# ePOS-Print API

## User’s Manual

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Overview</strong></td>
<td>Describes the features and development environment.</td>
</tr>
<tr>
<td><strong>Sample Program</strong></td>
<td>Describes how to use the sample program and how to build a system.</td>
</tr>
<tr>
<td><strong>Programming Guide</strong></td>
<td>Describes how to write programs in Web application development.</td>
</tr>
<tr>
<td><strong>ePOS-Print API</strong></td>
<td>Describes the ePOS-Print API.</td>
</tr>
<tr>
<td><strong>ePOS-Print Canvas API</strong></td>
<td>Describes the ePOS-Print CanvasAPI.</td>
</tr>
<tr>
<td><strong>ePOS-Print Editor</strong></td>
<td>Describes the ePOS-Print Editor.</td>
</tr>
<tr>
<td><strong>Appendix</strong></td>
<td>Describes the specifications for printers used for ePOS-Print, how to use the rendering of images in HTML5 Canvas.</td>
</tr>
</tbody>
</table>
Cautions

• No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Seiko Epson Corporation.

• The contents of this document are subject to change without notice. Please contact us for the latest information.

• While every precaution has taken in the preparation of this document, Seiko Epson Corporation assumes no responsibility for errors or omissions.

• Neither is any liability assumed for damages resulting from the use of the information contained herein.

• Neither Seiko Epson Corporation nor its affiliates shall be liable to the purchaser of this product or third parties for damages, losses, costs, or expenses incurred by the purchaser or third parties as a result of: accident, misuse, or abuse of this product or unauthorized modifications, repairs, or alterations to this product, or (excluding the U.S.) failure to strictly comply with Seiko Epson Corporation’s operating and maintenance instructions.

• Seiko Epson Corporation shall not be liable against any damages or problems arising from the use of any options or any consumable products other than those designated as Original EPSON Products or EPSON Approved Products by Seiko Epson Corporation.

Trademarks

EPSON is a registered trademark of Seiko Epson Corporation.

Exceed Your Vision and ESC/POS are registered trademarks or trademarks of Seiko Epson Corporation.

Microsoft®, Windows®, Internet Explorer® and Visual Studio® are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.

Safari™ and TrueType® are either registered trademarks or trademarks of Apple Inc. in the United States and other countries.

Android™ and Google Chrome™ are either registered trademarks or trademarks of Google Inc. in the United States and other countries.

Mozilla® and Firefox® are either registered trademarks or trademarks of Mozilla Foundation in the United States and other countries.

IOS® is registered trademarks or trademarks of Cisco in the United States and other countries.

All other trademarks are the property of their respective owners and used for identification purpose only.

ESC/POS® Command System

EPSON has been taking industry’s initiatives with its own POS printer command system (ESC/POS). ESC/POS has a large number of commands including patented ones. Its high scalability enables users to build versatile POS systems. The system is compatible with all types of EPSON POS printers (excluding the TM-C100) and displays. Moreover, its flexibility makes it easy to upgrade the future. The functionality and the user-friendliness is valued around the world.

© Seiko Epson Corporation 2011-2014. All rights reserved.
For Safety

Key to Symbols

The symbols in this manual are identified by their level of importance, as defined below. Read the following carefully before handling the product.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Provides information that must be observed to avoid damage to your equipment or a malfunction.</td>
</tr>
<tr>
<td>✍️</td>
<td>Provides important information and useful tips.</td>
</tr>
</tbody>
</table>

Restriction of Use

When this product is used for applications requiring high reliability/safety such as transportation devices related to aviation, rail, marine, automotive etc.; disaster prevention devices; various safety devices etc; or functional/precision devices etc, you should use this product only after giving consideration to including fail-safes and redundancies into your design to maintain safety and total system reliability. Because this product was not intended for use in applications requiring extremely high reliability/safety such as aerospace equipment, main communication equipment, nuclear power control equipment, or medical equipment related to direct medical care etc, please make your own judgment on this product’s suitability after a full evaluation.
About this Manual

Aim of the Manual

This manual is intended to provide development engineers with all the information necessary for building/designing a printing system or developing/designing an ePOS-Print printer application.

In this manual, "ePOS-Print supported printer" is a generic term for the TM-i series and TM printers that support the ePOS-Print API.

The TM-i series in this manual is a generic term for the following printers.

- TM-T88V-i
- TM-T70-i
- TM-L90-i

Manual Content

The manual is made up of the following sections:

Chapter 1 Overview
Chapter 2 Sample Program
Chapter 3 Programming Guide
Chapter 4 ePOS-Print API
Chapter 5 ePOS-Print Canvas API
Chapter 6 ePOS-Print Editor
Appendix Printer specifications
  Rendering in HTML5 Canvas
  Windows Store Apps
## Contents

- **For Safety** ........................................................................................................................................... 3
  - Key to Symbols ................................................................................................................................... 3
- **Restriction of Use** .............................................................................................................................. 3
- **About this Manual** ............................................................................................................................. 4
  - Aim of the Manual ............................................................................................................................... 4
  - Manual Content ................................................................................................................................... 4
- **Contents** ............................................................................................................................................... 5

### Overview ................................................................................................................................................ 11

- **Overview of ePOS-Print** ..................................................................................................................... 11
  - Features ............................................................................................................................................... 12
  - Print Example ..................................................................................................................................... 13
  - Print Flow .......................................................................................................................................... 15
  - Features ............................................................................................................................................... 16
- **Operating Environment** ..................................................................................................................... 17
  - Applications environment .................................................................................................................... 17
  - Terminal ............................................................................................................................................. 17
  - ePOS-Print Supported TM printer ......................................................................................................... 17
  - Printers That Can Be Controlled .......................................................................................................... 18
- **System Construction Example** ........................................................................................................... 19
  - Registering a Web Application Into the Web Server ........................................................................... 19
  - Registering a Web Application Into a TM-i ........................................................................................ 20
  - Registering a Web application to a cloud ............................................................................................ 21
  - Specifying the Print Job ID from the Application ................................................................................. 22
  - Spooler Function ................................................................................................................................. 23
- **Contents in the Package** ..................................................................................................................... 26
  - Download ............................................................................................................................................ 27
- **Restrictions** ......................................................................................................................................... 28

### Sample Program .................................................................................................................................... 29

- **Sample Program System Overview** ...................................................................................................... 29
  - Sample Program Screen ....................................................................................................................... 29
  - Print Image ........................................................................................................................................... 31
  - Program Flow ....................................................................................................................................... 32
- **Operating Environment** ..................................................................................................................... 35
  - TM-i ..................................................................................................................................................... 35
- **Environment Settings** .......................................................................................................................... 36
  - Registration of Sample Program .......................................................................................................... 38
  - Sample Program Settings .................................................................................................................... 39
Constructor ......................................................................................................................................... 135
send method ..................................................................................................................................... 136
getPrintJobStatus method ....................................................................................................... 138
open method ..................................................................................................................................... 139
close method ....................................................................................................................................... 140
address property ................................................................................................................................ 141
enabled property ................................................................................................................................ 142
interval property ................................................................................................................................ 142
status property .................................................................................................................................. 143
battery property .................................................................................................................................. 143
timeout property ................................................................................................................................ 144
onreceive event .................................................................................................................................. 145
onerror event .................................................................................................................................... 148
onstatuschange event ......................................................................................................................... 149
onbatterystatuschange event ............................................................................................................ 149
ononline event .................................................................................................................................. 150
onoffline event .................................................................................................................................. 150
onpoweroff event ............................................................................................................................... 151
oncoverclose event ............................................................................................................................. 151
oncoveropen event .............................................................................................................................. 152
onpapereok event ............................................................................................................................... 152
onpapernearend event ......................................................................................................................... 153
onpapereend event ............................................................................................................................. 153
ondrawerclosed event ......................................................................................................................... 154
ondraweropen event ............................................................................................................................ 154
onbatteryok event .............................................................................................................................. 155
onbatterylow event ............................................................................................................................ 155
addPageBegin method ......................................................................................................................... 106
addPageEnd method ............................................................................................................................ 107
addPageArea method .......................................................................................................................... 108
addPageDirection method .................................................................................................................. 110
addPagePosition method .................................................................................................................... 112
addPageLine method ........................................................................................................................... 114
addPageRectangle method .................................................................................................................. 116
addCut method .................................................................................................................................... 118
addPulse method .................................................................................................................................. 119
addSound method ................................................................................................................................ 121
addLayout method ................................................................................................................................ 123
addRecovery method ........................................................................................................................... 127
addReset method ................................................................................................................................ 128
addCommand method .......................................................................................................................... 129
toString method .................................................................................................................................. 130
halftone property ................................................................................................................................. 131
brightness property .............................................................................................................................. 132
force property .................................................................................................................................... 133
message property .................................................................................................................................. 134
ePOS-Print Canvas API ................................................................. 157

- List of ePOS-Print Canvas API functions ............................................................... 157
  window.epson.CanvasPrint Components ................................................................. 157

- ePOS-Print Canvas API Object ........................................................................... 159
  Constructor ........................................................................................................... 159
  print method ......................................................................................................... 160
  getPrintJobStatus method .................................................................................... 162
  open method ......................................................................................................... 163
  close method ......................................................................................................... 164
  recover method ..................................................................................................... 165
  reset method ......................................................................................................... 165
  address property ................................................................................................... 166
  enabled property ................................................................................................. 167
  interval property ................................................................................................. 168
  status property ..................................................................................................... 169
  battery property ................................................................................................. 170
  timeout property ................................................................................................. 171
  halftone property ............................................................................................... 172
  brightness property ............................................................................................ 173
  cut property ......................................................................................................... 174
  mode property ...................................................................................................... 175
  align property ...................................................................................................... 176
  color property ...................................................................................................... 177
  feed property ........................................................................................................ 178
  paper property ..................................................................................................... 179
  layout property ..................................................................................................... 180
  onreceive event .................................................................................................... 185
  onerror event ....................................................................................................... 188
  onstatuschange event .......................................................................................... 189
  onbatterystatuschange event ............................................................................... 190
  onbatteryok event ............................................................................................... 191
  onbatterylow event .............................................................................................. 191
  ononline event .................................................................................................... 192
  onoffline event ..................................................................................................... 192
  onpoweroff event ................................................................................................. 193
  oncoverok event ................................................................................................. 193
  oncoveropen event .............................................................................................. 194
  onpaperok event ................................................................................................. 194
  onpapermearend event ......................................................................................... 195
  onpaperend event ............................................................................................... 195
  ondrawerclosed event ......................................................................................... 196
  ondraweropen event ............................................................................................ 196

ePOS-Print Editor .................................................................................................. 197

- ePOS-Print Editor Operating Environment .......................................................... 197
- Displaying ePOS-Print Editor ............................................................................... 197
- Setting .................................................................................................................. 197
- Creating a Sample Code ....................................................................................... 199
  Print ....................................................................................................................... 201
  Import .................................................................................................................... 202
Appendix.............................................................................................................203

■ Printer specifications ........................................................................................................ 203
  TM-L90-i/ TM-L90 ........................................................................................................... 203
  TM-T70-i/ TM-T70 ........................................................................................................... 205
  TM-T70-i (Multi-language model) ................................................................................... 206
  TM-T88V-i/ TM-T88V ...................................................................................................... 207
  TM-P60ii ......................................................................................................................... 209
  TM-P60ii with Peeler ....................................................................................................... 211
  TM-P80 ........................................................................................................................... 213
  TM-T20 ........................................................................................................................... 215
  TM-T70 (Multi-language model) ..................................................................................... 217
  TM-T88IV ....................................................................................................................... 218
  TM-T90 ........................................................................................................................... 220
  TM-U220 ........................................................................................................................ 222

■ Rendering in HTML5 Canvas .......................................................................................... 224
  Rendering Text (canvas-print-text.html) ......................................................................... 224
  Rendering Images (canvas-print-image.html) .................................................................... 226
  Rendering Graphics (canvas-print-graph.html) .................................................................. 228
  Rendering Handwritten Images (canvas-print-hand.html) ............................................... 230
  Rendering Barcode (canvas-print-barcode.html) .............................................................. 232
  Rendering Label (canvas-print-label.html) ...................................................................... 234

■ Windows Store Apps .................................................................................................... 236
  Sample Program Screen .................................................................................................. 236
  Environment of Sample Program .................................................................................... 238
  Sample Program Settings ............................................................................................... 239
  Printing ........................................................................................................................... 240
Overview

This chapter describes the features of and the specifications for ePOS-Print API.

Overview of ePOS-Print

ePOS-Print API is a function used to allow printing to ePOS-Print-supported printers under a multi-platform environment. Printing is available from a Web browser on devices such as computers, smart phones and tablets.

In addition, print images rendered in HTML5 Canvas can be printed.

ePOS-Print API provides the API for print commands.

When a print document (Request) is sent via HTTP from the host to the ePOS-Print Service of a ePOS-Print supported TM printer, ePOS-Print processes the printing of that document and returns a response document (Response).
Features

- As long as it is in a network environment, a terminal with an HTML5-supported Web browser can perform printing from anywhere.
- It supports Windows store apps (JavaScript).
- Installation of drivers and plug-ins is not required.
- No PCs or servers are required for printing.
- Allows printing from public and private clouds.
- Allows printing in languages supported in Web browsers.
- Automatically checks the status of the TM printer before printing. There is no need for checking the status of the TM printer in advance.
- Does not respond to a printer's function to automatically send its status (AutoStatusBack). Instead, capable of sending an empty print command and checking the status of the TM printer based on the result of command transmission.
- Printing requests can be made to the printer by specifying the printing job ID. *2
  For details, refer to Specifying the Print Job ID from the Application(p.22).
- Printing data can be sent to the printer without waiting for a printing complete response from the printer. (Spooler) *2
  For details, refer to Spooler Function(p.23).
- Print data can be sent to another printer if the printer sends an error response (PAPER END, COVER OPEN, etc.) or does not return a response. (Print forwarding) *2
  For details, refer to Forwarding(p.25). To change the printer settings, utility programs dedicated to each printer or other utility programs should be used.
- In case of TM-i series, it can print to other TM printer via TM-i.
- Provides ePOS-Print API and ePOS-Print Canvas API.
  <ePOS-Print API>>
  - Allows device fonts to be used for printing.
  - Allows barcode printing.
  <ePOS-Print Canvas API>>
  - Allows printing of images rendered in HTML5 Canvas.
  - Allows TrueType fonts to be used for printing.

*1 ePOS-Print Service Ver.4.1 or later versions supported.
*2 TM-i firmware Ver.4.1 or later versions supported.
Print Example

**ePOS-Print API**

---

**Sample Shop**

VERY VERY

STORE GT XXXX-XXXX

THANK YOU FOR SHOPPING WITH US!

<table>
<thead>
<tr>
<th>Item</th>
<th>Quantity</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>101023 Orange juice</td>
<td>1</td>
<td>1.49</td>
</tr>
<tr>
<td>108956 Chocolate</td>
<td>2</td>
<td>2.10</td>
</tr>
<tr>
<td>000210 GT-Special</td>
<td>1</td>
<td>29.80</td>
</tr>
</tbody>
</table>

Subtotal: 33.39

To Stay Total: 33.39

Cash: 34.00

Change Due: 0.61

---

**Sample Shop**

Matsunoto Nagano

Your Number: **0001**

Please wait until your ticket number is called.

*0001*

Mon Aug 01 2011 16:18:00

---

- Printing a Logo
- Alignment: Center
- Paper Feed
- Paper Feed and Paper Cut
- Printing a raster image
- Printing text in the double-sized width style
- Scale: x 6 (horizontal) and x 4 (vertical)
- Alignment: Center
- Printing a Barcode
ePOS-Print Canvas API

Sample Shop

Enjoy!
FREE Coffee
Expires Mon Aug 01 2011
Sample Shop

No. 012301

Monochrome or Grayscale
Monochrome
Print Flow

1. A Web application is placed.
2. A Web browser displays the Web application.
3. The Web browser sends print data. For an ePOS-Print supported printer, the print data for the unit printer is printed.
4. For a TM-i, the print data is sent to a controllable printer.
5. The data is printed from printers that can be controlled.
6. The ePOS-Print supported TM printer returns a response document to the terminal.
The installed functions vary depending on the model. For details, refer to Printer specifications (p.203).

ePOS-Print API contains the following printing methods. These may be used successively within a single document, however, they cannot be used simultaneously.

<table>
<thead>
<tr>
<th>Print Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print API</td>
<td>This is suited to mainly text printing. Printing is done with the device font mounted on the TM printer. Barcodes can also be printed upon developing them as images on the TM printer side.</td>
</tr>
<tr>
<td>ePOS-Print Canvas API</td>
<td>Highly expressive printing is possible. Images painted on HTML5 Canvas are printed.</td>
</tr>
</tbody>
</table>

Printing functions of ePOS-Print API

- Print setting (alignment/line feed space/text rotation/page mode)
- Character data setting (language/font (device font)/double-sizing/scale/smoothing/print position)
- Character style setting (inversion of black and white/underline/bold)
- Paper feed setting (in dots/in lines)
- Image printing (raster image/NV graphics)
- Barcode printing
  (For barcodes that can be printed by each model, refer to "Printer specifications" on page 203)
- Two dimensional symbol printing
  (For two dimensional symbols that can be printed by each model, refer to "Printer specifications" on page 203.)
- Ruled line setting
- Control of label paper/black mark paper
- Drawer kick function
- Buzzer function
- ESC/POS command transmission
- Response document acquisition (print result/printer status/system error status)
- Paper layout setting (Label paper)
- Recovery from an error
- Reset

Printing functions of ePOS-Print Canvas API

- Printing of images (raster images) rendered in HTML5 Canvas
- Control of label paper/black mark paper
- Feed cut
- Response document acquisition (print result/printer status/system error status)
- Paper layout setting
- Recovery from an error
- Reset
Operating Environment

Applications environment

- HTML5-supported Web browser
  - Windows Internet Explorer 9 or later
  - From Internet Explorer, Web pages (HTTPS) that are securely protected cannot be printed on the TM printer.
  - Mozilla Firefox 3.6 or later
  - Google Chrome 7 or later
  - Safari in iOS4.0 or later
  - Standard browser in Android 2.2 or later
- Windows Store apps
  - JavaScript project

Terminal

Terminal with an HTML5-supported Web browser

ePOS-Print Supported TM printer

**TM-i Series**

- TM-T88V-i
- TM-T70-i
- TM-L90-i

- TM-L90-i does not support TM-i Firmware Ver.4.0 or later.

**TM Printer**

When using the following printers, use ePOS-Print SDK for JavaScript:

- TM-P60II
- TM-P60II with Peeler
- TM-P80

Unable to control other TM printer
**ePOS-Print Service Versions**

The table below shows the relationship between the ePOS-Print Service version and TM-i Firmware version of the TM-i series.

