value	Description
CLS_SENS_LOCATION_FRONT (0)	Front sensor
CLS_SENS_LOCATION_ADJUSTABLE (1)	Rear sensor

Setter method
setSensorLocation (sensorLocation)

Returns CLS_SUCCESS (0) on success. See "5.4. Return value" for the error codes.

Getter method
getSensorLocation ()

Returns the set value. If nothing has been set, it returns CLS_PROPERTY_DEFAULT(999999).

Example
```
var location;

print.setSensorLocation(citizen.LabelConst.CLS_SENS_LOCATION_ADJUSTABLE);
location = print.getSensorLocation();
```

5.5.27. CommandInterpreterInAction property

Syntax
CommandInterpreterInAction

Attribute
Read

Description
Shows the printer status if the interpreter is processing commands.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getCommandInterpreterInAction ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.28. PaperError property

Syntax
PaperError

Attribute
Read

Description
Shows the printer status if there is a paper error.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getPaperError ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.29. RibbonEnd property

Syntax
RibbonEnd

Attribute
Read

Description
Shows the printer status if there is a ribbon end error.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getRibbonEnd ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.30. BatchProcessing property

Syntax
BatchProcessing

Attribute
Read

Description
Shows the status if the printer is processing batch print jobs.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getBatchProcessing ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.31. Printing property

Syntax
Printing

Attribute
Read

Description
Shows the status if the printer is currently printing.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getPrinting ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.32. Pause property

Syntax
Pause

Attribute
Read

Description
Shows the status if the printer is in the pause mode.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getPause ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.33. WaitingForPeeling property

Syntax
WaitingForPeeling

Attribute
Read

Description
Shows the status if the printer is in the wait for peeling status.

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getWaitingForPeeling ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.34.PrintHeadLowTemp property

Syntax
PrintHeadLowTemp

Attribute
Read

Description
Shows the printer status if there is a print head low temp error. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getPrintHeadLowTemp ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.35.PrintHeadFailure property

Syntax
PrintHeadFailure

Attribute
Read

Description
Shows the printer status if there is a print head failure. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getPrintHeadFailure ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.36.PrintHeadOverheat property

Syntax
PrintHeadOverheat

Attribute
Read

Description
Shows the printer status if there is a print head over heat error. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getPrintHeadOverheat ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.37.MechanismOpen property

Syntax
MechanismOpen

Attribute
Read

Description
Shows the printer status if there is a mechanism open error. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getMechanismOpen ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.38.AutoCutterError property

Syntax
AutoCutterError

Attribute
Read

Description
Shows the printer status if there is an auto cutter error. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getAutoCutterError ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.39.FanMotorError property

Syntax
FanMotorError

Attribute
Read

Description
Shows the printer status if there is a fan motor error. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getFanMotorError ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.5.40.MiscError property

Syntax
MiscError

Attribute
Read

Description
Shows the printer status if there is a misc error. (Supported by IF5-EFX1 of wired LAN I/F)

Note that the status properties won't be updated automatically.
printerCheck method checks the status and updates the value, must be called before referring to this property.

Setter method
none

Getter method
getMiscError ()

Returns one of the following values.

Value	Description
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	Nothing has been set.

Example
See printerCheck method.

5.6. LabelDesign class

5.6.1. Constructor

Syntax
citizen.LabelDesign ()

Parameters
none

Description
Creates an instance of the LabelDesign class.

Return value
none

Example
```
var design = new citizen.LabelDesign();
```

5.6.2. drawTextPtrFont method

Syntax

drawTextPtrFont (data, locale, font, rotation, hexp, vexp, size, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	* Unsupported characters will be replaced with "?" text.
locale	[IN]	Locale	CLS_LOCALE_JP: Japan CLS_LOCALE_OTHER: Other CLS_LOCALE_CN: Chinese CLS_LOCALE_KR: Korean
font	[IN]	Font typeface	CLS_PRT_FNT_0: SystemFont 0 CLS_PRT_FNT_1: SystemFont 1 CLS_PRT_FNT_2: SystemFont 2 CLS_PRT_FNT_3: SystemFont 3 CLS_PRT_FNT_4: SystemFont 4 CLS_PRT_FNT_5: SystemFont 5 CLS_PRT_FNT_6: SystemFont 6 CLS_PRT_FNT_7: SystemFont 7 CLS_PRT_FNT_8: SystemFont 8 CLS_PRT_FNT_TRIUMVIRATE: Smooth font CLS_PRT_FNT_TRIUMVIRATE_B: Smooth font (bold) CLS_PRT_FNT_KANJI: Kanji (left to right) CLS_PRT_FNT_KANJIT: Kanji (top to bottom)
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
hexp	[IN]	Horizontal magnification	1 – 24
vexp	[IN]	Vertical magnification	1 – 24
size	[IN]	Font size	CLS_PRT_FNT_SIZE_4: 4pt CLS_PRT_FNT_SIZE_5: 5pt CLS_PRT_FNT_SIZE_6: 6pt CLS_PRT_FNT_SIZE_8: 8pt CLS_PRT_FNT_SIZE_10: 10pt CLS_PRT_FNT_SIZE_12: 12pt CLS_PRT_FNT_SIZE_14: 14pt CLS_PRT_FNT_SIZE_18: 18pt CLS_PRT_FNT_SIZE_24: 24pt CLS_PRT_FNT_SIZE_30: 30pt CLS_PRT_FNT_SIZE_36: 36pt CLS_PRT_FNT_SIZE_48: 48pt CLS_PRT_FNT_KANJI_SIZE_16: Kanji 16dot CLS_PRT_FNT_KANJI_SIZE_24: Kanji 24dot CLS_PRT_FNT_KANJI_SIZE_32: Kanji 32dot * CLS_PRT_FNT_KANJI_SIZE_48: Kanji 48dot * * Not supported in locale for Chinese model.
x	[IN]	X-coordinate	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws characters by using a printer device font with specifying options such as rotation, magnification, size and coordinates.

Available font sizes depend on typeface.
Kanji: 16, 24, 32, 48 *dots
Smooth font: 4, 5, 6, 8, 10, 12, 14, 18, 24, 30, 36, 48
Other: Fixed sizes depend on typeface.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
//Locale:OTHER
design.drawTextPtrFont("drawTextPtrFont",
        citizen.LabelConst.CLS_LOCALE_OTHER,
        citizen.LabelConst.CLS_PRT_FNT_TRIUMVIRATE,
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1,
        citizen.LabelConst.CLS_PRT_FNT_SIZE_10,
        100, 100);


//Locale:JP
design.drawTextPtrFont("テキスト印刷(プリンタフォント)",
        citizen.LabelConst.CLS_LOCALE_JP,
        citizen.LabelConst.CLS_PRT_FNT_KANJI,
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1,
        citizen.LabelConst.CLS_PRT_FNT_KANJI_SIZE_16, 100, 300);


//Locale:CN
design.drawTextPtrFont("测试打印",
        citizen.LabelConst.CLS_LOCALE_CN,
        citizen.LabelConst.CLS_PRT_FNT_KANJI,
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1,
        citizen.LabelConst.CLS_PRT_FNT_KANJI_SIZE_16, 100, 300);