<table>
<thead>
<tr>
<th>ePOS-Print Service</th>
<th>TM-i Firmware</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>1.2x</td>
</tr>
<tr>
<td>2.0</td>
<td>2.0x</td>
</tr>
<tr>
<td>2.1</td>
<td>2.1x</td>
</tr>
<tr>
<td>2.2</td>
<td>-</td>
</tr>
<tr>
<td>3.0</td>
<td>3.0x, 3.1x, 4.0x</td>
</tr>
<tr>
<td>3.2</td>
<td>-</td>
</tr>
<tr>
<td>4.1</td>
<td>4.1x</td>
</tr>
</tbody>
</table>

If API of newly added ePOS-Print Builder is used on unsupported models, schema error is returned and printing cannot be done. The latest version of ePOS-Print API JavaScript is recommended regardless of ePOS-Print Service version installed on a printer. ePOS-Print API JavaScript is bundled with the sample program. For details, refer to "Contents in the Package" on page 26.

**Printers That Can Be Controlled**

TM printers that can be controlled via the TM-i series. Available TM printers are different depending on the TM-i Firmware version installed in the product. For details, refer to Technical Reference Guide for each printer.
System Construction Example

Registering a Web Application Into the Web Server

- **Web Server**
  A Web application is placed.

- **Terminal**
  Executes the Web application using a browser (HTML5-supported Web browser).

- **TM-i**
  Receives/prints print data sent from the Web browser or controls other devices.

- **Printers That Can Be Controlled**
  Print the print data received from the TM-i.
Registering a Web Application Into a TM-i

- **Terminal**
  Executes the Web application using a browser.

- **TM-i**
  Mounts the Web application. Creates and prints the print data using transmission with the input terminal's Web browser.
Registering a Web application to a cloud

- Terminal
  - Executes the Web application using a browser.
- TM Printer (Wireless LAN Model)

Diagram:
- <Web Server>
- Web Application
- Wireless LAN / Router
- <Terminal> Web Browser
- <TM Printer> (Wireless LAN Model)
Specifying the Print Job ID from the Application

A response containing the specified print job ID will be returned when sending a request from the application by specifying the print job ID. (ePOS-Print Service Ver.4.1 or later versions)
Spooler Function

The spooler function is a function used to temporarily store printing data received from the application in the TM-i to carry out background printings sequentially. (TM-i firmware Ver.4.1 or later versions)

TM-i return response data to the application the moment the printing data enters the spooler to make it possible for applications to proceed to the next process without waiting for a printing complete response. The application can use the job ID included in the response data to send an inquiry on the printing status. EPSON TMNet WebConfig is used to configure the spooler.

Perform the following settings. For details, refer to the Technical Reference Guide for each TM-i:
- Settings to enable the spooler
- Print process retry settings

Forwarding can be used to send a printing job to a different printer if the printer does not respond to retry processing. For details, refer to Forwarding(p.25).

Example

1. Send print data A to a TM-i (ePOS-Print Service I/F) from the input terminal.
2. The TM-i will save the print data A to the spooler and will return response data to the input terminal.
3. The TM-i will retrieve the print data and issue a print command to another TM-i printer.
4 Print data B will be sent to the TM-i from the input terminal.

5 The TM-i will save the print data B to the spooler and will return response data to the input terminal.

6 The TM-i will retrieve the print data B and issue a print command to printer A.

7 Print data A is complete.

8 A query on print data A results will be sent to the TM-i from the input terminal.

9 The TM-i will return response data back to the input terminal signifying that print data A completed without error.
1. Send print data to ePOS-Print Service I/F from the input terminal.
2. The TM-i issues a print command to printer A.
3. Printer A does not respond.
4. The TM-i reissues a print command to printer A. If printer A still does not respond to the reissued print command, the print command will be issued to printer B.
5. Print data sent from the input terminal will be printed using printer B.

EPSON TMNet WebConfig is used to configure forwarding settings. For details, refer to the Technical Reference Guide for each TM-i.
Contents in the Package

**Manual**
- ePOS-Print API User’s Manual (This Document)
- ePOS-Print XML User’s Manual
- TM-P60II Technical Reference Guide
- TM-P80 Technical Reference Guide

**Sample Program**
ePOS-Print_Sample_API_V4.x.xE.zip
The following are included:
- epos-print-4.x.x.js (ePOS-Print JavaScript for embedding)
- sample/index.html (Sample program)
- editor/index.html (ePOS-Print Editor)
- win8/ePOS-Print Demo.zip (Windows Store apps sample program)
- Rendering in HTML5 Canvas
  - canvas/canvas-print-text.html (Rendering text)
  - canvas/canvas-print-image.html (Rendering images)
  - canvas/canvas-print-graph.html (Rendering graphics)
  - canvas/canvas-print-hand.html (Rendering handwritten images)
  - canvas/canvas-print-barcode.html (Rendering barcode)
  - canvas/canvas-print-label.html (Rendering label)
- README.txt

**Utility**
- TM-i Series

<table>
<thead>
<tr>
<th>Utility</th>
<th>TM-T88V-i</th>
<th>TM-T70-i</th>
<th>TM-L90-i</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-Dedicated Utility</td>
<td>●</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Memory Switch Setting Utility</td>
<td>-</td>
<td>-</td>
<td>●</td>
</tr>
<tr>
<td>TM Flash Logo Setup Utility (TMFLogo)</td>
<td>-</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>TMNet WinConfig (EpsonNet Config)</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>
Download

For customers in North America, go to the following web site:
http://www.epsonexpert.com/ and follow the on-screen instructions.
For customers in other countries, go to the following web site:
https://download.epson-biz.com/?service=pos
Restrictions

- The drawer and the buzzer cannot be used together.
- The buzzer function cannot be used if the printer is not provided with the buzzer.
- Internet Explorer 9 does not allow printing to the printer to be performed from security-protected Web pages (HTTPS).
- When multiple tones are set for raster images, intermittent printing may occur because the amount of data to print increases and white stripes may appear in the print result.
- The scan quality of barcodes/two-dimensional symbols printed as multiple-tone raster images cannot be guaranteed. Print them as two-tone images.
- If printing was cancelled, perform the following settings to clear the data left in the printer.
  - In the printer DIP switches (memory switches), configure the Busy condition only for the receive buffer full.
  - Disable the command execution (offline). (TM-P60II, TM-P80)
Sample Program

This chapter describes how to use the sample program.

- In this chapter, descriptions are made based on a system configuration using a Web server.
- Descriptions are made assuming that the Web server in this chapter is a Web server configured by using IIS (Microsoft Internet Information Services). If your Web server is used in a different environment, interpret the descriptions accordingly.

Sample Program System Overview

Sample Program Screen

The screen compositions for the sample program are as follows:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Queue Ticket</td>
<td>Prints queue ticket numbers. This is a sample program using the ePOS-Print API.</td>
</tr>
<tr>
<td>2 Coupon</td>
<td>Prints coupons. This is a sample program using the ePOS-Print Canvas API.</td>
</tr>
<tr>
<td>3 Label</td>
<td>Prints labels. This is a sample program using the ePOS-Print API.</td>
</tr>
</tbody>
</table>
### 4 Settings

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Settings</strong></td>
<td>Displays the &quot;Settings&quot; screen. The screen is used to set the following:</td>
</tr>
<tr>
<td>IP address of ePOS-Print supported printer</td>
<td>• IP address of the ePOS-Print supported TM printer</td>
</tr>
<tr>
<td>192.168.192.168</td>
<td>(Default :</td>
</tr>
<tr>
<td></td>
<td>* TM-I: DHCP</td>
</tr>
<tr>
<td></td>
<td>(If an address fails to be assigned via DHCP, the value becomes &quot;192.168.192.168&quot;).</td>
</tr>
<tr>
<td>Device ID of the target printer</td>
<td>* TM Printer:</td>
</tr>
<tr>
<td>local_printer</td>
<td>192.168.192.168</td>
</tr>
<tr>
<td>Print timeout (milliseconds)</td>
<td>)</td>
</tr>
<tr>
<td>60000</td>
<td>• Device ID of the target printer</td>
</tr>
<tr>
<td></td>
<td>(Default : local_printer)</td>
</tr>
<tr>
<td>Print in grayscale (Coupon)</td>
<td>• Print timeout (milliseconds)</td>
</tr>
<tr>
<td>(Only for supported models)</td>
<td>(Default : 60000)</td>
</tr>
<tr>
<td>Set the paper layout (Label)</td>
<td>• Prints coupons in gray scale (Coupon)</td>
</tr>
<tr>
<td>(Only for TM-P600II)</td>
<td>(Default : No)</td>
</tr>
<tr>
<td></td>
<td>• Prints labels with specified layout (Label)</td>
</tr>
<tr>
<td></td>
<td>(Default: No)</td>
</tr>
</tbody>
</table>
Print Image

The sample program prints the following:

Your Number
(ePOS-Print API)

Your Number:
0001

Please wait until your ticket number is called.
Mon Aug 01 2011 16:18:00

Coupon
(ePOS-Print Canvas API)

Sample Shop
Nitobetsu Branch

Your Number:
0001

Enjoy!
FREE Coffee
Expires Mon Aug 01 2011

Label *
(ePOS-Print API)

Name Item A
Color Red
Code *2012001*

*: Die cut label: mount width 58 mm or above
Label size: width 54 mm x height 25.4 mm or above
Program Flow

From its initial display state up to print job completion, the sample program flows as below.

**Queue ticket number issuance (ePOS-Print API)**

1. Open the sample program Web page
2. Settings
   - Settings of IP address of ePOS-Print supported TM printer
   - Device ID of the printer to be controlled
   - Print timeout time
3. Click "Queue Ticket"
4. Creates print data (ePOS-Print Builder Object)
5. Prints (ePOS-Print Object)

Action on the sample program
Action on the customer
**Coupon issuance (ePOS-Print Canvas API)**

1. Open the sample program Web page

2. Settings
   - Settings of IP address of ePOS-Print supported TM printer
   - Device ID of the printer to be used for printing
   - Print timeout time

3. Click “Coupon”

4. Render in HTML5 Canvas

5. Prints (ePOS-Print Canvas API Object)

Action on the sample program
Action on the customer
**Label issuance (ePOS-Print API)**

1. Open the sample program Web page

2. Settings
   - Settings of IP address of ePOS-Print supported TM printer
   - Device ID of the printer to be controlled
   - Print timeout time
   - Settings of with or without layout specification in label printing of TM-P60II

3. Click "Label"

4. Creates print data (ePOS-Print Builder Object)

5. Prints (ePOS-Print Object)

---

Action on the sample program

Action on the customer
Operating Environment

**TM-i**

The system configuration diagram for the sample programs is as below.

- The figure below also describes an example of IP address settings as network settings.
- In the sample program, "Printer to be controlled" is not required. Refer to it if necessary.

---

- Web server/computer to configure the settings
  - (Descriptions here are made assuming that the Web server is the same as the computer to configure the settings.)
- Wireless LAN Router
- TM-i (1 set)
- Terminal
  - Terminal with an HTML5-supported Web browser

---

![Diagram of system configuration](image-url)
Environment Settings

A flow for configuring the environment settings for the sample program is shown as follows:

1. Router Settings
   Configure the settings such as SSID, IP address, DHCP, and allocated IP address. For details, refer to the manual for the device you are using.

2. Computer Settings
   1. Network Settings
      Configure the network settings for the computer such as IP address.
   2. Web Server Configuration
      Configure a Web server on the computer.

3. Registration of Sample Program (p.38)

4. Terminal settings
   Configure the wireless LAN (Wi-Fi) settings of the terminal to match the router settings so as to enable network connection. For details, refer to the manual for the device you are using.

5. Network Setting of ePOS-Print Supported TM printer
   Configure the network settings for an ePOS-Print-supported TM printer such as IP address. Configure the network settings for the computer such as IP address by using EPSON TMNet WebConfig from the Web browser on the computer to configure the settings.
   For the configuration procedure, refer to Technical Reference Guide for each TM printer.

6. Network settings for the printer to be controlled
   Configure these settings by using the network setting utility. For details, refer to the detailed instruction manual for the printer (these settings are not required in this sample program).
7. Device ID Settings
Configure the settings for the computer by using EPSON TMNet WebConfig from a Web browser. For the configuration procedure, refer to Technical Reference Guide for each TM-i. (these settings are not required in this sample program)

8. Print Forwarding Settings
Configure these settings to set forwarding on the TM-i series. For details, refer to Technical Reference Guide for TM-i. (these settings are not required in this sample program)

9. Sample Program Settings (p.39)
Configuration is done from a Web browser (these settings are not required in this sample program).
**Registration of Sample Program**

Register the sample program into the Web server.

- Download ePOS-Print_Sample_API_V4.x.xE.zip.
  For details, refer to Contents in the package (p. 26).

Register the program according to the following procedure:

1. Start the Web server.

2. Explode the sample program (ePOS-Print_Sample_API_V4.x.xE.zip) and then copy the exploded contents into the following folder:
   - Example: Web server configured by using IIS
     - System drive: \Inetpub\wwwroot
   - Copy the sample program as a user with administrator authority.
Sample Program Settings

Configure the settings for the sample program according to the procedure below.

In the sample program, “Device ID Settings” are not required. Refer to it if necessary.

1. Start the Web server.
2. Connect all the printers to the network and turn their power ON.
3. Open the following URL page using the Web browser.
   http://Web server IP address/sample/index.html
4. The sample program page opens. Click (Settings).
The “Settings” screen appears. Specify the following and click (OK).

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP address of the intelligent printer</td>
<td>Specifies the IP address of the ePOS-Print supported TM printer.</td>
</tr>
<tr>
<td></td>
<td>(Default value:</td>
</tr>
<tr>
<td></td>
<td>* TM-i: DHCP</td>
</tr>
<tr>
<td></td>
<td>(If an address fails to be assigned via DHCP, the value becomes &quot;192.168.192.168&quot;).)</td>
</tr>
<tr>
<td></td>
<td>* TM Printer: 192.168.192.168</td>
</tr>
<tr>
<td>Device ID of the target printer</td>
<td>Specifies the Device ID of the printer to print queue ticket numbers and coupons.</td>
</tr>
<tr>
<td></td>
<td>(Default value: local_printer)</td>
</tr>
<tr>
<td>Print timeout (millisecond)</td>
<td>Specifies the timeout time. (default: 60000)</td>
</tr>
<tr>
<td>Print in grayscale (Coupon) (Only for supported models)</td>
<td>Prints coupons in gray scale. (Default: No)</td>
</tr>
<tr>
<td>Set the paper layout (Label) (Only for TM-P60II)</td>
<td>Prints labels with specified layout (Default: No)</td>
</tr>
</tbody>
</table>
Programming Guide

This chapter describes how to write programs in the application development using ePOS-Print.

ePOS-Print API

Print Mode

There are two types of print modes: standard and page modes.

Standard mode

In standard mode, characters are printed line by line. The line feed space is adjusted based on the font size and the height of images, barcodes, etc. This mode is suitable for the type of printing such as printing receipts that requires the paper length to change according to the print space.

Page mode

In page mode, you set a print area, lay out data in it, and print the data in a batch operation. Characters, images, and barcodes are laid out in the print positions (coordinates).

Programming Flow

For the ePOS-Print API, programming is performed based on the following work flow:

1. Embedding of ePOS-Print API (p.42)

2. Print Document Creation (p.43)
   - To create a text print document: (p.44)
   - To create a graphic print document: (p.44)
   - To create a page mode print document (p.45)

3. Transmission of Print Document (p.46)

4. Reception of Print Result (p.47)

- ePOS-Print supported TM printer checks the status of the TM printer to be used for printing and then starts printing operation.
- A status event helps check the status of the TM printer. For details on the procedure, refer to Reception of Status Event (p. 49).
Embedding of ePOS-Print API

The ePOS-Print API is provided so that ePOS-Print can be used from the JavaScript on the client side. It is provided as JavaScript, and its file name is “epos-print-4.x.x.js”. The ePOS-Print API is used by embedding epos-print-4.x.x.js into applications.

**Preparation**

To use the ePOS-Print API, place epos-print-4.x.x.js on the Web server.

**Embedding into Web pages**

Embed the script into the Web page by using the HTML `<script>` tags.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    .
    .
}</script>
</head>
<body>
.
.
</body>
</html>
```
A print document is created using an ePOS-Print Builder object. Create an ePOS-Print Builder object using the constructor for it; create a print document using the object’s methods; and then acquire that print document using the toString method. For details, refer to List of API functions (p.61).

Refer to the following program for print document creation.

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function buildMessage() {
    //Create an ePOS-Print Builder object
    var builder = new epson.ePOSBuilder();
    //Create a print document
    builder.addTextLang('en')
    builder.addTextSmooth(true);
    builder.addTextFont(builder.FONT_A);
    builder.addTextSize(3, 3);
    builder.addText('Hello, World!
    builder.addCut(builder.CUT_FEED);
    //Acquire the print document
    var request = builder.toString();
    alert(request);
}
</script>
</head>
<body>
<button onclick="buildMessage()">Run</button>
</body>
</html>
```
To create a text print document:

To create a text print document, store the font settings into the command buffer using text methods and then create a print document. Refer to the following program.

For the string "Hello World!", to create a print document based on the following settings:

- Font: FontA
- Scale: x 4 (horizontal) and x 4 (vertical)
- Style: Bold

```javascript
function buildMessage() {
  //Create an ePOS-Print Builder object
  var builder = new epson.ePOSBuilder();
  //Create a print document
  //<Configure the print character settings>
  builder.addTextLang('en');
  builder.addTextSmooth(true);
  builder.addTextFont(builder.FONT_A);
  builder.addTextSize(4, 4);
  builder.addTextStyle(false, false, true, undefined);
  //<Specify the print data>
  builder.addText('Hello, World!
');
  builder.addCut(builder.CUT_FEED);
  //Acquire the print document
  var request = builder.toString();
}
```

To create a graphic print document:

To create a graphic print document, store a raster image obtained by rendering an image in HTML5 Canvas into the command buffer using the addImage method. Refer to the following program.

To create a print document for the image file “logo.bmp”

```javascript
function buildMessage() {
  //Create an ePOS-Print Builder object
  var builder = new epson.ePOSBuilder();
  //Render an image in HTML5 Canvas
  var canvas = document.getElementById('canvas');
  var context = canvas.getContext('2d');
  context.drawImage(document.getElementById('logo'), 0, 0, 200, 70);
  //Create a print document
  builder.addTextAlign(builder.ALIGN_CENTER);
  builder.addImage(context, 0, 0, canvas.width, canvas.height, builder.COLOR_1);
  builder.addCut(builder.CUT_FEED);
  //Acquire the print document
  var request = builder.toString();
}
```

This section describes how to print a raster image. In addition, there is also a method of printing graphics registered in the NV memory of the printer. For details, refer to addLogo method (p.90).
To create a page mode print document

When the addPageBegin method is stored in the command buffer, the page mode starts. Store the print area (addPageArea method) and the print start position (addPagePosition method) into the command buffer. Specify the print start position according to the print data. After that, store the methods into the command buffer to create print data. For the end of page mode, store the addPageEnd method into the command buffer.

For the string "Hello World!", to create a print document based on the following settings:

```javascript
function buildMessage() {
    //Create an ePOS-Print Builder object
    var builder = new epson.ePOSBuilder();
    //Create a print document
    //<The page mode starts>
    builder.addPageBegin();
    //<Specify the page mode print area>
    builder.addPageArea(100, 50, 200, 100);
    //<Specify the page mode print position>
    builder.addPagePosition(0, 42);
    //<Specify the print data>
    builder.addTextLang('en');
    builder.addTextFont(builder.FONT_A);
    builder.addTextSize(4, 4);
    builder.addTextStyle(false, false, true, undefined);
    builder.addText('Hello,	World!
');
    //<The page mode ends>
    builder.addPageEnd();
    builder.addCut(builder.CUT_FEED);
    //Acquire the print document
    var request = builder.toString();
}
</script>
```
Transmission of Print Document

A print document is sent using an ePOS-Print object. Create an ePOS-Print object using the constructor and specify the end point address for the printer to be used for printing as well as the print document into the send method to send the document. For the details about the printer end point address, refer to Printer End Point Address (p.46). Refer to the following program.

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function buildMessage() {
    //Create a print document
    var builder = new epson.ePOSBuilder();
    builder.addTextLang('en');
    builder.addTextSmooth(true);
    builder.addTextFont(builder.FONT_A);
    builder.addTextSize(3, 3);
    builder.addText('Hello,	World!
');
    builder.addCut(builder.CUT_FEED);
    var request = builder.toString();
    //Set the end point address
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer&timeout=10000';
    //Create an ePOS-Print object
    var epos = new epson.ePOSPrint(address);
    //Send the print document
    epos.send(request);
}
</script>
</head>
<body>
<button onclick="buildMessage()">Run</button>
</body>
</html>
```

Printer End Point Address

Specify the printer end point address in the following format:
http://(domain)/cgi-bin/epos/service.cgi?devid=(device ID)&timeout=(timeout time)

<table>
<thead>
<tr>
<th>Items to specify</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>Specify IP address or domain of ePOS-Print supported TM printer.</td>
</tr>
<tr>
<td>Device ID</td>
<td>Specifies the printer to be used for printing. Specify device ID registered with EPSON TMNet WebConfig of ePOS-Print supported TM printer.</td>
</tr>
<tr>
<td>Timeout period</td>
<td>Specifies the time to abort the process in milliseconds. The timeout parameter is optional; when it is omitted, 300 seconds (300000) is set. When the timeout period elapses, the print job is canceled; the data already interpreted by the printer before the start of the print abort process is printed.</td>
</tr>
</tbody>
</table>
Reception of Print Result

The print result can be received by setting a callback function using the onreceive property (p. 145) of the ePOS-Print object. The following information is obtained:

- Print result
- Error code
- Printer status

The printer status can be obtained when communication with the printer is possible.

Refer to the following program. For the details about how to program a callback function in detail, refer to Error handling (p.48).

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function buildMessage() {
  //Create a print document
  var builder = new epson.ePOSBuilder();
  builder.addTextLang('en');
  builder.addTextSmooth(true);
  builder.addTextFont(builder.FONT_A);
  builder.addTextSize(3, 3);
  builder.addText('Hello, World!
');
  builder.addCut(builder.CUT_FEED);
  var request = builder.toString();

  var address = 'http://192.168.192.168/cgi-bin/epos/
                   service.cgi?devid=local_printer&timeout=10000';

  //Create an ePOS-Print object
  var epos = new epson.ePOSPrint(address);
  //Set a response receipt callback function
  epos.onreceive = function (res) {
    //When the printing is not successful, display a message
    if (!res.success) {
      alert('A print error occurred');
    }
  }

  //Send the print document
  epos.send(request);
}
</script>
<body>
<button onclick="buildMessage()">Run</button>
</body>
</html>
```
Error handling

Refer to the following program for the error handling method by a callback function.

```javascript
//Create an ePOS-Print object
var epos = new epson.ePOSPrint(address);
// Set a response receipt callback function
epos.onreceive = function (res) {
    // Obtain the print result and error code
    var msg = 'Print' + (res.success ? 'Success' : 'Failure') + '
    Code:' + res.code + '
    Status:
    var asb = res.status;
    if (asb & epos.ASB_NO_RESPONSE) {
        msg += ' No printer response
    }
    if (asb & epos.ASB_PRINT_SUCCESS) {
        msg += ' Print complete
    }
    if (asb & epos.ASB_DRAWER_KICK) {
        msg += ' Status of the drawer kick number 3 connector pin = "H"
    }
    if (asb & epos.ASB_OFF_LINE) {
        msg += ' Offline status
    }
    if (asb & epos.ASB_COVER_OPEN) {
        msg += ' Cover is open
    }
    if (asb & epos.ASB_PAPER_FEED) {
        msg += ' Paper feed switch is feeding paper
    }
    if (asb & epos.ASB_WAIT_ON_LINE) {
        msg += ' Waiting for online recovery
    }
    if (asb & epos.ASB_PANEL_SWITCH) {
        msg += ' Panel switch is ON
    }
    if (asb & epos.ASB_MECHANICAL_ERR) {
        msg += ' Mechanical error generated
    }
    if (asb & epos.ASB_AUTOCUTTER_ERR) {
        msg += ' Auto cutter error generated
    }
    if (asb & epos.ASB_UNRECOVER_ERR) {
        msg += ' Unrecoverable error generated
    }
    if (asb & epos.ASB_AUTORECOVER_ERR) {
        msg += ' Auto recovery error generated
    }
    if (asb & epos.ASB_RECEIPT_NEAR_END) {
        msg += ' No paper in the roll paper near end detector
    }
    if (asb & epos.ASB_RECEIPT_END) {
        msg += ' No paper in the roll paper end detector
    }
    if (asb & epos.ASB_BUZZER) {
        msg += ' Sounding the buzzer (limited model)
    }
    if (asb & epos.ASB_SPOOLER_IS_STOPPED) {
        msg += ' Stop the spooler
    }
    //Display in the dialog box
    alert(msg);
}
```
Reception of Status Event

The status event notification function is used to check the printer status without printing. Refer to the following:

```javascript
//Set the end point address
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer &timeout=10000';

//Create an ePOS-Print Builder object
var builder = new epson.ePOSBuilder(address);

//Set an event callback function (cover open)
epos.oncoveropen = function () {
    alert('coveropen');
};

//Set an event callback function (paper near end)
epos.onpapernearend = function () {
    alert('papernearend');
};

//Enable status event operation
epos.open();
```
For the ePOS-Print Canvas API, programming is performed based on the following workflow:

1. Embedding of ePOS-Print Canvas API (p.51)

2. Rendering in HTML5 Canvas (p.52)

3. Prints an Canvas image (p.53)

4. Reception of Print Result (p.54)

- ePOS-Print supported TM printer starts printing after checking the status of TM printer.
- A status event helps check the status of the TM printer. For details on the procedure, refer to Reception of Status Event (p. 49).
Embedding of ePOS-Print Canvas API

The ePOS-Print Canvas API is provided as JavaScript. And its file name is "epos-print-4.x.x.js". It is used by embedding epos-print-4.x.x.js into applications.

Preparation
To use the ePOS-Print Canvas API, place epos-print-4.x.x.js on the Web server.

Embedding into Web pages
Embed the script into the Web page by using the HTML <script> tags.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>TITLE</title>
  <script type="text/javascript" src="epos-print-4.x.x.js"></script>
  <script type="text/javascript">
    function drawCanvas() {
      //
    }
  </script>
</head>
<body>
  //
</body>
</html>
```
Rendering in HTML5 Canvas

Render an image in HTML5 Canvas.

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function drawCanvas() {
  // Rendering in HTML5 Canvas
  // <Obtain the context>
  var canvas = document.getElementById('myCanvas');
  var context = canvas.getContext('2d');
  // <Render an image>
  context.clearRect(0, 0, 512, 480);
  context.drawImage(document.getElementById('coffee'), 0, 0,
      512, 384);
  context.fillStyle = 'rgba(255, 255, 255, 0.5)';
  context.fillRect(0, 0, 512, 480);
  context.fillStyle = 'rgba(0, 0, 0, 1.0)';
  // <Render a water mark for the image>
  context.drawImage(document.getElementById('wmark'), 0, 0);
  context.drawImage(document.getElementById('wmark'), 256, 324);
  // <Render text>
  context.textAlign = 'center';
  context.textBaseline = 'alphabetic';
  context.font = 'bold normal normal 48px "Times New Roman", serif';
  context.fillText('FREE Coffee', 256, 224);
}
</script>
</head>
<body>
<button onclick="drawCanvas()">Run</button>
<canvas id="myCanvas" width="512" height="480"></canvas>
<img id="coffee" src="img/coffee.jpg" alt="">
<img id="wmark" src="img/wmark.png" alt="">
</body>
</html>
```
Prints an Canvas image

Content drawn in HTML5 Canvas is printed using the ePOS-Print Canvas API. Create an ePOS-Print Canvas API object using the constructor; for the Print method, specify the end point address for the printer to be used for printing as well as the canvas content and whether to select paper cut; and then print a document. For the details about the printer end point address, refer to Printer End Point Address (p.46).

Refer to the following program.

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function drawCanvas() {
  // Rendering in HTML5 Canvas
  //<Obtain the context>
  var canvas = document.getElementById('myCanvas');
  var context = canvas.getContext('2d');
  
  //Set the end point address
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer&timeout=10000';
  //Create an ePOS-Print Canvas API object
  var epos = new epson.CanvasPrint(address);
  //Print
  epos.cut = true;
  epos.print(canvas);
}
</script>
</head>
<body>
<button onclick="drawCanvas()">Run</button>
<canvas id="myCanvas" width="512" height="480"></canvas>
<img id="coffee" src="img/coffee.jpg" alt="" />
<img id="wmark" src="img/wmark.png" alt="" />
</body>
</html>
```

For the details about the printer end point address, refer to Printer End Point Address (p.46).
Reception of Print Result

The print result can be received by setting a callback function using the onreceive property (p. 145) of the ePOS-Print Canvas API object. The following information is obtained:

- Print result
- Error code
- Printer Status

The printer status can be obtained when communication with the printer is possible.

Refer to the following program. For the details about how to program a callback function in detail, refer to Error handling (p.48).

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>TITLE</title>
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function drawCanvas() {
    // Rendering in HTML5 Canvas
    //<Obtain the context>
    var canvas = document.getElementById('myCanvas');
    var context = canvas.getContext('2d');
    
    //Set the end point address
    var address = 'http://192.168.192.168/cgi-bin/epos/
    service.cgi?devid=local_printer&timeout=10000';
    
    //Create an ePOS-Print Canvas API object
    var epos = new epson.CanvasPrint(address);
    //Set a response receipt callback function
    epos.onreceive = function (res) {
        //When the printing is not successful, display a message
        if (!res.success) {
            alert('A print error occurred');
        }
    }
    
    //Print
    epos.cut = true;
    epos.print(canvas);
}
</script>
</head>
<body>
<button onclick="drawCanvas()">Run</button>
<canvas id="myCanvas" width="512" height="480"></canvas>
<img id="coffee" src="img/coffee.jpg" alt="">
<img id="wmark" src="img/wmark.png" alt="">
</body>
</html>
```
Error handling

Refer to the following program for the error handling method by a callback function.

```javascript
var epos = new epson.CanvasPrint(address);
// Set a response receipt callback function
epos.onreceive = function (res) {
    // Obtain the print result and error code
    var msg = '
    // Obtain the printer status
    var asb = res.status;
    if (asb & epos.ASB_NO_RESPONSE) {
        msg += ' No printer response\n';
    }
    if (asb & epos.ASB_PRINT_SUCCESS) {
        msg += ' Print complete\n';
    }
    if (asb & epos.ASB_DRAWER_KICK) {
        msg += ' Status of the drawer kick number 3 connector pin = "H"\n';
    }
    if (asb & epos.ASB_OFF_LINE) {
        msg += ' Offline status\n';
    }
    if (asb & epos.ASB_COVER_OPEN) {
        msg += ' Cover is open\n';
    }
    if (asb & epos.ASB_PAPER_FEED) {
        msg += ' Paper feed switch is feeding paper\n';
    }
    if (asb & epos.ASB_WAIT_ON_LINE) {
        msg += ' Waiting for online recovery\n';
    }
    if (asb & epos.ASB_PANEL_SWITCH) {
        msg += ' Panel switch is ON\n';
    }
    if (asb & epos.ASB_MECHANICAL_ERR) {
        msg += ' Mechanical error generated\n';
    }
    if (asb & epos.ASB_AUTOCUTTER_ERR) {
        msg += ' Auto cutter error generated\n';
    }
    if (asb & epos.ASB_UNRECOVER_ERR) {
        msg += ' Unrecoverable error generated\n';
    }
    if (asb & epos.ASB_AUTORECOVER_ERR) {
        msg += ' Auto recovery error generated\n';
    }
    if (asb & epos.ASB_RECEIPT_NEAR_END) {
        msg += ' No paper in the roll paper near end detector\n';
    }
    if (asb & epos.ASB_RECEIPT_END) {
        msg += ' No paper in the roll paper end detector\n';
    }
    if (asb & epos.ASB_BUZZER) {
        msg += ' Sounding the buzzer (limited model)\n';
    }
    if (asb & epos.ASB_SPOOLER_IS_STOPPED) {
        msg += ' Stop the spooler\n';
    }

    // Display in the dialog box
    alert(msg);
}
```
Reception of Status Event

The status event notification function is used to check the printer status without printing. Refer to the following.

```javascript
//Set the endpoint address
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer &timeout=10000';

//Create an ePOS-Print Canvas API object
var epos = new epson.CanvasPrint(address);

//Set an event callback function (cover open)
epos.oncoveropen = function () {
    alert('coveropen');
};

//Set an event callback function (paper near end)
epos.onpapernearend = function () {
    alert('papernearend');
};

//Enable status event operation
epos.open();
```
Using the Spooler Function

A printing job will be executed immediately and a response will be returned to the application after printing is complete when sending a printing request to ePOS-Print Service I/F from the application when the spooler function is disabled.

The print data job will be added to the queue and a response will be returned to the application without waiting for printing to complete when sending a printing request to ePOS-Print Service I/F from the application when the spooler function is enabled.
Even if the output printer cannot complete the printing task, ePOS-Print Service I/F does not return an error to the application. Enabling forward printing will allow for the printing task to be completed on a substitute printer, with the application acquiring the printing results afterward. Refer to the sequence diagram below.
**Programming Example**

A response containing the specified print job ID will be returned when the spooler function is enabled. ePOS-Print Service I/F will issue a print job ID if the print job ID is not specified by the application. The print job status can be acquired using the print job ID contained in the response.

```javascript
var printjobid = '';  

function sendJob() {  
  var builder = new epson.ePOSBuilder();  
  builder.addText('Hello, World!\n');  
  var request = builder.toString();  
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';  
  var epos = new epson.ePOSPrint(address);  
  epos.onreceive = function (res) {  
    if (res.success) {  
      printjobid = res.printjobid;  
    }  
  }  
  epos.onerror = function (err) { alert(err.status); };  
  epos.send(request);  
}

function getJobStatus() {  
  if (printjobid.length > 0) {  
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi';  
    var epos = new epson.ePOSPrint(address);  
    epos.onreceive = function (res) { alert(res.success); };  
    epos.onerror = function (err) { alert(err.status); };  
    epos.getPrintJobStatus(printjobid);  
  }
}
```
This chapter describes the ePOS-Print API.