//Locale:KR
design.drawTextPtrFont("테스트 인쇄",
        citizen.LabelConst.CLS_LOCALE_KR,
        citizen.LabelConst.CLS_PRT_FNT_KANJI,
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1,
        citizen.LabelConst.CLS_PRT_FNT_KANJI_SIZE_16, 100, 300);
```

5.6.3. drawTextDLFont method

Syntax

drawTextDLFont (data, encoding, fontID, rotation, hexp, vexp, point, x, y)

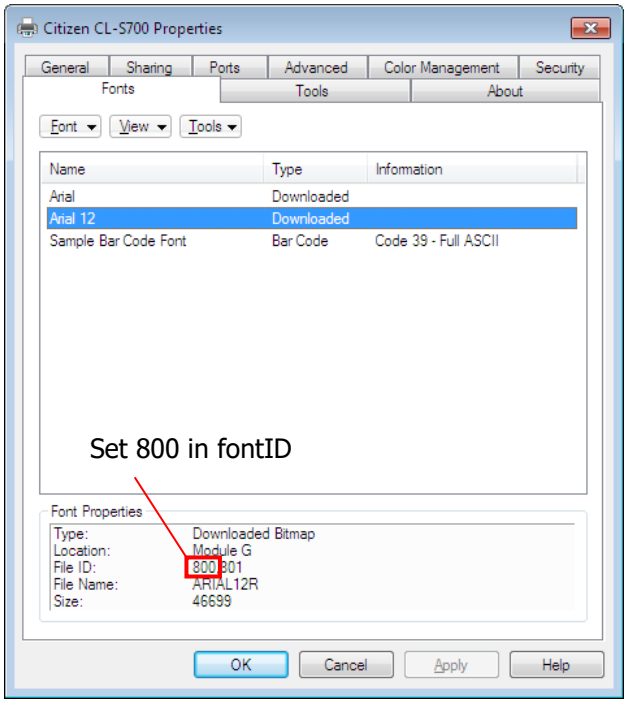
Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	* Unsupported characters will be replaced with "?" text.
encoding	[IN]	Character encoding	See "5.7.2. Predefined Constants".
fontID	[IN]	Font ID stored in the printer	Numeric characters - Bitmap font Alphanumeric characters - TrueType font
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
hexp	[IN]	Horizontal magnification	1 - 24
vexp	[IN]	Vertical magnification	1 - 24
point	[IN]	Font size	001 - 999
x	[IN]	X-coordinate	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws characters by using a downloaded font with specifying options such as rotation, magnification, size and coordinates.

In case of bitmap font, the driver shows two IDs. Use the smaller number of ID.



Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
//TrueType font
design.drawTextDLFont("TrueType",
        citizen.LabelConst.CLS_ENC_CDPG_IBM850, "S50",
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1, 12, 100, 100);


//Bitmap font
design.drawTextDLFont("Bitmap",
        citizen.LabelConst.CLS_ENC_CDPG_IBM850, "800",
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1, 12, 100, 200);
```

5.6.4. drawNVBitmap method

Syntax

drawNVBitmap (name, hexp, vexp, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
name	[IN]	Graphic file name	ASCII characters except below: -Underscore cannot be the first character. -The following symbols. □/ : * ? " < > |
hexp	[IN]	Horizontal magnification	1 - 24
vexp	[IN]	Vertical magnification	1 - 24
x	[IN]	X-coordinate	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws a graphic by using a graphic image stored in the printer, with specifying options such as magnification and coordinates. This doesn't check the file availability or validity.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawNVBitmap("Dice_1.bmp", 1, 1, 10, 10);
```

5.6.5. drawBitmap method

Syntax

drawBitmap (image, rotation, width, height, x, y, resolution, measurementUnit)

Parameters

Parameter	[IN/OUT]	Description	Setting range
image	[IN]	Image object	
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
width	[IN]	Drawing width [Pixel]	
height	[IN]	Drawing height [Pixel]	
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	
resolution	[IN]	Resolution [dpi]	CLS_PRT_RES_203(203dpi) CLS_PRT_RES_300(300dpi) * 203 dpi by default.
measurementUnit	[IN]	Printer Metric / Inch setting	CLS_UNIT_MILLI CLS_UNIT_INCH * CLS_UNIT_INCH by default.

Description

Draws a graphic image stored in the local computer, with specifying options such as rotation, size and coordinates.
What this method does internally is to store a graphic image into the printer and to print the stored graphic image. This doesn't check the availability or validity of the stored file. When specifying image rotation, it is necessary to specify the resolution and unit selection settings according to the printer for reference position correction.
Supported graphic file formats are BMP/GIF/EXIF/JPG/PNG/TIFF.

The graphic will be resized based on the width/height parameters with keeping the aspect ratio and taking the maximum available size to fit.

Example: The sizes will be "200x50" in the case below.
Original sizes: 400x100 pixels
Width parameter: 200
Height parameter: 200

When 0 is set to either width or height, the sizes will be calculated based on another (non-zero) parameter. When both parameters are zero, the original sizes will be used.

Example：The sizes will be "800x200" in the case below.
Original sizes: 400x100 pixels
Width parameter: 0
Height parameter: 200

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
var image = new Image();

//Omit the image reading process

design.drawBitmap(image,
        citizen.LabelConst.CLS_RT_RIGHT90, 0, 0, 10, 10,
        citizen.LabelConst.CLS_PRT_RES_203,
        citizen.LabelConst.CLS_UNIT_INCH);
```

5.6.6. drawBarCode method

Syntax

drawBarCode (data, symbology, rotation, thick, narrow, height, x, y, showText)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	ASCII code character string.
symbology	[IN]	Barcode type	CLS_BCS_CODE39: Code 3 of 9 CLS_BCS_UPCA: UPC-A CLS_BCS_UPCE: UPC-E CLS_BCS_INTERLEAVED25: Interleaved 2 of 5 CLS_BCS_CODE128: Code 128 CLS_BCS_EAN13: EAN-13(JAN-13) CLS_BCS_EAN8: EAN-8(JAN-8) CLS_BCS_HIBC: HIBC CLS_BCS_CODABAR: CODABAR(NW-7) CLS_BCS_INT25: Int 2 of 5 CLS_BCS_PLESSEY: Plessey CLS_BCS_CASECODE: CASE CODE CLS_BCS_UPC2DIG: UPC 2DIG ADD CLS_BCS_UPC5DIG: UPC 5DIG ADD CLS_BCS_CODE93: Code93 CLS_BCS_ITF14: ITF-14 CLS_BCS_ZIP: ZIP CLS_BCS_ITF16: IFT-16 CLS_BCS_UCCEAN128: UCC/EAN-128 CLS_BCS_INDUSTRIAL25: Industrial 2 of 5 CLS_BCS_UCCEAN128KMART: UCC/EAN-128(for K-MART) CLS_BCS_COOP25: COOP 2 of 5 CLS_BCS_UCCEAN128RANDOMWEIGHT: UCC/EAN-128 Random Weight CLS_BCS_TELEPEN: Telepen
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
thick	[IN]	Thick bar width	1-24
narrow	[IN]	Narrow bar width	1-24
height	[IN]	Height of barcode	001-999
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	
showText	[IN]	Human readable text	CLS_BCS_TEXT_HIDE: Hide CLS_BCS_TEXT_SHOW: Show

Description

Draws a 1D (one-dimensional) barcode with specifying options such as barcode type, rotation, thick/narrow bar width, coordinates and human readable text.