**List of API functions**

ePOS-Print provides the following objects:
- ePOS-Print Builder (window.epson.ePOSBuilder) Object  (p. 61)
- ePOS-Print (window.epson.ePOSPrint) Object  (p. 65)

### window.epson.ePOSBuilder Components

<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Standard mode</th>
<th>Page mode</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constructor</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ePOS Builder</td>
<td>Initializes an ePOS-Print XML Builder object</td>
<td>●</td>
<td>●</td>
<td>66</td>
</tr>
<tr>
<td><strong>Method</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>addTextAlign</td>
<td>Adds a tag for the text alignment setting.</td>
<td>●</td>
<td>-</td>
<td>67</td>
</tr>
<tr>
<td></td>
<td>addTextLineSpace</td>
<td>Adds a tag for the line feed space setting.</td>
<td>●</td>
<td>●</td>
<td>68</td>
</tr>
<tr>
<td></td>
<td>addTextRotate</td>
<td>Adds a tag for the text rotation setting.</td>
<td>●</td>
<td>-</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td>addText</td>
<td>Adds a tag for printing text.</td>
<td>●</td>
<td>●</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>addTextLang</td>
<td>Adds a tag for the target language setting.</td>
<td>●</td>
<td>●</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td>addTextFont</td>
<td>Adds a tag for the text font setting.</td>
<td>●</td>
<td>●</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>addTextSmooth</td>
<td>Adds a tag for the text smoothing setting.</td>
<td>●</td>
<td>●</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>addTextDouble</td>
<td>Adds a tag for specifying the double-sized text setting.</td>
<td>●</td>
<td>●</td>
<td>76</td>
</tr>
<tr>
<td></td>
<td>addTextSize</td>
<td>Adds a tag for the text scale setting.</td>
<td>●</td>
<td>●</td>
<td>78</td>
</tr>
<tr>
<td></td>
<td>addTextStyle</td>
<td>Adds a tag for the text style setting.</td>
<td>●</td>
<td>●</td>
<td>79</td>
</tr>
<tr>
<td></td>
<td>addTextPosition</td>
<td>Adds a tag for specifying the print position of text.</td>
<td>●</td>
<td>●</td>
<td>81</td>
</tr>
<tr>
<td></td>
<td>addTextVPosition</td>
<td>Adds a tag for specifying the print vertical position of text.</td>
<td>-</td>
<td>●</td>
<td>82</td>
</tr>
</tbody>
</table>

---

*Note: Elements marked with *3 are specific to certain versions or contexts.*
<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Standard mode</th>
<th>page mode</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Feed</td>
<td>addFeedUnit</td>
<td>Adds a tag for paper feeding (in dots).</td>
<td>●</td>
<td>●</td>
<td>83</td>
</tr>
<tr>
<td></td>
<td>addFeedLine</td>
<td>Adds a tag for paper feeding (in lines).</td>
<td>●</td>
<td>●</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>addFeedPosition</td>
<td>Adds control of label paper/black mark paper to command buffer</td>
<td>●</td>
<td>-</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td>addFeed</td>
<td>Adds a line feed to the command buffer.</td>
<td>●</td>
<td>-</td>
<td>87</td>
</tr>
<tr>
<td>Graphic</td>
<td>addImage</td>
<td>Adds a tag for a raster image to be printed.</td>
<td>●</td>
<td>●</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>addLogo</td>
<td>Adds a tag for an NV logo to be printed.</td>
<td>●</td>
<td>●</td>
<td>90</td>
</tr>
<tr>
<td>Barcode</td>
<td>addBarcode</td>
<td>Adds a tag for a bar code to be printed.</td>
<td>●</td>
<td>●</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>addSymbol</td>
<td>Adds a tag for a two-dimensional code to be printed.</td>
<td>●</td>
<td>●</td>
<td>96</td>
</tr>
<tr>
<td>Ruled line</td>
<td>addHLine</td>
<td>Adds a tag for a horizontal line to be printed.</td>
<td>●</td>
<td>-</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>addVLineBegin</td>
<td>Adds a tag for starting a vertical line.</td>
<td>●</td>
<td>-</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td>addVLineEnd</td>
<td>Adds a tag for finishing a vertical line.</td>
<td>●</td>
<td>-</td>
<td>105</td>
</tr>
<tr>
<td>Page-mode</td>
<td>addPageBegin</td>
<td>Adds a tag for switching to page mode.</td>
<td>●</td>
<td>-</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>addPageEnd</td>
<td>Adds a tag for finishing page mode.</td>
<td>●</td>
<td>-</td>
<td>107</td>
</tr>
<tr>
<td></td>
<td>addPageArea</td>
<td>Adds a tag for specifying the print area in page mode.</td>
<td>-</td>
<td>●</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td>addPageDirection</td>
<td>Adds a tag for specifying the print direction in page mode.</td>
<td>-</td>
<td>●</td>
<td>110</td>
</tr>
<tr>
<td></td>
<td>addPagePosition</td>
<td>Adds a tag for specifying the print position in page mode.</td>
<td>-</td>
<td>●</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td>addPageLine</td>
<td>Adds a tag for drawing a line in page mode.</td>
<td>-</td>
<td>●</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>addPageRectangle</td>
<td>Adds a tag for drawing a rectangle in page mode.</td>
<td>-</td>
<td>●</td>
<td>116</td>
</tr>
<tr>
<td>Cut</td>
<td>addCut</td>
<td>Adds a tag for paper cut.</td>
<td>-</td>
<td>●</td>
<td>118</td>
</tr>
<tr>
<td>Drawer kick-out</td>
<td>addPulse</td>
<td>Adds a tag for the drawer kick-out.</td>
<td>●</td>
<td>-</td>
<td>119</td>
</tr>
<tr>
<td>Buzzer</td>
<td>addSound</td>
<td>Adds a tag for turning on the buzzer.</td>
<td>●</td>
<td>-</td>
<td>121</td>
</tr>
<tr>
<td>Layout</td>
<td>addLayout *2</td>
<td>Adds the paper layout setup to command buffer.</td>
<td>●</td>
<td>-</td>
<td>123</td>
</tr>
<tr>
<td>Recovery</td>
<td>addRecovery *3</td>
<td>Adds a tag for recovering from an error.</td>
<td>●</td>
<td>-</td>
<td>127</td>
</tr>
<tr>
<td>Reset</td>
<td>addReset *3</td>
<td>Adds a tag for resetting the printer.</td>
<td>●</td>
<td>-</td>
<td>128</td>
</tr>
</tbody>
</table>
### Chapter 4  ePOS-Print API

#### Method

<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Standard mode</th>
<th>page mode</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Command</td>
<td>addCommand</td>
<td>Adds commands to the command buffer. Sends ESC/POS commands.</td>
<td>●</td>
<td>●</td>
<td>129</td>
</tr>
<tr>
<td>Create a Print Document</td>
<td>toString</td>
<td>Obtains a print document generated by an ePOS-Print Builder object.</td>
<td>●</td>
<td>-</td>
<td>130</td>
</tr>
</tbody>
</table>

●: Available, -: Not available
*1 ePOS-Print Service Ver.2.1 or later versions supported
*2 ePOS-Print Service Ver.2.2 or later versions supported
*3 ePOS-Print Service Ver.3.0 or later versions supported

#### Property

<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>halftone</td>
<td>FONT_*</td>
<td>Raster image halftone processing method</td>
<td>131</td>
</tr>
<tr>
<td>brightness</td>
<td>COLOR_*</td>
<td>Raster image brightness correction value</td>
<td>132</td>
</tr>
<tr>
<td>force</td>
<td>SYMBOL_*</td>
<td>Forced transmission mode</td>
<td>133</td>
</tr>
<tr>
<td>message</td>
<td>LEVEL_*</td>
<td>Message buffer</td>
<td>134</td>
</tr>
</tbody>
</table>

#### Constant

<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALIGN_*</td>
<td>FONT_*</td>
<td>font</td>
<td></td>
</tr>
<tr>
<td>COLOR_*</td>
<td>ALIGN_*</td>
<td>alignment</td>
<td></td>
</tr>
<tr>
<td>HALFTONE_*</td>
<td>COLOR_*</td>
<td>color specification</td>
<td></td>
</tr>
<tr>
<td>MODE_*</td>
<td>BARCODE_*</td>
<td>Halftone type</td>
<td></td>
</tr>
<tr>
<td>HRI_*</td>
<td>BARCODE_*</td>
<td>Color mode</td>
<td></td>
</tr>
<tr>
<td>SYMBOL_*</td>
<td>HRI_*</td>
<td>bar code type</td>
<td></td>
</tr>
<tr>
<td>LEVEL_*</td>
<td>SYMBOL_*</td>
<td>two-dimensional code type</td>
<td></td>
</tr>
<tr>
<td>LINE_*</td>
<td>DIRECTION_*</td>
<td>error correction level</td>
<td></td>
</tr>
<tr>
<td>DIRECTION_*</td>
<td>LINE_*</td>
<td>line style</td>
<td></td>
</tr>
<tr>
<td>CUT_*</td>
<td>DIRECTION_*</td>
<td>page mode print direction</td>
<td></td>
</tr>
<tr>
<td>PULSE_*</td>
<td>CUT_*</td>
<td>paper cut type</td>
<td></td>
</tr>
<tr>
<td>PATTERN_*</td>
<td>DIRECTION_*</td>
<td>drawer kick-out connector</td>
<td></td>
</tr>
<tr>
<td>FEED_*</td>
<td>PATTERN_*</td>
<td>buzzer sound pattern</td>
<td></td>
</tr>
<tr>
<td>LAYOUT_*</td>
<td>FEED_*</td>
<td>Paper feed position of label paper/black mark paper</td>
<td></td>
</tr>
<tr>
<td>LAYOUT_*</td>
<td>LAYOUT_*</td>
<td>Type of papers</td>
<td></td>
</tr>
</tbody>
</table>

*1 ePOS-Print Service Ver.2.1 or later versions supported
*2 ePOS-Print Service Ver.2.2 or later versions supported
*3 ePOS-Print Service Ver.3.0 or later versions supported
Numerical values to be set to parameters

In the ePOS-Print Builder object API, numerical values are set to some parameters. Set values with the following in mind:

- **Unit**
  - Specify numbers in dots for units that represent length.
  - (Print position, paper feed space, width and height of images and barcodes, etc.)

- **Range**
  - Depending on the printer specifications, a specifiable range is predetermined. For details, refer to Printer specifications (p.203).

- **Resolution**
  - The resolution varies depending on the printer. It affects the actual print size. The higher the resolution is, the smaller the print size becomes, and vice versa. For each printer's resolution, refer to Printer specifications (p.203).
## window.epson.ePOSPrint Components

<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ePOS-Print</td>
<td></td>
<td>Initializes an ePOS-Print object</td>
<td>135</td>
</tr>
<tr>
<td>Method</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>send</td>
<td></td>
<td>Sends a message</td>
<td>136</td>
</tr>
<tr>
<td>getPrintJobStatus</td>
<td></td>
<td>Acquires print job status.</td>
<td>138</td>
</tr>
<tr>
<td>open</td>
<td></td>
<td>Enables status event operation</td>
<td>139</td>
</tr>
<tr>
<td>close</td>
<td></td>
<td>Disables status event operation</td>
<td>140</td>
</tr>
<tr>
<td>Property</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>address</td>
<td></td>
<td>URL of the printer</td>
<td>141</td>
</tr>
<tr>
<td>enabled</td>
<td></td>
<td>Enabling/disabling of status event</td>
<td>142</td>
</tr>
<tr>
<td>interval</td>
<td></td>
<td>Printer status update interval</td>
<td>142</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Status</td>
<td>143</td>
</tr>
<tr>
<td>battery</td>
<td></td>
<td>Battery status</td>
<td>143</td>
</tr>
<tr>
<td>timeout</td>
<td></td>
<td>The connecting was timeout.</td>
<td>144</td>
</tr>
<tr>
<td>Event</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onreceive</td>
<td></td>
<td>Response message receipt event</td>
<td>145</td>
</tr>
<tr>
<td>onerror</td>
<td></td>
<td>Communication error event</td>
<td>148</td>
</tr>
<tr>
<td>onstatuschange</td>
<td></td>
<td>Status change event</td>
<td>149</td>
</tr>
<tr>
<td>onbatterystatuschange</td>
<td></td>
<td>Battery status change event</td>
<td>149</td>
</tr>
<tr>
<td>ononline</td>
<td></td>
<td>Online event</td>
<td>150</td>
</tr>
<tr>
<td>onoffline</td>
<td></td>
<td>Offline event</td>
<td>150</td>
</tr>
<tr>
<td>onpoweroff</td>
<td></td>
<td>Non-response event</td>
<td>151</td>
</tr>
<tr>
<td>oncoverok</td>
<td></td>
<td>Cover close event</td>
<td>151</td>
</tr>
<tr>
<td>oncoveropen</td>
<td></td>
<td>Cover open event</td>
<td>152</td>
</tr>
<tr>
<td>onpaperox</td>
<td></td>
<td>Paper remaining event</td>
<td>152</td>
</tr>
<tr>
<td>onpapernarend</td>
<td></td>
<td>Paper near end event</td>
<td>153</td>
</tr>
<tr>
<td>onpaperend</td>
<td></td>
<td>Paper end event</td>
<td>153</td>
</tr>
<tr>
<td>Event</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ondrawerclosed</td>
<td></td>
<td>Drawer close event</td>
<td>154</td>
</tr>
<tr>
<td>ondraweropen</td>
<td></td>
<td>Drawer open event</td>
<td>154</td>
</tr>
<tr>
<td>onbatteryok</td>
<td></td>
<td>Battery OK event</td>
<td>155</td>
</tr>
<tr>
<td>onbatterylow</td>
<td></td>
<td>Battery low event</td>
<td>155</td>
</tr>
<tr>
<td>Constant</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASB_*</td>
<td></td>
<td>Status</td>
<td></td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported
ePOS-Print Builder Object

This object creates a print document for printer control commands that specify strings or graphics to be printed, paper cut, etc.

Constructor

Constructor for an ePOS-Print Builder object.
Creates a new ePOS-Print Builder object and initializes it.

Syntax

```
ePOSBuilder();
```

Example

```
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
!---
function buildMessage() {
    var builder = new epson.ePOSBuilder();
}
//-->
</script>
```
addTextAlign method

Adds the text alignment setting to the command buffer.

- This API setting is applied to raster image/NV logo/barcode/two-dimensional symbol.
- When using the standard mode, specify addTextAlign in "Position at the beginning of lines".
- In the page mode, addTextAlign method specification cannot be used.
  In the page mode, the addTextPosition method to designate the horizontal print position.
- When the page mode is selected for the print mode, to set text rotation, use the
  addPageDirection method (p. 110) instead of this API function.

**Syntax**

```javascript
addTextAlign(align);
```

**Parameter**

- **align**: (Required parameter, Object type: String)
  Specifies the text alignment.

<table>
<thead>
<tr>
<th>Constant(align)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALIGN_LEFT (default)</td>
<td>Alignment to the left</td>
</tr>
<tr>
<td>ALIGN_CENTER</td>
<td>Alignment to the center</td>
</tr>
<tr>
<td>ALIGN_RIGHT</td>
<td>Alignment to the right</td>
</tr>
</tbody>
</table>

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot;, is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

To set alignment to the center:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addTextAlign(builder.ALIGN_CENTER);
}
//-->
</script>
```
addTextLineSpace method

Adds the line feed space setting to the command buffer.

Syntax

```javascript
addTextLineSpace(linespc);
```

Parameter

- linespc :  (Required parameter, Object type : Number)
  Specifies the line feed space (in dots). Specifies an integer from 0 to 255.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To set the line feed space to 30 dots:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
 <!--
   function buildMessage() {
     var builder = new epson.ePOSBuilder();
     builder.addTextLineSpace(30);
   }
   //-->
</script>
```
### addTextRotate method

Adds the text rotation setting to the command buffer.

- This API setting also applies to barcodes/two dimensional symbols.
- When using the standard mode, specify addTextAlign in "Position at the beginning of lines".
- In the page mode, addTextAlign method specification cannot be used.
- When the page mode is selected for the print mode, to set text rotation, use the addPageDirection method (p.110) instead of this API function.

#### Syntax

```javascript
addTextRotate(rotate);
```

#### Parameter

- `rotate` (Required parameter, Object type : Boolean)
  
  Specifies whether to rotate text.

#### Return value

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies rotated printing of text.</td>
</tr>
<tr>
<td>false or 0</td>
<td>Cancels rotated printing of text.</td>
</tr>
</tbody>
</table>

#### Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

#### Example

To set text rotation:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addTextRotate(true);
  }
</script>
```
addText method

Adds the printing of text to the command buffer.

- After printing text, to print content other than text, execute line feed or paper feed.
- In page mode, characters are laid out in the current print position with the reference point being the character baseline dot (Printer specifications (p.203)).

Syntax

```
addText(data);
```

Parameter

- data: (Required parameter, Object type: String)
  Specify a character string to be printed.
  For the horizontal tab/line feed, use the following escape sequences:

<table>
<thead>
<tr>
<th>String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\t</td>
<td>Horizontal tab(HT)</td>
</tr>
<tr>
<td>\n</td>
<td>Line feed (LF)</td>
</tr>
<tr>
<td>\</td>
<td>Carriage return</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To add character strings:

```
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addText('Hello,\t').addText('World\n');
  }
  //-->
</script>
```
addTextLang method

Adds the language setting to the command buffer.

**Syntax**

```
addTextLang(lang);
```

**Parameter**

- **lang** : (Required parameter, Object type : String)
  Specifies the target language.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>en(default)</td>
<td>English (ANK)</td>
</tr>
<tr>
<td>de</td>
<td>German (ANK)</td>
</tr>
<tr>
<td>fr</td>
<td>French (ANK)</td>
</tr>
<tr>
<td>it</td>
<td>Italian (ANK)</td>
</tr>
<tr>
<td>es</td>
<td>Spanish (ANK)</td>
</tr>
<tr>
<td>ja</td>
<td>Japanese (International character set changes to Japan.)</td>
</tr>
<tr>
<td>ja-ja</td>
<td>Japanese (International character set changes to Japan.)</td>
</tr>
<tr>
<td>ko</td>
<td>Korean (International character set changes to Korean.)</td>
</tr>
<tr>
<td>ko-kr</td>
<td>Korean (International character set changes to Korean.)</td>
</tr>
<tr>
<td>zh-hans</td>
<td>Simplified Chinese (International character set changes to China.)</td>
</tr>
<tr>
<td>zh-cn</td>
<td>Simplified Chinese (International character set changes to China.)</td>
</tr>
<tr>
<td>zh-hant</td>
<td>Traditional Chinese</td>
</tr>
<tr>
<td>zh-tw</td>
<td>Traditional Chinese</td>
</tr>
</tbody>
</table>

Language code besides above

- **en(ANK)**
- **fr(ANK)**
- **it(ANK)**
- **es(ANK)**
- **ja(ANK)**
- **ja-ja(ANK)**
- **ko(ANK)**
- **ko-kr(ANK)**
- **zh-hans(ANK)**
- **zh-cn(ANK)**
- **zh-hant(ANK)**
- **zh-tw(ANK)**

* ePOS-Print Service Ver.2.2 or later versions supported

- Characters not installed in a printer cannot be printed.

For printable character code, refer to the Technical Reference Guide of your printer.
Depending on language specification, a part of characters is printed as follows.

<table>
<thead>
<tr>
<th>Language</th>
<th>Characters $(U+0024)$</th>
<th>Characters (U+005C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Japanese</td>
<td>¥</td>
<td>¥</td>
</tr>
<tr>
<td>Korean</td>
<td>$</td>
<td>\</td>
</tr>
<tr>
<td>Simplified Chinese</td>
<td>¥</td>
<td>\</td>
</tr>
<tr>
<td>Traditional Chinese</td>
<td>$</td>
<td>\</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To set the language as English:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addTextLang('en');
}
//-->
</script>
```

To set the language as Korean:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addTextLang('ko');
}
//-->
</script>
```
To set the language as Simplified Chinese:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
 <!--
 function buildMessage() {
   var builder = new epson.ePOSBuilder();
   builder.addTextLang('zh-hans');
 }
 //--> 
</script>
```

To set the language as Traditional Chinese:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
 <!--
 function buildMessage() {
   var builder = new epson.ePOSBuilder();
   builder.addTextLang('zh-hant');
 }
 //--> 
</script>
```
addTextFont method

Adds the text font setting to the command buffer.

**Syntax**

```javascript
addTextFont(font);
```

**Parameter**

- `font`: (Required parameter, Object type: String)
  Specifies the font.

<table>
<thead>
<tr>
<th>Constant (font)</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>FONT_A (default)</td>
<td>Font A</td>
</tr>
<tr>
<td>FONT_B</td>
<td>Font B</td>
</tr>
<tr>
<td>FONT_C</td>
<td>Font C</td>
</tr>
<tr>
<td>FONT_D *</td>
<td>Font D</td>
</tr>
<tr>
<td>FONT_E *</td>
<td>Font E</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.3.2 or later versions supported

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;... &quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

To set the font B:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//!--
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addTextFont(builder.FONT_B);
}
//-->
</script>
```
**addTextSmooth method**

Adds the smoothing setting to the command buffer.

**Syntax**

```
addTextSmooth(smooth);
```

**Parameter**

- `smooth`: (Required parameter, Object type: Boolean)
  Specifies whether to enable smoothing.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies smoothing.</td>
</tr>
<tr>
<td>false or 0 (default)</td>
<td>Cancels smoothing</td>
</tr>
</tbody>
</table>

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

To enable smoothing:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addTextSmooth(true);
  }
</script>
```
addTextDouble method

Adds the double-sized text setting to the command buffer.

**Syntax**

```javascript
addTextDouble(dw, dh);
```

**Parameter**

- **dw**: (Optional parameter, Object type: Boolean)
  Specifies the double-sized width.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies the double-sized width.</td>
</tr>
<tr>
<td>false or 0 (default)</td>
<td>Cancels the double-sized width</td>
</tr>
<tr>
<td>undefined</td>
<td>Retains the current setting for double-sized width.</td>
</tr>
</tbody>
</table>

- **dh**: (Optional parameter, Object type: Boolean)
  Specifies the double-sized height.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies the double-sized height</td>
</tr>
<tr>
<td>false or 0 (default)</td>
<td>Cancels the double-sized height</td>
</tr>
<tr>
<td>undefined</td>
<td>Retains the current setting for double-sized height</td>
</tr>
</tbody>
</table>

When true or 1 is set for both the dw and dh parameters, double width and height characters are printed.

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To set the size as double width and height:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addTextDouble(true, true);
}
//-->
</script>
```
addTextSize method

Adds the text scale setting to the command buffer.

Syntax

```
addTextSize(width, height);
```

Parameter

- width: (Optional parameter, Object type: Number)
  Specifies the horizontal scale of text.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer from 1 to 8</td>
<td>Horizontal scale (default: 1)</td>
</tr>
<tr>
<td>undefined</td>
<td>Retains the current setting for the horizontal scale.</td>
</tr>
<tr>
<td>(When not specified)</td>
<td></td>
</tr>
</tbody>
</table>

- height: (Optional parameter, Object type: Number)
  Specifies the vertical scale of text.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer from 1 to 8</td>
<td>Vertical scale (default: 1)</td>
</tr>
<tr>
<td>undefined</td>
<td>Retains the current setting for the vertical scale.</td>
</tr>
<tr>
<td>(When not specified)</td>
<td></td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot; ... &quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To set a horizontal scale of x 4 and a vertical scale of x 4:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    <!--
    function buildMessage() {
        var builder = new epson.ePOSBuilder();
        builder.addTextSize(4, 4);
    }
    //-->
</script>
```
addTextStyle method

Adds the text style setting to the command buffer.

**Syntax**

```plaintext
addTextStyle(reverse, ul, em, color);
```

**Parameter**

- **reverse**: (Optional parameter, Object type : Boolean)
  Specifies inversion of black and white for text.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies the inversion of black and white parts of characters.</td>
</tr>
<tr>
<td>false or 0 (default)</td>
<td>Cancels the inversion of black and white parts of characters.</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Retains the current setting for inversion of black and white.</td>
</tr>
</tbody>
</table>

- **ul**: (Optional parameter, Object type : Boolean)
  Specifies the underline style.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies underlining.</td>
</tr>
<tr>
<td>false or 0 (default)</td>
<td>Cancels underlining.</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Retains the current underlining setting.</td>
</tr>
</tbody>
</table>

- **em**: (Optional parameter, Object type : Boolean)
  Specifies the bold style.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Specifies emphasized printing of characters.</td>
</tr>
<tr>
<td>false or 0 (default)</td>
<td>Cancels emphasized printing of characters.</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Retains the current setting for emphasized printing.</td>
</tr>
</tbody>
</table>

- **color**: (Optional parameter, Object type : String)
  Specifies the color.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLOR_NONE</td>
<td>Characters are not printed.</td>
</tr>
<tr>
<td>COLOR_1 (default)</td>
<td>First color</td>
</tr>
<tr>
<td>COLOR_2</td>
<td>Second color</td>
</tr>
<tr>
<td>COLOR_3</td>
<td>Third color</td>
</tr>
<tr>
<td>COLOR_4</td>
<td>Fourth color</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Retains the current color setting</td>
</tr>
</tbody>
</table>
### Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

### Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot; ... &quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

### Example

To set the underline style:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//-->
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addTextStyle(undefined, true);
}
//-->
</script>
```
addTextPosition method

Adds the horizontal print start position of text to the command buffer.

**Syntax**

```
addTextPosition(x);
```

**Parameter**

- `x`: (Required parameter, Object type: Number)
  Specifies the horizontal print start position (in dots).
  Specifies an integer from 0 to 65535.

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

To set the print position at 120 dots from the left end:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addTextPosition(120);
}
//-->
</script>
```
addTextVPosition method

Adds the vertical print start position of text to the command buffer. (in ePOS-Print Service Ver.3.0 and later)

Use this API function by inserting it between addPageBegin to addPageEnd.

Syntax

    addTextVPosition(y);

Parameter

- y :  (Required parameter, Object type : Number)
  Specifies the vertical print start position (in dots).
  Specifies an integer from 0 to 65535.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To set the print position at 120 dots from the top:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addPageBegin();
    builder.addPageBegin();
    builder.addTextVPosition(120);
    builder.addPageEnd();
  }
  //-->
</script>
```
addFeedUnit method

Adds paper feeding in dots to the command buffer.

Syntax

```
addFeedUnit(unit);
```

Parameter

- `unit`: (Required parameter, Object type: Number)
  Specifies the paper feed space (in dots). Specifies an integer from 0 to 255.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To feed paper by 30 dots:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addFeedUnit(30);
  }
  //-->
  </script>
```
addFeedLine method

Adds paper feeding in lines to the command buffer.

Syntax

```
addFeedLine(line);
```

Parameter

- `line` : (Required parameter, Object type: Number)
  Specifies the paper feed space (in lines). Specifies an integer from 0 to 255.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To feed paper by 3 lines:

```
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addFeedLine(3);
}
//-->
</script>
```
addFeedPosition method

Adds label/black mark paper feeding to the command buffer.

- Control of label paper/black mark paper must be done in the standard mode.
- In the page mode, addFeedPosition method specification cannot be used.

**Syntax**

```addFeedPosition(pos);```

**Parameter**

- `pos` : (Required parameter, Object type : String)
  Specifies the feed position.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEED_PEELING</td>
<td>Feeds to the peeling position.</td>
</tr>
<tr>
<td>FEED_CUTTING</td>
<td>Feeds to the cutting position.</td>
</tr>
<tr>
<td>FEED_CURRENT_TOF</td>
<td>Feeds to the top of the current label.</td>
</tr>
<tr>
<td>FEED_NEXT_TOF</td>
<td>Feeds to the top of the next label.</td>
</tr>
</tbody>
</table>

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot; ... &quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To print while peeling the label one by one

```javascript
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addFeedPosition(builder.FEED_CURRENT_TOF);
    builder.addBarcode('0001', builder.BARCODE_CODE39, builder.HRI_BELOW);
    builder.addFeedPosition(builder.FEED_PEELING);
}
```

To print labels consecutively

```javascript
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addFeedPosition(builder.FEED_CURRENT_TOF);
    builder.addBarcode('0001', builder.BARCODE_CODE39, builder.HRI_BELOW);
    builder.addFeedPosition(builder.NEXT_TOF);
    builder.addBarcode('0002', builder.BARCODE_CODE39, builder.HRI_BELOW);
    builder.addFeedPosition(builder.NEXT_TOF);
    builder.addBarcode('0003', builder.BARCODE_CODE39, builder.HRI_BELOW);
    builder.addFeedPosition(builder.NEXT_TOF);
}
```

To print tickets with black mark paper

```javascript
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addFeedPosition(builder.FEED_CURRENT_TOF);
    builder.addBarcode('0001', builder.BARCODE_CODE39, builder.HRI_BELOW);
    builder.addFeedPosition(builder.FEED_CUTTING);
    builder.addCut(builder.CUT_NO_FEED);
}
```
addFeed method

Adds a line feed to the command buffer.

**Syntax**

```javascript
addFeed();
```

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Example**

To start a new line after printing a character string:

```javascript
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addText("Hello").addFeed();
    builder.addText("World").addFeed();
}
```

**addImage method**

Adds raster image printing to the command buffer. Prints graphics rendered in HTML5 Canvas. Converts the specified range in a RGBA full-color image of HTML5 Canvas into raster image data according to the settings of the halftone and brightness properties. One pixel in an image equals to one printer dot. When an image contains any transparent color, the background color of the image is assumed to be white.

If an HTML5 Canvas image contains images downloaded from different domains, you cannot print the image. In this case, a security error occurs due to violation of the same origin policy of JavaScript.

- To print a raster image at high speed, specify ALIGN_LEFT for the addTextAlign method (p. 67), and specify a multiple of 8 not exceeding the printer’s paper width for the width parameter of this API.
- In page mode, a raster image is laid out in the current print position with the reference point being its bottom left dot. The print position will not move.
- Multiple tone printing is not supported in Page Mode. Multiple tone graphic printing is supported in Standard Mode only.

**Syntax**

```javascript
addImage(context, x, y, width, height, color, mode);
```

**Parameter**

- **context**: (Required parameter, Object type : Context)
  Specifies the 2D context of HTML5 Canvas.
- **x**: (Required parameter, Object type : Number)
  Specifies the horizontal start position in the print area. Specifies an integer from 0 to 65535.
- **y**: (Required parameter, Object type : Number)
  Specifies the vertical start position in the print area. Specifies an integer from 0 to 65535.
- **width**: (Required parameter, Object type : Number)
  Specifies the width of the print area. Specifies an integer from 0 to 65535.
- **height**: (Required parameter, Object type : Number)
  Specifies the height of the print area. Specifies an integer from 0 to 65535.
- **color**: (Optional parameter, Object type : String)
  Specifies the color.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLOR_NONE</td>
<td>Characters are not printed.</td>
</tr>
<tr>
<td>COLOR_1 (default)</td>
<td>First color</td>
</tr>
<tr>
<td>COLOR_2</td>
<td>Second color</td>
</tr>
<tr>
<td>COLOR_3</td>
<td>Third color</td>
</tr>
<tr>
<td>COLOR_4</td>
<td>Fourth color</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>First color</td>
</tr>
</tbody>
</table>


- **mode**: (Optional parameter, Object type: String)
  Specifies the color mode.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_MONO</td>
<td>Monochrome (two-tone)</td>
</tr>
<tr>
<td>MODE_GRAY16</td>
<td>Gray scale (16-tone)</td>
</tr>
<tr>
<td>undefined</td>
<td>(When not specified)</td>
</tr>
<tr>
<td></td>
<td>Monochrome (two-tone)</td>
</tr>
</tbody>
</table>

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    var canvas = document.getElementById('canvas');
    if (canvas.getContext) {
      var context = canvas.getContext('2d');
      builder.addImage(context, 0, 0, canvas.width, canvas.height);
    }
  }
  //-->
</script>
```

To print an image 300 dots wide and 300 dots high in page mode:

```javascript
var canvas = document.getElementById('myCanvas');
var context = canvas.getContext('2d');
var builder = new epson.ePOSBuilder();
builder.addPageBegin();
builder.addPageArea(0, 0, 300, 300);
builder.addPagePosition(0, 299);
builder.addImage(context, 0, 0, canvas.width, canvas.height);
builder.addPageEnd();
```
**addLogo method**

Adds NV logo printing to the command buffer.
Prints a logo registered in the NV memory of the printer.

- Using model-dedicated utility or logo registration utility (TMFLogo), register a logo in the printer in advance.
- In page mode, a logo is laid out in the current print position with the reference point being its bottom left dot.
- Multiple tone printing is not supported in Page Mode. Multiple tone graphic printing is supported in Standard Mode only.

**Syntax**

```
addLogo(key1, key2);
```

**Parameter**

- **key1**: (Required parameter, Object type : Number)
  Specifies the key code 1 of an NV logo. Specifies an integer from 0 to 255.

- **key2**: (Required parameter, Object type : Number)
  Specifies the key code 2 of an NV logo. Specifies an integer from 0 to 255.

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    <!--
    function buildMessage() {
        var builder = new epson.ePOSBuilder();
        builder.addLogo(48, 48);
    }
    //-->
</script>
addBarcode method

Adds barcode printing to the command buffer.

In page mode, a barcode is laid out in the current print position with the reference point being its bottom left dot (except for HRI).

Syntax

```
addBarcode(data, type, hri, font, width, height);
```

Parameter

- `data` : (Required parameter, Object type : String)
  Specifies the barcode data as a string.

<table>
<thead>
<tr>
<th>Barcode type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPC-A</td>
<td>When an 11-digit number is specified, a check digit is automatically added. When a 12-digit number is specified, the 12th digit is processed as a check digit but the check digit is not validated.</td>
</tr>
<tr>
<td>UPC-E</td>
<td>Specify 0 as the first digit. Specify the manufacturer code in the digits 2 to 6. Specify (right-align) the item code in the digits 7 to 11. The number of item code digits varies depending on the manufacturer code. Specify 0s in empty digits. When an 11-digit number is specified, a check digit is automatically added. When a 12-digit number is specified, the 12th digit is processed as a check digit but the check digit is not validated.</td>
</tr>
<tr>
<td>EAN13</td>
<td>When an 12-digit number is specified, a check digit is automatically added.</td>
</tr>
<tr>
<td>JAN13</td>
<td>When a 13-digit number is specified, the 12th digit is processed as a check digit but the check digit is not validated.</td>
</tr>
<tr>
<td>EAN8</td>
<td>When a 7-digit number is specified, a check digit is automatically added.</td>
</tr>
<tr>
<td>JAN8</td>
<td>When an 8-digit number is specified, the 8th digit is processed as a check digit but the check digit is not validated.</td>
</tr>
<tr>
<td>CODE39</td>
<td>When the first character is *, the character is processed as the start character. In other cases, a start character is automatically added.</td>
</tr>
<tr>
<td>ITF</td>
<td>Start and stop codes are automatically added. Check digits are not added or validated.</td>
</tr>
<tr>
<td>CODABAR</td>
<td>Specify a start character (A to D, a to d). Specify a stop character (A to D, a to d). Check digits are not added or validated.</td>
</tr>
</tbody>
</table>
### How to specify Code128 special characters

A check digit is automatically calculated and added.

To encode each of the following characters, specify two characters starting with the character "\{":

<table>
<thead>
<tr>
<th>Barcode type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE93</td>
<td>Start and stop characters are automatically added. A check digit is automatically calculated and added.</td>
</tr>
<tr>
<td>CODE128</td>
<td>Specify a start character (CODE A, CODE B, CODE C). A stop character is automatically added. A check digit is automatically calculated and added. To encode each of the following characters, specify two characters starting with the character &quot;{&quot;:&lt; How to specify special characters &gt; (^*1) &lt; How to specify CODE C &gt; (^*2)</td>
</tr>
<tr>
<td>GS1-128</td>
<td>A start character, FNC1, a check digit, and a stop character are automatically added. To automatically calculate and add a check digit for an application identifier (AI) and the subsequent data, specify the character &quot;*&quot; in the position of the check digit. You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data. You can insert spaces between an application identifier (AI) and data. The spaces are used as HRI print characters and are not encoded as data. To encode each of the following characters, specify two characters starting with the character &quot;{&quot;:&lt; How to specify special characters &gt; (^*3) &lt; How to specify CODE C &gt; (^*2)</td>
</tr>
<tr>
<td>GS1 DataBar Omnidirectional</td>
<td>Specify a 13-digit global trade item number (GTIN) not including an application identifier (AI) or a check digit.</td>
</tr>
<tr>
<td>GS1 DataBar Truncated</td>
<td>You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data.</td>
</tr>
<tr>
<td>GS1 DataBar Limited</td>
<td>You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data.</td>
</tr>
<tr>
<td>GS1 Databar Expanded</td>
<td>You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data.</td>
</tr>
</tbody>
</table>

\(^*1\): How to specify Code128 special characters

<table>
<thead>
<tr>
<th>Data</th>
<th>Specified character string</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNC1</td>
<td>(\text{C})</td>
</tr>
<tr>
<td>FNC2</td>
<td>(\text{2})</td>
</tr>
<tr>
<td>FNC3</td>
<td>(\text{3})</td>
</tr>
<tr>
<td>FNC4</td>
<td>(\text{4})</td>
</tr>
<tr>
<td>CODE A</td>
<td>(\text{A})</td>
</tr>
<tr>
<td>CODE B</td>
<td>(\text{B})</td>
</tr>
<tr>
<td>CODE C</td>
<td>(\text{1})</td>
</tr>
<tr>
<td>SHIFT</td>
<td>(\text{S})</td>
</tr>
<tr>
<td>{</td>
<td>(\text{{}})</td>
</tr>
</tbody>
</table>
**2: How to specify Code128 CODE C and GS1-128 CODE C.**

<table>
<thead>
<tr>
<th>Data</th>
<th>Specified character string</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>\x00</td>
</tr>
<tr>
<td>01</td>
<td>\x01</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>09</td>
<td>\x09</td>
</tr>
<tr>
<td>10</td>
<td>\x0a</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>98</td>
<td>\x62 or b</td>
</tr>
<tr>
<td>99</td>
<td>\x63 or c</td>
</tr>
</tbody>
</table>

**3: How to specify GS1-128 special characters**

<table>
<thead>
<tr>
<th>Data</th>
<th>Specified character string</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNC1</td>
<td>{1</td>
</tr>
<tr>
<td>FNC3</td>
<td>{3</td>
</tr>
<tr>
<td>(</td>
<td>(</td>
</tr>
<tr>
<td>)</td>
<td>)</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>{</td>
<td>{</td>
</tr>
</tbody>
</table>

**4: How to specify GS1 DataBar Expanded special characters**

<table>
<thead>
<tr>
<th>Data</th>
<th>Specified character string</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNC1</td>
<td>{1</td>
</tr>
<tr>
<td>(</td>
<td>(</td>
</tr>
<tr>
<td>)</td>
<td>)</td>
</tr>
</tbody>
</table>

To specify binary data that cannot be represented by character strings, use the following escape sequences.

<table>
<thead>
<tr>
<th>String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\xnn</td>
<td>Control code</td>
</tr>
<tr>
<td>\ \</td>
<td>Back slash</td>
</tr>
</tbody>
</table>
• type : 
  (Required parameter, Object type : String)
  Specifies the barcode type.

<table>
<thead>
<tr>
<th>Constant (type)</th>
<th>Barcode type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BARCODE_UPC_A</td>
<td>UPC-A</td>
</tr>
<tr>
<td>BARCODE_UPC_E</td>
<td>UPC-E</td>
</tr>
<tr>
<td>BARCODE_EAN13</td>
<td>EAN13</td>
</tr>
<tr>
<td>BARCODE_JAN13</td>
<td>JAN13</td>
</tr>
<tr>
<td>BARCODE_EAN8</td>
<td>EAN8</td>
</tr>
<tr>
<td>BARCODE_JAN8</td>
<td>JAN8</td>
</tr>
<tr>
<td>BARCODE_CODE39</td>
<td>CODE39</td>
</tr>
<tr>
<td>BARCODE_ITF</td>
<td>ITF</td>
</tr>
<tr>
<td>BARCODE_CODABAR</td>
<td>CODABAR</td>
</tr>
<tr>
<td>BARCODE_CODE93</td>
<td>CODE93</td>
</tr>
<tr>
<td>BARCODE_CODE128</td>
<td>CODE128</td>
</tr>
<tr>
<td>BARCODE_GS1_128</td>
<td>GS1-128</td>
</tr>
<tr>
<td>BARCODE_GS1_DATABAR_OMNIDIRECTIONAL</td>
<td>GS1 DataBar Omnidirectional</td>
</tr>
<tr>
<td>BARCODE_GS1_DATABAR_TRUNCATED</td>
<td>GS1 DataBar Truncated</td>
</tr>
<tr>
<td>BARCODE_GS1_DATABAR_LIMITED</td>
<td>GS1 DataBar Limited</td>
</tr>
<tr>
<td>BARCODE_GS1_DATABAR_EXPANDED</td>
<td>GS1 DataBar Expanded</td>
</tr>
</tbody>
</table>

• hri : 
  (Optional parameter, Object type : String)
  Specifies the HRI position.

<table>
<thead>
<tr>
<th>Constant (hri)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRI_NONE (default)</td>
<td>HRI not printed</td>
</tr>
<tr>
<td>HRI_ABOVE</td>
<td>Above the bar code</td>
</tr>
<tr>
<td>HRI_BELOW</td>
<td>Below the bar code</td>
</tr>
<tr>
<td>HRI_BOTH</td>
<td>Both above and below the bar code</td>
</tr>
</tbody>
</table>

• font : 
  (Optional parameter, Object type : String)
  Specifies the HRI font.

<table>
<thead>
<tr>
<th>Constant (font)</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>FONT_A (default)</td>
<td>Font A</td>
</tr>
<tr>
<td>FONT_B</td>
<td>Font B</td>
</tr>
<tr>
<td>FONT_C</td>
<td>Font C</td>
</tr>
<tr>
<td>FONT_D *</td>
<td>Font D</td>
</tr>
<tr>
<td>FONT_E *</td>
<td>Font E</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported

• width : 
  (Optional parameter, Object type : Number)
  Specifies the width of each module in dots. Specifies an integer from 2 to 6.

• height : 
  (Optional parameter, Object type : Number)
  Specifies the barcode height in dots. Specifies an integer from 1 to 255.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>
### Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

### Example

To print barcodes:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addBarcode('01234567890', builder.BARCODE_UPC_A, builder.HRI_BELOW, undefined, 2, 64);
    builder.addBarcode('01234500005', builder.BARCODE_UPC_E);
    builder.addBarcode('201234567890', builder.BARCODE_EAN13);
    builder.addBarcode('201234567890', builder.BARCODE_JAN13);
    builder.addBarcode('2012345', builder.BARCODE_EAN8);
    builder.addBarcode('2012345', builder.BARCODE_JAN8);
    builder.addBarcode('ABCDE', builder.BARCODE_CODE39);
    builder.addBarcode('012345', builder.BARCODE_ITF);
    builder.addBarcode('A012345A', builder.BARCODE_CODABAR);
    builder.addBarcode('ABCDE', builder.BARCODE_CODE93);
    builder.addBarcode('0201234567890', builder.BARCODE_GS1_128);
    builder.addBarcode('0201234567890', builder.BARCODE_GS1_DATABAR_OMNIDIRECTIONAL);
    builder.addBarcode('0201234567890', builder.BARCODE_GS1_DATABAR_TRUNCATED);
    builder.addBarcode('0201234567890', builder.BARCODE_GS1_DATABAR_LIMITED);
    builder.addBarcode('02012345678903', builder.BARCODE_GS1_DATABAR_EXPANDED);
  }
</script>
```
addSymbol method