* Supported thick/narrow bar ratio or data type depends on symbology. Refer to the command reference for details.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
// Code3of9
var code39 = "ABC123456789";
design.drawBarCode(code39, citizen.LabelConst.CLS_BCS_CODE39,
        citizen.LabelConst.CLS_RT_NORMAL, 6, 2, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UPC-A
var upca = "01234567890";
design.drawBarCode(upca, citizen.LabelConst.CLS_BCS_UPCA,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UPC-E
var upce = "123456";
design.drawBarCode(upce, citizen.LabelConst.CLS_BCS_UPCE,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// Interleaved2of5
var interleaved25 = "1234567890";
design.drawBarCode(interleaved25, citizen.LabelConst.CLS_BCS_INTERLEAVED25,
        citizen.LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// Code128
var code128 = "1234567890";
design.drawBarCode(code128, citizen.LabelConst.CLS_BCS_CODE128,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 4, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// EAN-13
var ean13 = "123456789012";
design.drawBarCode(ean13, citizen.LabelConst.CLS_BCS_EAN13,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 3, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// EAN-8
var ean8 = "1234567";
design.drawBarCode(ean8, citizen.LabelConst.CLS_BCS_EAN8,
        citizen.LabelConst.CLS_RT_NORMAL, 4, 4, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// HIBC
var hibc = "1234567890";
design.drawBarCode(hibc, citizen.LabelConst.CLS_BCS_HIBC,
        citizen.LabelConst.CLS_RT_NORMAL, 6, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// CODABAR
var codabar = "a1234567890b";
design.drawBarCode(codabar, citizen.LabelConst.CLS_BCS_CODABAR,
        citizen.LabelConst.CLS_RT_NORMAL, 6, 2, 40, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);
```

```
// Interleaved2of5 W/BARS
var int25 = "1234567890";
design.drawBarCode(int25, citizen.LabelConst.CLS_BCS_INT25,
        citizen.LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// PLESSEY
var plessey = "1234567890";
design.drawBarCode(plessey, citizen.LabelConst.CLS_BCS_PLESSEY,
        citizen.LabelConst.CLS_RT_NORMAL, 4, 2, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// CASE CODE
var casecode = "1234567890123";
design.drawBarCode(casecode, citizen.LabelConst.CLS_BCS_CASECODE,
        citizen.LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UPC 2DIG ADD
var upca = "01234567890";
var upc2dig = "12";

design.drawBarCode(upca, citizen.LabelConst.CLS_BCS_UPCA,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);

design.drawBarCode(upc2dig, citizen.LabelConst.CLS_BCS_UPC2DIG,
        citizen.LabelConst.CLS_RT_NORMAL, 4, 2, 50, 130, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UPC 5DIG ADD
var upca = "01234567890";
var upc5dig = "12345";

design.drawBarCode(upca, citizen.LabelConst.CLS_BCS_UPCA,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);

design.drawBarCode(upc5dig, citizen.LabelConst.CLS_BCS_UPC5DIG,
        citizen.LabelConst.CLS_RT_NORMAL, 4, 2, 50, 130, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);

// Code93
var code93 = "1234ABCD";
design.drawBarCode(code93, citizen.LabelConst.CLS_BCS_CODE93,
        citizen.LabelConst.CLS_RT_NORMAL, 6, 6, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// ITF-14
var itf14 = "1234567890123";
design.drawBarCode(itf14, citizen.LabelConst.CLS_BCS_ITF14,
        citizen.LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);
```

```
// ZIP
var zip = "123456789";
design.drawBarCode(zip, citizen.LabelConst.CLS_BCS_ZIP,
        citizen.LabelConst.CLS_RT_NORMAL, 1, 1, 10, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// ITF-16
var itf16 = "123456789012345";
design.drawBarCode(itf16, citizen.LabelConst.CLS_BCS_ITF16,
        citizen.LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UCC/EAN-128
var uccean128 = "12345678901234567890";
design.drawBarCode(uccean128, citizen.LabelConst.CLS_BCS_UCCEAN128,
        citizen.LabelConst.CLS_RT_NORMAL, 4, 4, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// Industrial2of5
var industrial25 = "1234567890";
design.drawBarCode(industrial25, citizen.LabelConst.CLS_BCS_INDUSTRIAL25,
        citizen.LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UCC/EAN-128(for K-MART)
var uccean128kmart = "12345678901234567890";
design.drawBarCode(uccean128kmart, citizen.LabelConst.CLS_BCS_UCCEAN128KMART,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 3, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// COOP 2 of 5
var coop25 = "0123456789";
design.drawBarCode(coop25, citizen.LabelConst.CLS_BCS_COOP25,
        citizen.LabelConst.CLS_RT_NORMAL, 10, 4, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// UCC/EAN-128 Random Weight
var uccean128randomweight = "12345678901234567890123456789099999";
design.drawBarCode(uccean128randomweight,
        citizen.LabelConst.CLS_BCS_UCCEAN128RANDOMWEIGHT,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 10,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);


// Telepen
var telepen = "1234567890";
design.drawBarCode(telepen, citizen.LabelConst.CLS_BCS_TELEPEN,
        citizen.LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
        citizen.LabelConst.CLS_BCS_TEXT_SHOW);
```

5.6.7. drawMaxiCode method

Syntax

drawMaxiCode (arryData, rotation, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
arryData	[IN]	String data	ASCII code character string array
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws the UPS MaxiCode barcode with specifying options such as rotation and coordinates.
The following information are required in the data parameter in order as below:

1) 5-digit - Zip code
2) 4-digit -+4 Zip code
3) 3-digit – Country Code
4) 3-digit - Class of Service
5) Up to 84 digits other information

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
var data = new Array(5);
data[0] = "90501";
data[1] = "6282";
data[2] = "840";
data[3] = "001";
data[4] = "1Z00004951";
design.drawMaxiCode(data, citizen.LabelConst.CLS_RT_NORMAL, 100, 0);
```