Adds two-dimensional symbol printing to the command buffer.

In page mode, a two-dimensional symbol is laid out in the current print position with the reference point being its bottom left dot.

Syntax

```java
addSymbol(data, type, level, width, height, size);
```

Parameter

- `data` : (Required parameter, Object type : String)
  Specifies two-dimensional symbol data as a character string.

<table>
<thead>
<tr>
<th>2D-Code type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard PDF417</td>
<td>Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. The data area can contain up to 928 code words in a maximum of 90 rows, each of which can contain up to 30 code words.</td>
</tr>
<tr>
<td>Truncated PDF417</td>
<td></td>
</tr>
<tr>
<td>QR Code Model 1</td>
<td>Convert the character string to the string in Shift-JIS, apply the escape sequence, and then encode the string based on the data type as shown below.</td>
</tr>
<tr>
<td>QR Code Model 2</td>
<td></td>
</tr>
<tr>
<td>Micro QR Code *2</td>
<td></td>
</tr>
<tr>
<td>2D-Code type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MaxiCode Mode 2</td>
<td>Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string.</td>
</tr>
<tr>
<td>MaxiCode Mode 3</td>
<td>In Modes 2 and 3, when the first piece of data is $&gt;$\x1e01\x1dyy (where yy is a two-digit number), this is processed as the message header, and the subsequent data is processed as the primary message. In other cases, from the first piece of data, data is processed as the primary message. In Mode 2, specify the primary message in the following format: Postal code (1- to 9-digit number) GS:(\x1d) ISO country code (1- to 3-digit number) GS:(\x1d) Service class code (1- to 3-digit number)</td>
</tr>
<tr>
<td>MaxiCode Mode 4</td>
<td>In Mode 3, specify the primary message in the following format: Postal code (1 to 6 pieces of data convertible by Code Set A) GS:(\x1d) ISO country code (1- to 3-digit number) GS:(\x1d) Service class code (1- to 3-digit number)</td>
</tr>
<tr>
<td>MaxiCode Mode 5</td>
<td></td>
</tr>
<tr>
<td>MaxiCode Mode 6</td>
<td></td>
</tr>
<tr>
<td>GS1 DataBar Stacked</td>
<td>Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. Specify a 13-digit global trade item number (GTIN) not including an application identifier (AI) or a check digit.</td>
</tr>
<tr>
<td>GS1 DataBar Stacked</td>
<td></td>
</tr>
<tr>
<td>Omnidirectional</td>
<td></td>
</tr>
<tr>
<td>GS1 DataBar Expanded Stacked</td>
<td>Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data. To encode each of the following characters, specify two characters starting with the character &quot;{{&quot;: FNC1: {{1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Aztec Code *1</td>
<td>After converting the character string to UTF-8, conduct the escape sequence and encode.</td>
</tr>
<tr>
<td>DataMatrix *1</td>
<td>After converting the character string to UTF-8, conduct the escape sequence and encode.</td>
</tr>
</tbody>
</table>

*1 ePOS-Print Service Ver.2.2 or later versions supported
*2 ePOS-Print Service Ver.4.1 or later versions supported
To specify binary data that cannot be represented by character strings, use the following escape sequences.

<table>
<thead>
<tr>
<th>String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\xn</td>
<td>Control code</td>
</tr>
<tr>
<td>\</td>
<td>Back slash</td>
</tr>
</tbody>
</table>

- type : ( Required parameter, Object type : String)
  Specifies the two-dimensional symbol type.

<table>
<thead>
<tr>
<th>Constant (type)</th>
<th>2D-Code type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYMBOL_PDF417_STANDARD</td>
<td>Standard PDF417</td>
</tr>
<tr>
<td>SYMBOL_PDF417_TRUNCATED</td>
<td>Truncated PDF417</td>
</tr>
<tr>
<td>SYMBOL_QRCODE_MODEL_1</td>
<td>QR Code Model 1</td>
</tr>
<tr>
<td>SYMBOL_QRCODE_MODEL_2</td>
<td>QR Code Model 2</td>
</tr>
<tr>
<td>SYMBOL_QRCODE_MICRO *2</td>
<td>Micro QR Code</td>
</tr>
<tr>
<td>SYMBOL_MAXICODE_MODE_2</td>
<td>MaxiCode Mode 2</td>
</tr>
<tr>
<td>SYMBOL_MAXICODE_MODE_3</td>
<td>MaxiCode Mode 3</td>
</tr>
<tr>
<td>SYMBOL_MAXICODE_MODE_4</td>
<td>MaxiCode Mode 4</td>
</tr>
<tr>
<td>SYMBOL_MAXICODE_MODE_5</td>
<td>MaxiCode Mode 5</td>
</tr>
<tr>
<td>SYMBOL_MAXICODE_MODE_6</td>
<td>MaxiCode Mode 6</td>
</tr>
<tr>
<td>SYMBOL_GS1_DATABAR_STACKED</td>
<td>GS1 DataBar Stacked</td>
</tr>
<tr>
<td>SYMBOL_GS1_DATABAR_STACKED_OMNIDIRECTIONAL</td>
<td>GS1 DataBar Stacked Omnidirectional</td>
</tr>
<tr>
<td>SYMBOL_GS1_DATABAR_EXPANDED_STACKED</td>
<td>GS1 DataBar Expanded Stacked</td>
</tr>
<tr>
<td>SYMBOL_AZTECCODE_FULLRANGE *1</td>
<td>Aztec Code Full-Range mode</td>
</tr>
<tr>
<td>SYMBOL_AZTECCODE_COMPACT *1</td>
<td>Aztec Code Compact mode</td>
</tr>
<tr>
<td>SYMBOL_DATAMATRIX_SQUARE *1</td>
<td>DataMatrix ECC200 square</td>
</tr>
<tr>
<td>SYMBOL_DATAMATRIX_RECTANGLE_8 *1</td>
<td>DataMatrix ECC200 rectangle, 8 lines</td>
</tr>
<tr>
<td>SYMBOL_DATAMATRIX_RECTANGLE_12 *1</td>
<td>DataMatrix ECC200 rectangle, 12 lines</td>
</tr>
<tr>
<td>SYMBOL_DATAMATRIX_RECTANGLE_16 *1</td>
<td>DataMatrix ECC200 rectangle, 16 lines</td>
</tr>
</tbody>
</table>

*1 ePOS-Print Service Ver.2.2 or later versions supported
*2 ePOS-Print Service Ver.4.1 or later versions supported
• **level**: (Optional parameter, Object type: String)
  Specifies the error correction level.

<table>
<thead>
<tr>
<th>Constant (level)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEVEL_0</td>
<td>PDF417 error correction level 0</td>
</tr>
<tr>
<td>LEVEL_1</td>
<td>PDF417 error correction level 1</td>
</tr>
<tr>
<td>LEVEL_2</td>
<td>PDF417 error correction level 2</td>
</tr>
<tr>
<td>LEVEL_3</td>
<td>PDF417 error correction level 3</td>
</tr>
<tr>
<td>LEVEL_4</td>
<td>PDF417 error correction level 4</td>
</tr>
<tr>
<td>LEVEL_5</td>
<td>PDF417 error correction level 5</td>
</tr>
<tr>
<td>LEVEL_6</td>
<td>PDF417 error correction level 6</td>
</tr>
<tr>
<td>LEVEL_7</td>
<td>PDF417 error correction level 7</td>
</tr>
<tr>
<td>LEVEL_8</td>
<td>PDF417 error correction level 8</td>
</tr>
<tr>
<td>LEVEL_L</td>
<td>QR Code error correction level L</td>
</tr>
<tr>
<td>LEVEL_M</td>
<td>QR Code error correction level M</td>
</tr>
<tr>
<td>LEVEL_Q</td>
<td>QR Code error correction level Q</td>
</tr>
<tr>
<td>LEVEL_H</td>
<td>QR Code error correction level H</td>
</tr>
<tr>
<td>LEVEL_DEFAULT</td>
<td>Default level</td>
</tr>
<tr>
<td>Integer from 5 to 95 *</td>
<td>Aztec Code error correction level (Default: 23)</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.2.2 or later versions supported

- Select the level according to the two-dimensional symbol type.
- For MaxiCode and two-dimensional GS1 DataBar, select LEVEL_DEFAULT.
- Micro QR Code does not support LEVEL_H.

• **width**: (Optional parameter, Object type: Number)
  Specifies the module width. Specifies an integer from 0 to 255.

<table>
<thead>
<tr>
<th>2D-Code type</th>
<th>Valid value range</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDF417</td>
<td>2 to 8</td>
<td>3</td>
</tr>
<tr>
<td>QR Code</td>
<td>3 to 16</td>
<td>3</td>
</tr>
<tr>
<td>MaxiCode</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>2D GS1 Databar</td>
<td>2 to 8</td>
<td>2</td>
</tr>
<tr>
<td>Aztec Code *</td>
<td>2 to 16</td>
<td>3</td>
</tr>
<tr>
<td>DataMatrix *</td>
<td>2 to 16</td>
<td>3</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.2.2 or later versions supported
• height : (Optional parameter, Object type : Number)
  Specifies the module height. Specifies an integer from 0 to 255.

<table>
<thead>
<tr>
<th>2D-Code type</th>
<th>Valid value range</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDF417</td>
<td>2 to 8 (Magnification for width)</td>
<td>3</td>
</tr>
<tr>
<td>QR Code</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>MaxiCode</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>2D GS1 Databar</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>Aztec Code *</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>DataMatrix *</td>
<td>Ignored</td>
<td></td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.2.2 or later versions supported

• size : (Optional parameter, Object type : Number)
  Specifies the two-dimensional symbol maximum size. Specifies an integer from 0 to 65535.

<table>
<thead>
<tr>
<th>2D-Code type</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDF417</td>
<td>0 (Auto)</td>
<td>Specifies the number of code words for each row</td>
</tr>
<tr>
<td>QR Code</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>MaxiCode</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>2D GS1 Databar</td>
<td>0 (Auto)</td>
<td>Specifies the maximum width for the barcode (106 or above)</td>
</tr>
<tr>
<td>Aztec Code *</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>DataMatrix *</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>(Others)</td>
<td>Ignored</td>
<td></td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.2.2 or later versions supported

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
**Example**

To print two-dimensional symbols:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addSymbol('ABCDE', builder.SYMBOL_PDF417_STANDARD);
    builder.addSymbol('ABCDE', builder.SYMBOL_QRCODE_MODEL_2,
                      builder.LEVEL_Q);
    builder.addSymbol('908063840\xd850\xd001\xd04',
                      builder.SYMBOL_MAXICODE_MODE_2);
    builder.addSymbol('0201234567890', builder.SYMBOL_GS1_DATABAR_STACKED);
    builder.addSymbol('0201234567890',
                      builder.SYMBOL_GS1_DATABAR_STACKED_OMNIDIRECTIONAL);
    builder.addSymbol('(01)02012345678903',
                      builder.SYMBOL_GS1_DATABAR_EXPANDED_STACKED);
    builder.addSymbol('ABCDE', builder.SYMBOL_AZTECCODE_FULLRANGE, 23);
    builder.addSymbol('ABCDE', builder.SYMBOL_DATAMATRIX_SQUARE);
}
//-->
</script>
```
addHLine method

Adds horizontal line printing to the command buffer.
Draws horizontal lines.

Syntax

```
addHLine(x1, x2, style);
```

Parameter

- **x1**: (Required parameter, Object type: Number)
  Specifies the start position of the horizontal line (in dots). Specifies an integer from 0 to 65535.
- **x2**: (Required parameter, Object type: Number)
  Specifies the end position of the horizontal line (in dots). Specifies an integer from 0 to 65535.
- **style**: (Optional parameter, Object type: String)
  Specifies the line type.

<table>
<thead>
<tr>
<th>Constant (style)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE_THIN</td>
<td>Solid line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM</td>
<td>Solid line: Medium</td>
</tr>
<tr>
<td>LINE_THICK</td>
<td>Solid line: Thick</td>
</tr>
<tr>
<td>LINE_THIN_DOUBLE</td>
<td>Double line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM_DOUBLE</td>
<td>Double line: Medium</td>
</tr>
<tr>
<td>LINE_THICK_DOUBLE</td>
<td>Double line: Thick</td>
</tr>
<tr>
<td>undefined</td>
<td>Solid line: Thin</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
**Example**

To draw **double horizontal lines in the following positions:**
- Between 100 dots and 200 dots from the left end
- Between 400 dots and 500 dots from the left end

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  //builder.addHLine(100, 200, builder.LINE_THIN_DOUBLE);
  builder.addHLine(100, 200, builder.LINE_THIN_DOUBLE);
  builder.addHLine(400, 500, builder.LINE_THIN_DOUBLE);
//-->
</script>
```
addVLineBegin method

Adds the beginning of vertical line to the command buffer. Starts to draw vertical lines.

- Not available in page mode.
- Vertical lines are drawn until their end is specified by addVLineEnd (p. 105). Use this API function with addVLineEnd.

Syntax

```
addVLineBegin(x, style);
```

Parameter

- **x**: (Required parameter, Object type : Number)
  
  Specifies the start position of the vertical line (in dots). Specifies an integer from 0 to 65535.

- **style**: (Optional parameter, Object type : String)
  
  Specifies the line type.

<table>
<thead>
<tr>
<th>Constant (style)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE_THIN</td>
<td>Solid line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM</td>
<td>Solid line: Medium</td>
</tr>
<tr>
<td>LINE_THICK</td>
<td>Solid line: Thick</td>
</tr>
<tr>
<td>LINE_THIN_DOUBLE</td>
<td>Double line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM_DOUBLE</td>
<td>Double line: Medium</td>
</tr>
<tr>
<td>LINE_THICK_DOUBLE</td>
<td>Double line: Thick</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Solid line: Thin</td>
</tr>
</tbody>
</table>

Return value

```
<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>
```

Exception

```
<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
```

Example

To draw thin vertical lines at 100 dots and 200 dots from the left end:

```javascript
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addVLineBegin(100).addVLineBegin(200);
  builder.addFeedUnit(100);
  builder.addVLineEnd(100).addVLineEnd(200);
}
```
### addVLineEnd method

Adds the end of vertical line to the command buffer. Finishes drawing vertical lines.