5.6.8. drawPDF417 method

Syntax

drawPDF417 (data, encoding, rotation, exp, ecLevel, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	* The characters available with the specified encoding.
encoding	[IN]	Character encoding	See "5.7.2. Predefined Constants".
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
exp	[IN]	Magnification	1-5
ecLevel	[IN]	Error correction level	CLS_PDF417_EC_LEVEL_0: Level 0 CLS_PDF417_EC_LEVEL_1: Level 1 CLS_PDF417_EC_LEVEL_2: Level 2 CLS_PDF417_EC_LEVEL_3: Level 3 CLS_PDF417_EC_LEVEL_4: Level 4 CLS_PDF417_EC_LEVEL_5: Level 5 CLS_PDF417_EC_LEVEL_6: Level 6 CLS_PDF417_EC_LEVEL_7: Level 7 CLS_PDF417_EC_LEVEL_8: Level 8
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws the PDF417 barcode with specifying options such as character encoding, rotation, magnification, EC level and coordinates.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawPDF417("0123456789",
        citizen.LabelConst.CLS_ENC_CDPG_IBM850,
        citizen.LabelConst.CLS_RT_NORMAL, 3,
        citizen.LabelConst.CLS_PDF417_EC_LEVEL_0, 0, 0);
```

5.6.9. drawDataMatrix method

Syntax

drawDataMatrix (data, encoding, rotation, exp, ecLevel, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	-Numeric: Up to 3,116 characters -Alphanumeric: Up to 2,335 characters * The characters available with the specified encoding.
encoding	[IN]	Character encoding	See #9 in "5.7.2. Predefined Constants".
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
exp	[IN]	Magnification	1-15
ecLevel	[IN]	Error correction level	CLS_DATAMATRIX_EC_LEVEL_200：200
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws the DataMatrix barcode with specifying options such as character encoding, rotation, magnification, EC level and coordinates.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawDataMatrix("0123456789",
        citizen.LabelConst.CLS_ENC_CDPG_IBM850,
        citizen.LabelConst.CLS_RT_NORMAL, 15,
        citizen.LabelConst.CLS_DATAMATRIX_EC_LEVEL_200, 0, 0);
```

5.6.10. drawQRCode method

Syntax

drawQRCode (data, encoding, rotation, exp, ecLevel, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	* The characters available with the specified encoding.
encoding	[IN]	Character encoding	See #9 in "5.7.2. Predefined Constants".
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
exp	[IN]	Magnification	1-15
ecLevel	[IN]	Error correction level	CLS_QRCODE_EC_LEVEL_L: Level L (7%) CLS_QRCODE_EC_LEVEL_M: Level M (15%) CLS_QRCODE_EC_LEVEL_Q: Level Q (25%) CLS_QRCODE_EC_LEVEL_H: Level H (30%)
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws the QR code with specifying options such as character encoding, rotation, magnification, EC level and coordinates.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawQRCode("ｱｲｳｴｵ12345",
        citizen.LabelConst.CLS_ENC_CDPG_SHIFT_JIS,
        citizen.LabelConst.CLS_RT_NORMAL, 10,
        citizen.LabelConst.CLS_QRCODE_EC_LEVEL_H, 100, 100);
```

5.6.11. drawAztec method

Syntax

drawAztec (data, encoding, rotation, exp, ecLevel, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	String data	* The characters available with the specified encoding.
encoding	[IN]	Character encoding	See #9 in "5.7.2. Predefined Constants".
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
exp	[IN]	Magnification	1-15
ecLevel	[IN]	Error correction level	CLS_AXTEC_EC_LEVEL_000: Level 0 (Error correction ratio 23%)
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws the Aztec barcode with specifying options such as character encoding, rotation, magnification, EC level and coordinates.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawAztec("0123456789",
        citizen.LabelConst.CLS_ENC_CDPG_IBM850,
        citizen.LabelConst.CLS_RT_NORMAL, 10,
        citizen.LabelConst.CLS_AXTEC_EC_LEVEL_000, 0, 0);
```

5.6.12. drawGS1DataBar method

Syntax

drawGS1DataBar (arryData, type, rotation, exp, x, y)

Parameters

Parameter	[IN/OUT]	Description	Setting range
arryData	[IN]	String data	ASCII code character string.
type	[IN]	Barcode type	CLS_GS1_DATABAR_OMNI_DIRECTIONAL: Omni-dirctional CLS_GS1_DATABAR_COMPOSITE: Conposite CLS_GS1_DATABAR_TRUNCATION: Truncation CLS_GS1_DATABAR_STACKED: Stacked CLS_GS1_DATABAR_STACKED_OMNI_DIRECTIONAL: Stacked Omni-dirctional CLS_GS1_DATABAR_LIMITED: Limited CLS_GS1_DATABAR_EXPANDED: Expanded
rotation	[IN]	Direction of rotation	CLS_RT_NORMAL: No rotation CLS_RT_RIGHT90: Rotate CW 90 CLS_RT_ROTATE180: Rotate CW 180 CLS_RT_LEFT90: Rotate CCW 90
exp	[IN]	Magnification	1-15
x	[IN]	X-coordinate	0000-9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Y-coordinate	

Description

Draws the GS1DataBar barcode with specifying options such as rotation, magnification and coordinates.

* The available text types vary depending on the symbology. Refer to the "GS1 DataBar (RSS) in the command reference.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
//GS1 DataBar Omni-Directional
var omnidirectional = new Array("1234567890123");
design.drawGS1DataBar(omnidirectional,
        citizen.LabelConst.CLS_GS1_DATABAR_OMNI_DIRECTIONAL,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);


//GS1 DataBar Composite
var composit = new Array(2);
composit[0] = "1234567890123";
composit[1] = "1234567890-07/07/07";
design.drawGS1DataBar(composit,
        citizen.LabelConst.CLS_GS1_DATABAR_COMPOSITE,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);


// GS1 DataBar Truncation
var truncation = new Array("1234567890123");
design.drawGS1DataBar(truncation,
        citizen.LabelConst.CLS_GS1_DATABAR_TRUNCATION,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);
```

```
// GS1 DataBar Stacked
var stacked = new Array("1234567890123");
design.drawGS1DataBar(stacked,
        citizen.LabelConst.CLS_GS1_DATABAR_STACKED,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);


// GS1 DataBar Stacked-Omni-Directional
var stackedOD = new Array("1234567890123");
design.drawGS1DataBar(stackedOD,
        citizen.LabelConst.CLS_GS1_DATABAR_STACKED_OMNI_DIRECTIONAL,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);


// GS1 DataBar Limited
var limited = new Array("1234567890123");
design.drawGS1DataBar(limited,
        citizen.LabelConst.CLS_GS1_DATABAR_LIMITED,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);


// GS1 DataBar Expanded
var expanded = new Array("041234567890123");
design.drawGS1DataBar(expanded,
        citizen.LabelConst.CLS_GS1_DATABAR_EXPANDED,
        citizen.LabelConst.CLS_RT_NORMAL, 3, 10, 10);
```

5.6.13. drawLine method

Syntax
drawLine (x1, y1, x2, y2, thickness)

Parameters

Parameter	[IN/OUT]	Description	Setting range
x1	[IN]	Start position (X-coordinate)	0000-9999 * The origin is at bottom-left.(0, 0)
y1	[IN]	Start position (Y-coordinate)	
x2	[IN]	End position (X-coordinate)	
y2	[IN]	End position (Y-coordinate)	
thickness	[IN]	Line width (Reference point is center)	0000-9999

Description
Draws a line of the specified width.
Start and stop positions will be located at the center of the line. Each coordinate must be within the range of 0000-9999. Moreover the whole line image (rectangle) including the line thickness must be within the range of 0000-9999 as well. Returns CLS_E_ILLEGAL(1101) otherwise.

Return value
Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example
```
//FormatAttribute:OR
print.setFormatAttribute(1);