- Not available in page mode.
- Use this API function with addVLineBegin (p. 104).

#### Syntax

```plaintext
addVLineEnd(x, style);
```

#### Parameter

- **x**: (Required parameter, Object type : Number)
  
  Specifies the end position of the vertical line (in dots). Specifies an integer from 0 to 65535.

- **style**: (Optional parameter, Object type : String)
  
  Specifies the type of the line you want to finish drawing.

#### Return value

<table>
<thead>
<tr>
<th>Constant (style)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE_THIN</td>
<td>Solid line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM</td>
<td>Solid line: Medium</td>
</tr>
<tr>
<td>LINE_THICK</td>
<td>Solid line: Thick</td>
</tr>
<tr>
<td>LINE_THIN_DOUBLE</td>
<td>Double line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM_DOUBLE</td>
<td>Double line: Medium</td>
</tr>
<tr>
<td>LINE_THICK_DOUBLE</td>
<td>Double line: Thick</td>
</tr>
<tr>
<td>undefined</td>
<td>Solid line: Thin (When not specified)</td>
</tr>
</tbody>
</table>

#### Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

#### Example

To draw thin vertical lines at 100 dots and 200 dots from the left end:

```javascript
<script src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addVLineBegin(100).addVLineBegin(200);
    builder.addFeedUnit(100);
    builder.addVLineEnd(100).addVLineEnd(200);
  }
</script>
addPageBegin method

Adds the switching to page mode to the command buffer. The page mode process starts.

Vertical lines are processed in page mode until their end is specified by addPageEnd (p. 107). Use this API function with addPageEnd.

Syntax

```javascript
addPageBegin();
```

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Example

To print the characters "ABCDE" in page mode:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//-->
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addPageBegin();
    builder.addText('ABCDE');
    builder.addPageEnd();
}
//-->
</script>
```
addPageEnd method

Adds the end of page mode to the command buffer. The page mode process ends.

Use this API function with addPageBegin (p. 106).

Syntax

```
addPageEnd();
```

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Example

To print the characters "ABCDE" in page mode:

```
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addPageBegin();
  builder.addText('ABCDE');
  builder.addPageEnd();
}
//-->
</script>
```
addPageArea method

Adds the print area in page mode to the command buffer.
Specifies the print area in page mode (coordinates). After this API function, specify a print data API function such as the addText method.

- Specify a print area to cover the content to be printed. If the print data extends beyond the print area, the print result will be such that the print data has been printed incompletely.
- Use this API function by inserting it between addPageBegin (p. 106) and addPageEnd (p. 107).

Syntax

```
addPageArea(x, y, width, height);
```

Parameter

- **x**: (Required parameter, Object type: Number)
  Specifies the origin of the horizontal axis (in dots). Specifies an integer from 0 to 65535. 0 is the left end of the printer's printable area.

- **y**: (Required parameter, Object type: Number)
  Specifies the origin of the vertical axis (in dots). Specifies an integer from 0 to 65535. 0 is the position in which no paper feed has been performed.

- **width**: (Required parameter, Object type: Number)
  Specifies the width of the print area (in dots). Specifies an integer from 0 to 65535.

- **height**: (Required parameter, Object type: Number)
  Specifies the height of the print area (in dots). Specifies an integer from 0 to 65535.

⚠️ Determine the width and height of the print area according to the print direction setting. Otherwise, the print data might not be printed completely.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To specify the print area with the origin (100, 50), a width of 200 dots, and a height of 30 dots and print the characters "ABCDE":

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addPageBegin();
    builder.addPageArea(100, 50, 200, 30);
    builder.addText('ABCDE');
    builder.addPageEnd();
}
//-->
</script>
addPageDirection method

Adds the page mode print direction setting to the command buffer. Specifies the print direction in page mode. This function can be omitted if rotation is not required.

Use this API function by inserting it between addPageBegin (p. 106) and addPageEnd (p. 107).

**Syntax**

```plaintext
addPageDirection(dir);
```

**Parameter**

- `dir` : (Required parameter, Object type : String)
  Specifies the print direction in page mode.

<table>
<thead>
<tr>
<th>Constant (dir)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRECTION_LEFT_TO_RIGHT(default)</td>
<td>Left to right (No rotation. Data is printed from the top left corner to the right.)</td>
</tr>
<tr>
<td>DIRECTION_BOTTOM_TO_TOP</td>
<td>Bottom to top (Counterclockwise rotation by 90 degrees. Data is printed from the bottom left corner to the top.)</td>
</tr>
<tr>
<td>DIRECTION_RIGHT_TO_LEFT</td>
<td>Right to left (Rotation by 180 degrees. Data is printed from the bottom right corner to the left.)</td>
</tr>
<tr>
<td>DIRECTION_TOP_TO_BOTTOM</td>
<td>Top to bottom (Clockwise rotation by 90 degrees. Data is printed from the top right corner to the bottom.)</td>
</tr>
</tbody>
</table>

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To print the characters "ABCDE" by rotating them 90 degrees clockwise:

```javascript
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addPageBegin();
    builder.addPageArea(100, 50, 30, 200);
    builder.addPageDirection(builder.DIRECTION_TOP_TO_BOTTOM);
    builder.addText('ABCDE');
    builder.addPageEnd();
}
```

//-->
addPagePosition method

Adds the page mode print-position-set area to the command buffer.
Specifies the print start position (coordinates) in the area specified by the addPageArea method.

Use this API function by inserting it between addPageBegin (p. 106) and addPageEnd (p. 107).

**Syntax**

```
addPagePosition(x, y);
```

**Parameter**

- **x**: (Required parameter, Object type : Number)
  Specifies the horizontal print position (in dots). Specifies an integer from 0 to 65535.
- **y**: (Required parameter, Object type : Number)
  Specifies the vertical print position (in dots). Specifies an integer from 0 to 65535.

Specify the print start position (coordinates) according to the content to be printed. Refer to the following.
- To print a character string:
  Specify the left end of the baseline for the first character. This can be omitted for left-aligned printing of standard-sized characters. To print double-sized height characters, specify a value equal to or greater than 42 for y.
- To print a barcode:
  Specify the bottom left of the symbol. And specify the barcode height for y.
- To print a graphic/logo:
  Specify the bottom left of the graphic data. And specify the graphic data height for y.
- To print a two-dimensional symbol:
  Specify the top left of the symbol. This can be omitted when printing from the top left.

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
**Example**

To specify (50,30) for the print start position in the area specified by the `addPageArea` method and print the characters "ABCDE":

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
!---
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addPageBegin();
    builder.addPageArea(100, 50, 200, 100);
    builder.addPagePosition(50, 30);
    builder.addText('ABCDE');
    builder.addPageEnd();
}
//--></script>
```
addPageLine method

Adds line drawing in page mode to the command buffer. Draws a line in page mode.

- Diagonal lines cannot be drawn.
- Use this API function by inserting it between addPageBegin (p. 106) and addPageEnd (p. 107).

Syntax

```javascript
addPageLine(x1, y1, x2, y2, style);
```

Parameter

- **x1**: (Required parameter, Object type: Number)
  Specifies the horizontal start position of the line (in dots). Specifies an integer from 0 to 65535.

- **y1**: (Required parameter, Object type: Number)
  Specifies the vertical start position of the line (in dots). Specifies an integer from 0 to 65535.

- **x2**: (Required parameter, Object type: Number)
  Specifies the horizontal end position of the line (in dots). Specifies an integer from 0 to 65535.

- **y2**: (Required parameter, Object type: Number)
  Specifies the vertical end position of the line (in dots). Specifies an integer from 0 to 65535.

- **style**: (Optional parameter, Object type: String)
  Specifies the line type.

<table>
<thead>
<tr>
<th>Constant (style)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE_THIN</td>
<td>Solid line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM</td>
<td>Solid line: Medium</td>
</tr>
<tr>
<td>LINE_THICK</td>
<td>Solid line: Thick</td>
</tr>
<tr>
<td>LINE_THIN_DOUBLE</td>
<td>Double line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM_DOUBLE</td>
<td>Double line: Medium</td>
</tr>
<tr>
<td>LINE_THICK_DOUBLE</td>
<td>Double line: Thick</td>
</tr>
<tr>
<td>undefined</td>
<td>Solid line: Thin</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To draw a thin solid line between the start position (100, 0) and the end position (500, 0):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addPageBegin();
    builder.addPageLine(100, 0, 500, 0, builder.LINE_THIN);
    builder.addPageEnd();
}
//-->
</script>
```
**addPageRectangle method**

Adds rectangle drawing in page mode to the command buffer. Draws a rectangle in page mode.

Use this API function by inserting it between `addPageBegin (p. 106)` and `addPageEnd (p. 107)`.

**Syntax**

```plaintext
addPageRectangle(x1, y1, x2, y2, style);
```

**Parameter**

- **x1**: (Required parameter, Object type : Number)
  Specifies the horizontal start position of the line (in dots). Specifies an integer from 0 to 65535.

- **y1**: (Required parameter, Object type : Number)
  Specifies the vertical start position of the line (in dots). Specifies an integer from 0 to 65535.

- **x2**: (Required parameter, Object type : Number)
  Specifies the horizontal end position of the line (in dots). Specifies an integer from 0 to 65535.

- **y2**: (Required parameter, Object type : Number)
  Specifies the vertical end position of the line (in dots). Specifies an integer from 0 to 65535.

- **style**: (Optional parameter, Object type : String)
  Specifies the line type.

**Constant (style)**

<table>
<thead>
<tr>
<th>Constant (style)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE_THIN</td>
<td>Solid line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM</td>
<td>Solid line: Medium</td>
</tr>
<tr>
<td>LINE_THICK</td>
<td>Solid line: Thick</td>
</tr>
<tr>
<td>LINE_THIN_DOUBLE</td>
<td>Double line: Thin</td>
</tr>
<tr>
<td>LINE_MEDIUM_DOUBLE</td>
<td>Double line: Medium</td>
</tr>
<tr>
<td>LINE_THICK_DOUBLE</td>
<td>Double line: Thick</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Solid line: Thin</td>
</tr>
</tbody>
</table>

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot; ... &quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To draw a rectangle with a thin double line, with the start position (100, 0) and the end position (500, 200) as its vertexes:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addPageBegin();
  builder.addPageLine(100, 0, 500, 200, builder.LINE_THIN_DOUBLE);
  builder.addPageEnd();
}
//-->
</script>
```
addCut method

Adds paper cut to the command buffer. Sets paper cut.

Not available in page mode.

Syntax

`addCut(type);`

Parameter

- type : (Optional parameter, Object type : String)
  Specifies the paper cut type.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUT_NO_FEED</td>
<td>Cut without feeding</td>
</tr>
<tr>
<td></td>
<td>(The paper is cut without being fed.)</td>
</tr>
<tr>
<td>CUT_FEED</td>
<td>Feed cut</td>
</tr>
<tr>
<td></td>
<td>(The paper is fed to the cut position and then is cut.)</td>
</tr>
<tr>
<td>CUT_RESERVE</td>
<td>Cut reservation</td>
</tr>
<tr>
<td></td>
<td>(Printing continues until the cut position is reached, at which the paper is cut.)</td>
</tr>
<tr>
<td>undefined</td>
<td>Feed cut</td>
</tr>
<tr>
<td></td>
<td>(When not specified)</td>
</tr>
<tr>
<td></td>
<td>(The paper is fed to the cut position and then is cut.)</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To perform feed cut operation:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//-->
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addCut(builder.CUT_FEED);
}
//-->
</script>```
addPulse method

Adds the drawer kick to the command buffer. Sets the drawer kick.

- Not available in page mode.
- The drawer and the buzzer cannot be used together.

Syntax

```
addPulse(drawer, time);
```

Parameter

- **drawer**: (Optional parameter, Object type: String)
  Specifies the drawer kick connector.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRAWER_1</td>
<td>Pin 2 of the drawer kick-out connector</td>
</tr>
<tr>
<td>DRAWER_2</td>
<td>Pin 5 of the drawer kick-out connector</td>
</tr>
<tr>
<td>undefined</td>
<td>Pin 2 of the drawer kick-out connector</td>
</tr>
</tbody>
</table>

- **time**: (Optional parameter, Object type: String)
  Specifies the ON time of the drawer kick signal.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PULSE_100</td>
<td>100 ms</td>
</tr>
<tr>
<td>PULSE_200</td>
<td>200 ms</td>
</tr>
<tr>
<td>PULSE_300</td>
<td>300 ms</td>
</tr>
<tr>
<td>PULSE_400</td>
<td>400 ms</td>
</tr>
<tr>
<td>PULSE_500</td>
<td>500 ms</td>
</tr>
<tr>
<td>undefined</td>
<td>100 ms</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>
Example

To send a 100 msec pulse signal to the pin 2 of the drawer kick connector:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<scrip type="text/javascript">
<!--
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  builder.addPulse(builder.DRAWER_1, builder.PULSE_100);
}
//-->
</script>
```
**addSound method**

Adds the turning on of the buzzer to the command buffer. Sets the buzzer.

- Not available in page mode.
- The buzzer function and the drawer cannot be used together.
- This API function cannot be used if the printer is not provided with the buzzer.

**Syntax**

```
addSound(pattern, repeat, cycle);
```

**Parameter**

- pattern: (Optional parameter, Object type: String)
  Specifies the buzzer pattern.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATTERN_NONE</td>
<td>Stop</td>
</tr>
<tr>
<td>PATTERN_A</td>
<td>Pattern A</td>
</tr>
<tr>
<td>PATTERN_B</td>
<td>Pattern B</td>
</tr>
<tr>
<td>PATTERN_C</td>
<td>Pattern C</td>
</tr>
<tr>
<td>PATTERN_D</td>
<td>Pattern D</td>
</tr>
<tr>
<td>PATTERN_E</td>
<td>Pattern E</td>
</tr>
<tr>
<td>PATTERN_ERROR</td>
<td>Error sound pattern</td>
</tr>
<tr>
<td>PATTERN_PAPER_END</td>
<td>Pattern when there is no paper</td>
</tr>
<tr>
<td>PATERN_1 *</td>
<td>Pattern 1</td>
</tr>
<tr>
<td>PATERN_2 *</td>
<td>Pattern 2</td>
</tr>
<tr>
<td>PATERN_3 *</td>
<td>Pattern 3</td>
</tr>
<tr>
<td>PATERN_4 *</td>
<td>Pattern 4</td>
</tr>
<tr>
<td>PATERN_5 *</td>
<td>Pattern 5</td>
</tr>
<tr>
<td>PATERN_6 *</td>
<td>Pattern 6</td>
</tr>
<tr>
<td>PATERN_7 *</td>
<td>Pattern 7</td>
</tr>
<tr>
<td>PATERN_8 *</td>
<td>Pattern 8</td>
</tr>
<tr>
<td>PATERN_9 *</td>
<td>Pattern 9</td>
</tr>
<tr>
<td>PATERN_10 *</td>
<td>Pattern 10</td>
</tr>
<tr>
<td>undefined (When not specified)</td>
<td>Pattern A</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.2.2 or later versions supported
• repeat : (Optional parameter, Object type: String)
  Specifies the number of repeats.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The buzzer does not stop.</td>
</tr>
<tr>
<td>1 to 255</td>
<td>Number of repeats</td>
</tr>
<tr>
<td>undefined</td>
<td>(When not specified)</td>
</tr>
<tr>
<td></td>
<td>One time</td>
</tr>
</tbody>
</table>

After “0” is specified for repeat, if you want to stop the buzzer, execute this API function and specify PATTERN_NONE for pattern.

• cycle : (Optional parameter, Object type: String, When not specified: 1000)
  (in ePOS-Print Service Ver.2.2 and later)
  Specifies the buzzer sounding cycle (in units of milliseconds)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000 to 25500</td>
<td>1000 to 25500 milliseconds</td>
</tr>
<tr>
<td>undefined</td>
<td>1000 milliseconds</td>
</tr>
</tbody>
</table>

PATTERN_A to PATTERN_E/PATTERN_ERROR/PATTERN_PAPER_END is disregarded.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

To repeat the sound pattern A three times:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addSound(builder.PATTERN_A, 3);
}
//-->
</script>
```
addLayout method

Adds paper layout information to the command buffer. (in ePOS-Print Service Ver.2.2 and later)

Setting of page layout must be done in the standard mode. In the page mode, addLayout cannot be specified.

Syntax