//Bar
design.drawLine(20, 30, 20, 300, 10);

//Horizontal line
design.drawLine(16, 34, 200, 34, 10);
```

5.6.14. drawRect method

Syntax

drawRect (x, y, width, height, thickness)

Parameters

Parameter	[IN/OUT]	Description	Setting range
x	[IN]	Start position (X-coordinate)	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Start position (Y-coordinate)	
width	[IN]	Box width	0000 - 9999
height	[IN]	Box height	
thickness	[IN]	Line width	0000 - 9999

Description

Draws a box of the specified width, height and line width. Thicker line width expands the line inward and doesn't affect the outline size.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawRect(20, 30, 180, 280, 10);
```

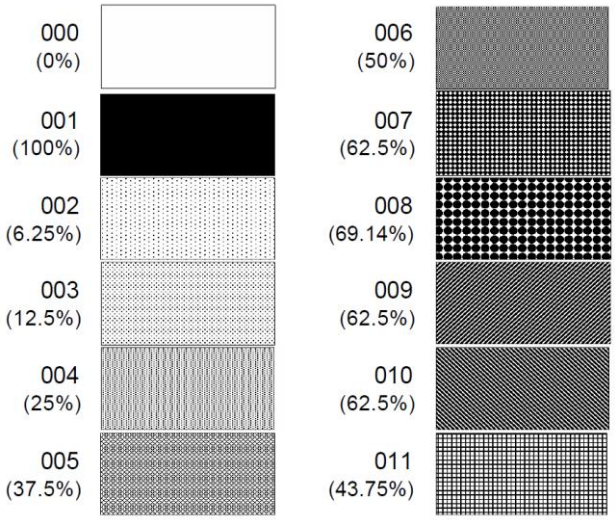
5.6.15. fillRect method

Syntax
fillRect (x, y, width, height, pattern)

Parameters

Parameter	[IN/OUT]	Description	Setting range
x	[IN]	Start position (X-coordinate)	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Start position (Y-coordinate)	
width	[IN]	Box width	0000 - 9999
height	[IN]	Box height	
pattern	[IN]	Shaded pattern	CLS_SHADED_PTN_0: Shaded pattern 000 CLS_SHADED_PTN_1: Shaded pattern 001 CLS_SHADED_PTN_2: Shaded pattern 002 CLS_SHADED_PTN_3: Shaded pattern 003 CLS_SHADED_PTN_4: Shaded pattern 004 CLS_SHADED_PTN_5: Shaded pattern 005 CLS_SHADED_PTN_6: Shaded pattern 006 CLS_SHADED_PTN_7: Shaded pattern 007 CLS_SHADED_PTN_8: Shaded pattern 008 CLS_SHADED_PTN_9: Shaded pattern 009 CLS_SHADED_PTN_10: Shaded pattern 010 CLS_SHADED_PTN_11: Shaded pattern 011

Description
Draws a box of the specified width, height and shaded pattern.



Return value
Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example
```
design.fillRect(20, 30, 180, 280, citizen.LabelConst.CLS_SHADED_PTN_10);
```

5.6.16. drawCircle method

Syntax

drawCircle (x, y, radius)

Parameters

Parameter	[IN/OUT]	Description	Setting range
x	[IN]	Start position (X-coordinate, center)	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Start position (Y-coordinate, center)	
radius	[IN]	Radius	0000 - 0398

Description

Draws a circle of the specified radius.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
design.drawCircle(50, 50, 15);
```

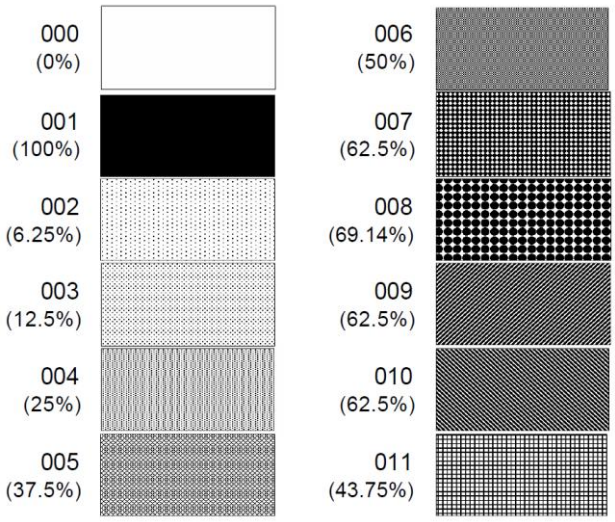
5.6.17. fillCircle method

Syntax
fillCircle (x, y, radius, pattern)

Parameters

Parameter	[IN/OUT]	Description	Setting range
x	[IN]	Start position (X-coordinate, center)	0000 - 9999 * The origin is at bottom-left.(0, 0)
y	[IN]	Start position (Y-coordinate, center)	
radius	[IN]	Radius	0000 - 0398
pattern	[IN]	Shaded pattern	CLS_SHADED_PTN_0: Shaded pattern 000 CLS_SHADED_PTN_1: Shaded pattern 001 CLS_SHADED_PTN_2: Shaded pattern 002 CLS_SHADED_PTN_3: Shaded pattern 003 CLS_SHADED_PTN_4: Shaded pattern 004 CLS_SHADED_PTN_5: Shaded pattern 005 CLS_SHADED_PTN_6: Shaded pattern 006 CLS_SHADED_PTN_7: Shaded pattern 007 CLS_SHADED_PTN_8: Shaded pattern 008 CLS_SHADED_PTN_9: Shaded pattern 009 CLS_SHADED_PTN_10: Shaded pattern 010 CLS_SHADED_PTN_11: Shaded pattern 011

Description
Draws a circle of the specified radius and shaded pattern.



Return value
Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example
```
design.fillCircle(100, 100, 40, citizen.LabelConst.CLS_SHADED_PTN_2);
```

5.6.18. drawPolygon method

Syntax

drawPolygon (arryX, arryY)

Parameters

Parameter	[IN/OUT]	Description	Setting range
arryX	[IN]	X points	0000 - 9999 * The origin is at bottom-left.(0, 0)
arryY	[IN]	Y points	

Description

Draws a closed polygon defined by arrays of x and y coordinates. Each pair of (x, y) coordinates defines a point. The number of elements must match. Returns CLS_E_ILLEGAL(1101) otherwise.

Return value

Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example

```
var arryX = new Array(100, 200, 250, 150);
var arryY = new Array(100, 100, 200, 200);

//Draw a parallelogram
design.drawPolygon(arryX, arryY);
```

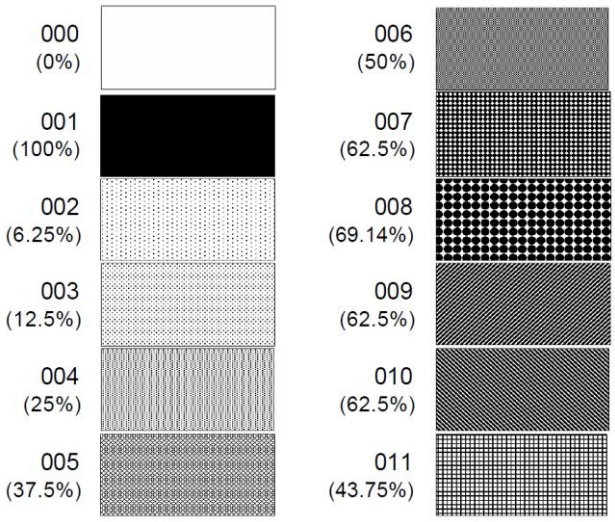
5.6.19. fillPolygon method

Syntax
fillPolygon (arryX, arryY, pattern)

Parameters

Parameter	[IN/OUT]	Description	Setting range
arryX	[IN]	X points	0000 - 9999 * The origin is at bottom-left.(0, 0)
arryY	[IN]	Y points	
pattern	[IN]	Shaded pattern	CLS_SHADED_PTN_0: Shaded pattern 000 CLS_SHADED_PTN_1: Shaded pattern 001 CLS_SHADED_PTN_2: Shaded pattern 002 CLS_SHADED_PTN_3: Shaded pattern 003 CLS_SHADED_PTN_4: Shaded pattern 004 CLS_SHADED_PTN_5: Shaded pattern 005 CLS_SHADED_PTN_6: Shaded pattern 006 CLS_SHADED_PTN_7: Shaded pattern 007 CLS_SHADED_PTN_8: Shaded pattern 008 CLS_SHADED_PTN_9: Shaded pattern 009 CLS_SHADED_PTN_10: Shaded pattern 010 CLS_SHADED_PTN_11: Shaded pattern 011

Description
Draws a closed polygon defined by arrays of x and y coordinates, with the specified shaded pattern. Each pair of (x, y) coordinates defines a point. The number of elements must match. Returns CLS_E_ILLEGAL(1101) otherwise.



Return value
Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example
```
var arryX = new Array(100, 200, 250, 150);
var arryY = new Array(100, 100, 200, 200);

// Draw a parallelogram
design.fillPolygon(arryX, arryY, citizen.LabelConst.CLS_SHADED_PTN_3);
```

5.6.20. embedRawDesignCommand method

Syntax
embedRawDesignCommand (data)

Parameters

Parameter	[IN/OUT]	Description	Setting range
data	[IN]	Design command	Data of binary.

Description
Embeds raw printer commands to the label design. This allows adding raw label format commands individually.

The SendData method in the LabelPrint class requires complete label commands (both system level commands and label format commands).

Return value
Returns CLS_SUCCESS(0) on success, an error code otherwise. See "5.4. Return value" for the error codes.

Example
```
To add a box command,
1X1100002000100b0300025001000100:

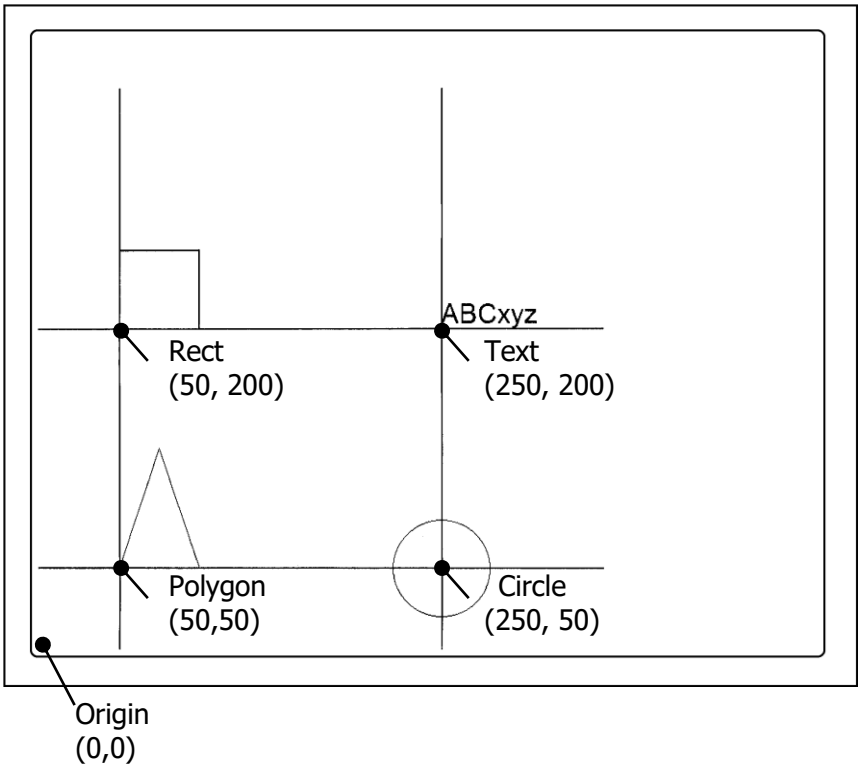
var hexdata = new Array(
        0x31, 0x58, 0x31, 0x31, 0x30, 0x30, 0x30, 0x30, 0x32, 0x30,
        0x30, 0x30, 0x31, 0x30, 0x30, 0x62, 0x30, 0x33, 0x30, 0x30,
        0x30, 0x32, 0x35, 0x30, 0x30, 0x31, 0x30, 0x30, 0x30, 0x31,
        0x30, 0x30, 0x0D, 0x0A );

design.embedRawDesignCommand(hexdata);
```

5.7. Appendix

5.7.1. Specifying object position

The coordinate origin of the label design is at bottom-left. The measurement unit, either inches or millimeters, is set in the printer. The MeasurementUnit property in the LabelPrint class allows setting the measurement unit.



Example

```
// Polygon
arryX = new Array(50, 75, 100);
arryY = new Array(50, 125, 50);
design.drawPolygon(arryX, arryY);;

// Rect
design.drawRect(50, 200, 50, 50, 1);

// Circle
design.drawCircle(250, 50, 30);

// Text
design.drawTextPtrFont("ABCxyz", citizen.LabelConst.CLS_LOCALE_JP,
        citizen.LabelConst.CLS_PRT_FNT_5, citizen.LabelConst.CLS_RT_NORMAL,
        1, 1, citizen.LabelConst.CLS_PRT_FNT_SIZE_6, 250, 200);

// Line
design.drawLine(0, 200, 350, 200, 1);       // Line-1
design.drawLine(50, 0, 50, 350, 1);         // Line-2
design.drawLine(0, 50, 350, 50, 1);         // Line-3
design.drawLine(250, 0, 250, 350, 1);       // Line-4
```

5.7.2. Predefined Constants

No	Type	Name	Data Type	Value	Description
1	Result/Error	CLS_SUCCESS	int	0	Successfully completed
		CLS_E_ILLEGAL	int	1101	Unsupported or invalid parameter
2	Property default value	CLS_PROPERTY_DEFAULT	int	999999	Default value of property
3	Status of printer	CLS_STS_NO	int	0	Status: NO
		CLS_STS_YES	int	1	Status: YES
4	Locale	CLS_LOCALE_JP	int	0	Japanese model
		CLS_LOCALE_OTHER	int	1	Non-Japanese model
		CLS_LOCALE_CN	int	2	Chinese model
		CLS_LOCALE_KR	int	3	Korean model
5	Font typeface	CLS_PRT_FNT_0	int	0	System font:0
		CLS_PRT_FNT_1	int	1	System font:1
		CLS_PRT_FNT_2	int	2	System font:2
		CLS_PRT_FNT_3	int	3	System font:3
		CLS_PRT_FNT_4	int	4	System font:4
		CLS_PRT_FNT_5	int	5	System font:5
		CLS_PRT_FNT_6	int	6	System font:6
		CLS_PRT_FNT_7	int	7	System font:7
		CLS_PRT_FNT_8	int	8	System font:8
		CLS_PRT_FNT_TRIUMVIRATE	int	9	Smooth font (Triumvirate)
		CLS_PRT_FNT_TRIUMVIRATE_B	int	10	Smooth font (Triumvirate Bold)
		CLS_PRT_FNT_KANJI	int	11	Kanji (Writing from left to right)
		CLS_PRT_FNT_KANJIT	int	12	Kanji (Writing from top to bottom)
6	Font size	CLS_PRT_FNT_SIZE_4	int	0	4pt
		CLS_PRT_FNT_SIZE_5	int	1	5pt
		CLS_PRT_FNT_SIZE_6	int	2	6pt
		CLS_PRT_FNT_SIZE_8	int	3	8pt
		CLS_PRT_FNT_SIZE_10	int	4	10pt
		CLS_PRT_FNT_SIZE_12	int	5	12pt
		CLS_PRT_FNT_SIZE_14	int	6	14pt
		CLS_PRT_FNT_SIZE_18	int	7	18pt
		CLS_PRT_FNT_SIZE_24	int	8	24pt
		CLS_PRT_FNT_SIZE_30	int	9	30pt
		CLS_PRT_FNT_SIZE_36	int	10	36pt
		CLS_PRT_FNT_SIZE_48	int	11	48pt
		CLS_PRT_FNT_KANJI_SIZE_16	int	100	Kanji 16dot
		CLS_PRT_FNT_KANJI_SIZE_24	int	101	Kanji 24dot
		CLS_PRT_FNT_KANJI_SIZE_32	int	102	Kanji 32dot
		CLS_PRT_FNT_KANJI_SIZE_48	int	103	Kanji 48dot
7	Encoding	CLS_ENC_CDPG_DEFAULT	int	0	Default value
		CLS_ENC_CDPG_IBM437	int	437	IBM437 OEM USA
		CLS_ENC_CDPG_IBM850	int	850	ibm850 Western European(DOS)
		CLS_ENC_CDPG_IBM852	int	852	ibm852 Central Europe (DOS)
		CLS_ENC_CDPG_IBM857	int	857	ibm857 Turkish (DOS)
		CLS_ENC_CDPG_IBM860	int	860	IBM860 Portuguese (DOS)
		CLS_ENC_CDPG_IBM861	int	861	ibm861 Icelandic (DOS)

		CLS_ENC_CDPG_IBM863	int	863	IBM863 French (Canada)(DOS)
		CLS_ENC_CDPG_IBM864	int	864	IBM864 Arabic
		CLS_ENC_CDPG_IBM865	int	865	IBM865 Norwegian (DOS)
		CLS_ENC_CDPG_CP866	int	866	cp866 Cyrillic (DOS)
		CLS_ENC_CDPG_SHIFT_JIS	int	932	shift_jis Japanese (Shift JIS)
		CLS_ENC_CDPG_BIG5	int	950	big5 traditional Chinese (Big5)
		CLS_ENC_CDPG_WINDOWS_1252	int	1252	Windows-1252Western European (Windows)
		CLS_ENC_CDPG_EUC_KR	int	51949	euc-kr Korean (EUC)
		CLS_ENC_CDPG_GB18030	int	54936	GB18030 simplified Chinese (GB18030)
8	Direction of rotation	CLS_RT_NORMAL	int	1	No rotation
		CLS_RT_RIGHT90	int	2	Rotate 90 CW
		CLS_RT_ROTATE180	int	3	Rotate 180
		CLS_RT_LEFT90	int	4	Rotate 90 CCW
9	Barcode symbology	CLS_BCS_CODE39	int	100	Code 3 of 9
		CLS_BCS_UPCA	int	101	UPC-A
		CLS_BCS_UPCE	int	102	UPC-E
		CLS_BCS_INTERLEAVED25	int	103	Interleaved 2 of 5
		CLS_BCS_CODE128	int	104	Code 128
		CLS_BCS_EAN13	int	105	EAN-13(JAN-13)
		CLS_BCS_EAN8	int	106	EAN-8(JAN-8)
		CLS_BCS_HIBC	int	107	HIBC
		CLS_BCS_CODABAR	int	108	CODABAR(NW-7)
		CLS_BCS_INT25	int	109	Int 2 of 5
		CLS_BCS_PLESSEY	int	110	Plessey
		CLS_BCS_CASECODE	int	111	CASE CODE
		CLS_BCS_UPC2DIG	int	112	UPC 2DIG ADD (supplementary code of 2 digits for UPC)
		CLS_BCS_UPC5DIG	int	113	UPC 5DIG ADD (supplementary code of 5 digits for UPC)
		CLS_BCS_CODE93	int	114	Code93
		CLS_BCS_ITF14	int	115	(Japanese model)ITF-14
		CLS_BCS_ZIP	int	116	(Non-Japanese model)ZIP
		CLS_BCS_ITF16	int	117	(Japanese model)ITF-16
		CLS_BCS_UCCEAN128	int	118	(Non-Japanese model)UCC/EAN-128
		CLS_BCS_INDUSTRIAL25	int	119	(Japanese model)Industrial 2 of 5
		CLS_BCS_UCCEAN128KMART	int	120	(Non-Japanese model) UCC/EAN-128(for K-MART)
		CLS_BCS_COOP25	int	121	(Japanese model)COOP 2 of 5
		CLS_BCS_UCCEAN128RANDOMWEIGHT	int	122	(Non-Japanese model) UCC/EAN-128 Random Weight
		CLS_BCS_TELEPEN	int	123	Telepen
10	Human readable text of barcode	CLS_BCS_TEXT_HIDE	int	0	Hide
		CLS_BCS_TEXT_SHOW	int	1	Show
11	Error correction level (PDF417)	CLS_PDF417_EC_LEVEL_0	int	0	Level 0
		CLS_PDF417_EC_LEVEL_1	int	1	Level 1
		CLS_PDF417_EC_LEVEL_2	int	2	Level 2
		CLS_PDF417_EC_LEVEL_3	int	3	Level 3
		CLS_PDF417_EC_LEVEL_4	int	4	Level 4

		CLS_PDF417_EC_LEVEL_5	int	5	Level 5
		CLS_PDF417_EC_LEVEL_6	int	6	Level 6
		CLS_PDF417_EC_LEVEL_7	int	7	Level 7
		CLS_PDF417_EC_LEVEL_8	int	8	Level 8
12	Error correction level (Data Matrix)	CLS_DATAMATRIX_EC_LEVEL_200	int	200	
13	Error correction level (QR Code)	CLS_QRCODE_EC_LEVEL_L	int	0	Error correction level L (7%)
		CLS_QRCODE_EC_LEVEL_M	int	1	Error correction level M (15%)
		CLS_QRCODE_EC_LEVEL_Q	int	2	Error correction level Q (25%)
		CLS_QRCODE_EC_LEVEL_H	int	3	Error correction level H (30%)
14	Error correction level (Aztec)	CLS_AXTEC_EC_LEVEL_000	int	0	Error correction ratio 23%
15	Barcode symbology (GS1 DataBar)	CLS_GS1_DATABAR_OMNI_DIRECTIONAL	int	0	GS1DataBar Omni-dirctional
		CLS_GS1_DATABAR_COMPOSITE	int	1	GS1DataBar Conposite
		CLS_GS1_DATABAR_TRUNCATION	int	2	GS1DataBar Truncation
		CLS_GS1_DATABAR_STACKED	int	3	GS1DataBar Stacked
		CLS_GS1_DATABAR_STACKED_OMNI_DIRECTIONAL	int	4	GS1DataBar Stacked Omni-dirctional
		CLS_GS1_DATABAR_LIMITED	int	5	GS1DataBar Limited
		CLS_GS1_DATABAR_EXPANDED	int	6	GS1DataBar Expanded
16	Shaded pattern	CLS_SHADED_PTN_0	int	0	Shaded pattern：000
		CLS_SHADED_PTN_1	int	1	Shaded pattern：001
		CLS_SHADED_PTN_2	int	2	Shaded pattern：002
		CLS_SHADED_PTN_3	int	3	Shaded pattern：003
		CLS_SHADED_PTN_4	int	4	Shaded pattern：004
		CLS_SHADED_PTN_5	int	5	Shaded pattern：005
		CLS_SHADED_PTN_6	int	6	Shaded pattern：006
		CLS_SHADED_PTN_7	int	7	Shaded pattern：007
		CLS_SHADED_PTN_8	int	8	Shaded pattern：008
		CLS_SHADED_PTN_9	int	9	Shaded pattern：009
		CLS_SHADED_PTN_10	int	10	Shaded pattern：010
		CLS_SHADED_PTN_11	int	11	Shaded pattern：011
17	Measurement unit	CLS_UNIT_MILLI	int	0	Metric
		CLS_UNIT_INCH	int	1	Inch
18	Speed setting	CLS_SPEEDSETTING_1	int	1	1：1.0inch(25.4mm) / sec
		CLS_SPEEDSETTING_2	int	2	2：2.0inch(50.8mm) / sec
		CLS_SPEEDSETTING_3	int	3	3：3.0inch(76.2mm) / sec
		CLS_SPEEDSETTING_4	int	4	4：4.0inch(101.6mm) / sec
		CLS_SPEEDSETTING_5	int	5	5：5.0inch(127.0mm) / sec
		CLS_SPEEDSETTING_6	int	6	6：6.0inch(152.4mm) / sec
		CLS_SPEEDSETTING_7	int	7	7：7.0inch(177.8mm) / sec
		CLS_SPEEDSETTING_8	int	8	8：8.0inch(203.2mm) / sec
		CLS_SPEEDSETTING_9	int	9	9：9.0inch(228.6mm) / sec
		CLS_SPEEDSETTING_A	int	10	A：1.0inch(25.4mm) / sec
		CLS_SPEEDSETTING_B	int	11	B：1.0inch(25.4mm) / sec
		CLS_SPEEDSETTING_C	int	12	C：2.0inch(50.8mm) / sec
		CLS_SPEEDSETTING_D	int	13	D：2.0inch(50.8mm) / sec

		CLS_SPEEDSETTING_E	int	14	E:3.0inch(76.2mm) / sec
		CLS_SPEEDSETTING_F	int	15	F:3.0inch(76.2mm) / sec
		CLS_SPEEDSETTING_G	int	16	G:4.0inch(101.6mm) / sec
		CLS_SPEEDSETTING_H	int	17	H:4.0inch(101.6mm) / sec
		CLS_SPEEDSETTING_I	int	18	I:5.0inch(127.0mm) / sec
		CLS_SPEEDSETTING_J	int	19	J:5.0inch(127.0mm) / sec
		CLS_SPEEDSETTING_K	int	20	K:6.0inch(152.4mm) / sec
		CLS_SPEEDSETTING_L	int	21	L:6.0inch(152.4mm) / sec
		CLS_SPEEDSETTING_M	int	22	M:7.0inch(177.8mm) / sec
		CLS_SPEEDSETTING_N	int	23	N:7.0inch(177.8mm) / sec
		CLS_SPEEDSETTING_O	int	24	O:8.0inch(203.2mm) / sec
		CLS_SPEEDSETTING_P	int	25	P:8.0inch(203.2mm) / sec
		CLS_SPEEDSETTING_Q	int	26	Q:9.0inch(228.6mm) / sec
		CLS_SPEEDSETTING_R	int	27	R:9.0inch(228.6mm) / sec
		CLS_SPEEDSETTING_S	int	28	S:10.0inch(254.0mm) / sec
		CLS_SPEEDSETTING_T	int	29	T:10.0inch(254.0mm) / sec
		CLS_SPEEDSETTING_U	int	30	U:11.0inch(279.4mm) / sec
		CLS_SPEEDSETTING_V	int	31	V:11.0inch(279.4mm) / sec
		CLS_SPEEDSETTING_W	int	32	W:12.0inch(304.8mm) / sec
		CLS_SPEEDSETTING_X	int	33	X:12.0inch(304.8mm) / sec
19	Media handling	CLS_MEDIAHANDLING_NONE	int	0	None
		CLS_MEDIAHANDLING_TEAROFF	int	1	Tear-off
		CLS_MEDIAHANDLING_DISPENSES	int	2	Dispense
		CLS_MEDIAHANDLING_PAUSE	int	3	Pause
		CLS_MEDIAHANDLING_CUT	int	4	Cut
		CLS_MEDIAHANDLING_CUTANDPAUSE	int	5	Cut and Pause
		CLS_MEDIAHANDLING_PEELOFF	int	6	Peel-off
		CLS_MEDIAHANDLING_REWIND	int	7	Rewind
20	Label sensor	CLS_SELSENSOR_NONE	int	0	None
		CLS_SELSENSOR_SEETHROUGH	int	1	See Through
		CLS_SELSENSOR_REFLECT	int	2	Reflect
21	Print method	CLS_PRTMETHOD_TT	int	0	Thermal transfer
		CLS_PRTMETHOD_DT	int	1	Direct thermal
22	Sensor location	CLS_SENS_LOCATION_FRONT	int	0	Front sensor
		CLS_SENS_LOCATION_ADJUSTABLE	int	1	Rear sensor

CITIZEN XML Print Service
JavaScript Label Print SDK - Programming Manual
July 23, 2026 Version 1.03
CITIZEN SYSTEMS JAPAN CO., LTD.
https://csj.citizen.co.jp/