```
addLayout(type, width, height, margin_top, margin_bottom, offset_cut, offset_label);
```

Parameter

- **type**: (Required parameter, Object type : String)
  Specifies the paper type.

<table>
<thead>
<tr>
<th>Constant (type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAYOUT_RECEIPT</td>
<td>Receipt (without black mark)</td>
</tr>
<tr>
<td>LAYOUT_RECEIPT_BM</td>
<td>Receipt (with black mark)</td>
</tr>
<tr>
<td>LAYOUT_LABEL</td>
<td>Die-cut label (without black mark)</td>
</tr>
<tr>
<td>LAYOUT_LABEL_BM</td>
<td>Die-cut label (with black mark)</td>
</tr>
</tbody>
</table>

- **width**: (Optional parameter, Object type : Number, When not specified : 580)
  Specifies paper width (in units of 0.1 mm). Specifies an integer from 290 to 600. *

- **height**: (Optional parameter, Object type : Number, When not specified : 0)
  Specifies paper height (in units of 0.1 mm).

<table>
<thead>
<tr>
<th>Paper Type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Distance from the top of black mark to the top of next black mark</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>0 (Auto)</td>
<td>Distance from the top of black mark to the top of next black mark</td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>284 to 1550 (Manual) *</td>
<td>Distance from the top of label to the top of next label</td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td></td>
<td>Distance from the bottom of black mark to the bottom of next black mark</td>
</tr>
</tbody>
</table>

- **margin_top**: (Optional parameter, Object type : Number, When not specified : 0)
  Specifies top margin (in units of 0.1 mm).

<table>
<thead>
<tr>
<th>Paper Type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>-150 to 1500 *</td>
<td>Distance from the top of black mark</td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>0 to 1500 *</td>
<td>Distance from the top of label</td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td></td>
<td>Distance from the bottom of black mark</td>
</tr>
</tbody>
</table>
- margin_bottom: (Optional parameter, Object type: Number, When not specified: 0)
  Specifies bottom margin (in units of 0.1 mm).

<table>
<thead>
<tr>
<th>Paper Type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>(without black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Receipt</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>(with black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Die-cut label</td>
<td>-15 to 0 *</td>
<td>Distance from the bottom of label (paper feed direction is a positive number)</td>
</tr>
<tr>
<td>(without black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Die-cut label</td>
<td>-15 to 15 *</td>
<td>Distance from the top of black mark (paper feed direction is a positive number)</td>
</tr>
<tr>
<td>(with black mark)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- offset_cut: (Optional parameter, Object type: Number, When not specified: 0)
  Specifies cut position (in units of 0.1 mm).
  In case of die cut label paper, it is a distance from the bottom of label.
  When a paper has black mark, it is a distance from the beginning of black mark.

<table>
<thead>
<tr>
<th>Paper Type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>(without black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Receipt</td>
<td>-290 to 50 *</td>
<td>Distance from the top of black mark to the cutting position</td>
</tr>
<tr>
<td>(with black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Die-cut label</td>
<td>0 to 50 *</td>
<td>Distance from the bottom of label to the cutting position</td>
</tr>
<tr>
<td>(without black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Die-cut label</td>
<td>0 to 50 *</td>
<td>Distance from the top of black mark to the cutting position</td>
</tr>
<tr>
<td>(with black mark)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- offset_label*: (Optional parameter, Object type: Number, When not specified: 0)
  Specifies label bottom position (sd) per 0.1 mm unit.

<table>
<thead>
<tr>
<th>Paper Type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>(without black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Receipt</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>(with black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Die-cut label</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>(without black mark)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Die-cut label</td>
<td>0 to 15 *</td>
<td>Distance from the top of black mark to the bottom of label</td>
</tr>
<tr>
<td>(with black mark)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*: Valid value of range is depending on the printer model. For detail, refer to Printer specifications (p.203).
Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

Detailed description

See below for the parameters that can be specified for each type of paper, and the positions for those parameters.

<table>
<thead>
<tr>
<th>Mark</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>sf</td>
<td>width</td>
</tr>
<tr>
<td>sa</td>
<td>height</td>
</tr>
<tr>
<td>sb</td>
<td>margin_top</td>
</tr>
<tr>
<td>se</td>
<td>margin_bottom</td>
</tr>
<tr>
<td>sc</td>
<td>offset_cut</td>
</tr>
<tr>
<td>sd</td>
<td>offset_label</td>
</tr>
</tbody>
</table>

Mark: sf

Parameter:

- **sf** width
- **sa** height
- **sb** margin_top
- **se** margin_bottom
- **sc** offset_cut
- **sd** offset_label

Receipt

<without Black Mark>

Label

<with Black Mark>

Cut

Top position

Bottom edge of print area

Top position

Bottom edge of label

Top position

Bottom edge of label

Standard eject
### Example

To set 58 mm receipt (without black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addLayout(builder.LAYOUT_RECEIPT, 580);
}
//-->
</script>
```

To set 58 mm receipt (with black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addLayout(builder.LAYOUT_RECEIPT_BM, 580, 0, 15, 0);
}
//-->
</script>
```

To set 58 mm die-cut label (without black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addLayout(builder.LAYOUT_LABEL, 580, 0, 15, -15, 25);
}
//-->
</script>
```

To set 58 mm die-cut label (with black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addLayout(builder.LAYOUT_LABEL_BM, 580, 0, 15, -15, 25, 15);
}
//-->
</script>
```
addRecovery method

Adds the recovery from errors to the command buffer. (In ePOS-Print Service Ver.3.0 and later)

Enable forced transmission mode to use this API. The printer recovers from errors that can be recovered from and clears the buffer.

**Syntax**

```javascript
addRecovery();
```

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

Recovers from errors that can be recovered from and clears the buffer:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.force = true;
    builder.addRecovery();
}
//-->
</script>
```
**addReset method**

Adds the printer reset to the command buffer. (in ePOS-Print Service Ver.3.0 and later)

Other printing commands in the print document are ignored.

**Syntax**

```javascript
addReset();
```

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

Resets the printer:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    <!--
    function buildMessage() {
        var builder = new epson.ePOSBuilder();
        builder.addReset();
    }
    //-->
</script>
```
addCommand method

Adds commands to the command buffer. Sends ESC/POS commands.

Syntax

```addCommand(data);```

Parameter

- data : (Optional parameter, Object type : String)
  Specifies ESC/POS command as a character string.

Return value

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePOS-Print Builder Object</td>
<td>ePOS Builder</td>
</tr>
</tbody>
</table>

Exception

| Exception             | Object type | |
|-----------------------|-------------|
| Parameter "..." is invalid | Error       |

Example

```xml
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function buildMessage() {
  var builder = new epson.ePOSBuilder();
  var doc = builder.addCommand('ABC\x44\x45\0a');
}
//-->
</script>
```
**toString method**

Obtains a print document generated by an ePOS-Print Builder object.

**Syntax**

```
toString();
```

**Return value**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document to be printed</td>
<td>String</td>
</tr>
</tbody>
</table>

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    var doc = builder.toString();
}
//-->
</script>
```
Halftone property

Halftone processing method.

Object type

String

Description

The halftone processing method to be applied to monochrome (two-tone) printing is specified. The default value is HALFTONE_DITHER.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HALFTONE_DITHER (default)</td>
<td>Dithering, suitable for printing graphics only.</td>
</tr>
<tr>
<td>HALFTONE_ERROR_DIFFUSION</td>
<td>Error diffusion, suitable for printing text and graphics together.</td>
</tr>
<tr>
<td>HALFTONE_THRESHOLD</td>
<td>Threshold, suitable for printing text only.</td>
</tr>
</tbody>
</table>

Example

To set the halftone type as error diffusion:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    var canvas = document.getElementById('canvas');
    if (canvas.getContext) {
      var context = canvas.getContext('2d');
      builder.halftone = epos.HALFTONE_ERROR_DIFFUSION;
      builder.addImage(context, 0, 0, canvas.width, canvas.height);
    }
  }
</script>
```
brightness property

Brightness correction value.

Object type
Number

Description
A gamma value in the range 0.1-10.0 is specified for the brightness correction value.
The default value is 1.0.

Example

To set brightness as 2.2:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
!-->
function buildMessage() {
var builder = new epson.ePOSBuilder();
var canvas = document.getElementById('canvas');
if (canvas.getContext) {
    var context = canvas.getContext('2d');
    builder.brightness = 2.2;
    builder.addImage(context, 0, 0, canvas.width, canvas.height);
}
//-->
</script>
```
**force property**

This is the forced transmission mode. (In ePOS-Print Service Ver.3.0 and later)

*Object type*
Boolean

*Description*
If you enable forced transmission mode, print commands are forcibly sent to the printer.

- Use forced transmission mode when the printer is offline. It will result in an error if the printer is online.
- The following functions are enabled in forced transmission mode.
  * Drawer kick-out (addPulse method (p.119))
  * Stopping the buzzer (addSound method (p.121))
  * Recovery from errors that can be recovered from (addRecovery method (p.127))
  * Reset (addReset method (p.128))
  * Sending commands in real time (addCommand method (p.129))

*Example*

Performs a drawer kick-out when the paper is at the end:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.force = true;
    builder.addPulse();
  }
</script>
```
message property

Command buffer.

Object type
String

Description
Commands, which are usually added by methods of the ePOS-Print Builder object, can be operated directly from this property for addition or deletion.

Example
To clear the command and reset it to the initial state:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>

<script type="text/javascript">
!-->
function buildMessage() {
    var builder = new epson.ePOSBuilder();
    builder.addText('ABCDE');
    builder.message = '';
}
//-->
</script>
```
**ePOS-Print Object**

Sends a print document created using an ePOS-Print Builder object to control the printer and monitor the transmission result or the communication status.

### Constructor

Constructor for an ePOS-Print object. Creates a new ePOS-Print object and initializes it.

**Syntax**

```javascript
ePOSPrint(address);
```

**Parameter**

- **address**: (Optional parameter, Object type : String)
  
  Specifies the URL of the printer to send a print document to.
  
  The URL is as follows:

  ```text
  http://[IP address of ePOS-Print supported TM printer]/cgi-bin/epos/
  service.cgi?devid=[device ID of printer to be used for printing]&timeout=[timeout time]
  ```

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  function sendMessage() {
    var address = 'http://192.168.192.168/cgi-bin/epos/
    service.cgi?devid=local_Printer';
    var epos = new epson.ePOSPrint(address);
  }
</script>
```
send method

Sends a print document created using an ePOS-Print Builder object.

A print document is obtained by executing the toString method (p. 130) of the ePOS-Print Builder object.

Syntax

- `send(request);`
- For acquiring the job ID
  
  `send(request, printjobid);`

The syntax used to specify job IDs is only compatible with ePOS-Print Service Ver.4.1 or later. For details, refer to Specifying the Print Job ID from the Application (p.22).

Parameter

- `request` : (Required parameter, Object type : String)
  
  Specifies the print document.

- `printjobid` : (Optional parameter, Object type : String)
  
  Specify the print job ID. (ePOS-Print Service Ver.4.1 or later versions)
  
  1 to 30 alphanumeric characters, including underscores, hyphens and periods, may be used.

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
<tr>
<td>XMLHttpRequest is not supported</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
   <!--
   function printHelloWorld() {
   var builder = new epson.ePOSBuilder();
   builder.addText('Hello, World!\n');
   builder.addCut();
   var request = builder.toString();
   var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
   var epos = new epson.ePOSPrint(address);
   epos.onreceive = function (res) { alert(res.success); };
   epos.onerror = function (err) { alert(err.status); };
   epos.send(request);
   }
   //-->
</script>
```
To specify a print job ID from the application

```javascript
var printjobid = 'ABC123';
var builder = new epson.ePOSBuilder();
builder.addText('Hello, World!\n');
builder.addCut();
var request = builder.toString();
var address = 'http://192.168.192.168/cgi-bin/epos/
service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onreceive = function (res) { alert(res.printjobid); }
epos.onerror = function (err) { alert(err.status); }
epos.send(request, printjobid);
```
getPrintJobStatus method

Acquires print job status. (ePOS-Print Service Ver.4.1 or later versions)

- The results of executing this API will be returned to an onreceive event (p.145).
- The latest print job status will be obtained when executing multiple printing processes with the same job ID.

Syntax

```
getPrintJobStatus(printjobid);
```

Parameter

- `printjobid` : (Required parameter, Object type : String)
  Specifies the print job ID to be verified.
  1 to 30 alphanumeric characters, including underscores, hyphens and periods, may be used.

Example

```javascript
var printjobid = '12345';
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi';
var epos = new epson.ePOSPrint(address);
epos.onreceive = function (res) { alert(res.success); };
epos.onerror = function (err) { alert(err.status); };
epos.getPrintJobStatus(printjobid);
```
open method

Enables status event operation.
Sends the status of the printer specified by the address property using an event.
Updates the status at the interval specified by the interval property.

Syntax

```javascript
open();
```

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.ePOSPrint(address);
  epos.oncoveropen = function () {
    alert('coveropen');
  };

  function startMonitor() {
    epos.open();
  }

  function stopMonitor() {
    epos.close();
  }
  //-->
</script>
```
close method

Disables status event operation.

**Syntax**

```
close();
```

**Example**

```<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.ePOSPrint(address);
  epos.oncoveropen = function () {
    alert('coveropen');
  };

  function startMonitor() {
    epos.open();
  }

  function stopMonitor() {
    epos.close();
  }
</script>```
address property

URL of the printer.

Object type
String

Description
The URL of the printer to be used for printing is specified.
The URL is shown as follows:

http://(IP address of ePOS-Print supported TM printer)/cgi-bin/epos/service.cgi?devid=(device ID of printer to be used for printing) & timeout=(timeout time)

The default value is the address specified by the constructor.

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//
var epos = new epson.ePOSPrint();
epos.address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
epos.oncoveropen = function () { alert('coveropen'); };
epos.open();
//-->
</script>
```
### enabled property

Retains the enabled/disabled setting for status event operation.

**Object type**
Boolean

**Description**
The enabled/disabled setting for status event operation is retained using a logical value. This is read-only. The default value is false.

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
 var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
 var epos = new epson.ePOSPrint(address);
 epos.oncoveropen = function () { alert('coveropen'); }; 
 epos.open();
 alert(epos.enabled);
//-->
</script>
```

### interval property

Specifies the interval of upgrading the status.

**Object type**
Number

**Description**
The interval of upgrading the status is specified in milliseconds. Default value: 3000 (three seconds) Minimum value: 1000 (one second or longer) When an invalid value is specified, it is assumed to be 3000.

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
 var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
 var epos = new epson.ePOSPrint(address);
 epos.interval = 1000;
 epos.oncoveropen = function () { alert('coveropen'); }; 
 epos.open();
//-->
</script>
```
status property

Status of the printer.

Object type
Number

Description
This is the status last obtained from the printer. This is read-only.
Default value: 0

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onoffline = function () {
    alert(epos.status);
};
epos.open();
//-->
</script>
```

battery property

Battery status of the printer.

Object type
Number

Description
Battery status obtained from the last printer status. This is read-only.
Default value: 0

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onbatterylow = function () {
    alert(epos.battery);
};
epos.open();
//-->
</script>
```
### timeout property

Specifies connection timeout.

**Object type**

**Number**

**Description**

Specifies connection timeout with ePOS-Print supported printer in milliseconds. When the transmission of print document by send method times out, onerror even is generated.

Default value: 300000 (5 minutes)

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var builder = new epson.ePOSBuilder();
builder.addText('Hello, World!
');
builder.addCut();
var request = builder.toString();
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.timeout = 60000;
epos.onreceive = function (res) { alert(res.success); }
epos.onerror = function (err) { alert(err.status); }
epos.send(request);
//-->
</script>
```
onreceive event

This property registers the callback function and obtains a response message receipt event.

**Syntax**

Function (response)

**Parameter of the callback function**

Parameters: response (See “Properties of the response object” on page 145.)

Name: Response message

Object type: Object

**Properties of the response object**

<table>
<thead>
<tr>
<th>Property</th>
<th>Name</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>success</td>
<td>Print result</td>
<td>Boolean</td>
</tr>
<tr>
<td>code</td>
<td>Error code</td>
<td>String</td>
</tr>
<tr>
<td>status</td>
<td>Status</td>
<td>Number</td>
</tr>
<tr>
<td>battery</td>
<td>Battery status</td>
<td>Number</td>
</tr>
<tr>
<td>printjobid</td>
<td>Print job ID</td>
<td>String</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported

- Value of success

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>• Printing succeeded</td>
</tr>
<tr>
<td></td>
<td>• Processing succeeded (When the spooler function is enabled)</td>
</tr>
<tr>
<td>false or 0</td>
<td>• Printing Failed</td>
</tr>
<tr>
<td></td>
<td>• Processing Failed (When the spooler function is enabled)</td>
</tr>
</tbody>
</table>

- Value of code

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'EPTR_AUTOMATICAL'</td>
<td>An automatically recoverable error occurred</td>
</tr>
<tr>
<td>'EPTR_BATTERY_LOW'</td>
<td>No remaining battery</td>
</tr>
<tr>
<td>'EPTR_COVER_OPEN'</td>
<td>A cover open error occurred</td>
</tr>
<tr>
<td>'EPTR_CUTTER'</td>
<td>An autocutter error occurred</td>
</tr>
<tr>
<td>'EPTR_MECHANICAL'</td>
<td>A mechanical error occurred</td>
</tr>
<tr>
<td>'EPTR_REC_EMPTY'</td>
<td>No paper in roll paper end sensor</td>
</tr>
<tr>
<td>'EPTR_UNRECOVERABLE'</td>
<td>An unrecoverable error occurred</td>
</tr>
<tr>
<td>'SchemaError'</td>
<td>The request document contains a syntax error</td>
</tr>
<tr>
<td>'DeviceNotFound'</td>
<td>The printer with the specified device ID does not exist</td>
</tr>
<tr>
<td>'PrintSystemError'</td>
<td>An error occurred on the printing system</td>
</tr>
<tr>
<td>'EX_BADPORT'</td>
<td>An error was detected on the communication port</td>
</tr>
<tr>
<td>'EX_TIMEOUT'</td>
<td>A print timeout occurred</td>
</tr>
<tr>
<td>'EX_SPOOLER' *</td>
<td>There is not enough space available in the printing queue</td>
</tr>
<tr>
<td>'JobNotFound' *</td>
<td>The specified job ID does not exist</td>
</tr>
<tr>
<td>'Printing' *</td>
<td>Print job now printing</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported
• Value of status

<table>
<thead>
<tr>
<th>Constant (status)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASB_NO_RESPONSE</td>
<td>No response from the TM printer</td>
</tr>
<tr>
<td>ASB_PRINT_SUCCESS</td>
<td>Printing is successfully completed</td>
</tr>
<tr>
<td>ASB_DRAWER_KICK</td>
<td>Status of the 3rd pin of the drawer kick-out connector = &quot;H&quot;</td>
</tr>
<tr>
<td>ASB_BATTERY_OFFLINE</td>
<td>Battery offline status (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_OFF_LINE</td>
<td>Offline</td>
</tr>
<tr>
<td>ASB_COVER_OPEN</td>
<td>The cover is open</td>
</tr>
<tr>
<td>ASB_PAPER_FEED</td>
<td>Paper is being fed by a paper feed switch operation</td>
</tr>
<tr>
<td>ASB_WAIT_ON_LINE</td>
<td>Waiting to be brought back online</td>
</tr>
<tr>
<td>ASB_PANEL_SWITCH</td>
<td>The paper feed switch is being pressed (ON)</td>
</tr>
<tr>
<td>ASB_MECHANICAL_ERR</td>
<td>A mechanical error occurred</td>
</tr>
<tr>
<td>ASB_AUTOCUTTER_ERR</td>
<td>An autocutter error occurred</td>
</tr>
<tr>
<td>ASB_UNRECOVER_ERR</td>
<td>An unrecoverable error occurred</td>
</tr>
<tr>
<td>ASB_AUTORECOVER_ERR</td>
<td>An automatically recoverable error occurred</td>
</tr>
<tr>
<td>ASB_RECEIPT_NEAR_END</td>
<td>No paper in roll paper near end sensor</td>
</tr>
<tr>
<td>ASB_RECEIPT_END</td>
<td>No paper in roll paper end sensor</td>
</tr>
<tr>
<td>ASB_BUZZER</td>
<td>A buzzer is on (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_WAIT_REMOVE_LABEL</td>
<td>Waiting for label to be removed (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_NO_LABEL</td>
<td>No paper in label peeling sensor (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_SPOOLER_IS_STOPPED *</td>
<td>The spooler has stopped</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported

• Value of battery

<table>
<thead>
<tr>
<th>Value (battery)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x30XX</td>
<td>The AC adapter is connected</td>
</tr>
<tr>
<td>0x31XX</td>
<td>The AC adapter is connected</td>
</tr>
</tbody>
</table>

Remaining battery

<table>
<thead>
<tr>
<th>Value (battery)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xXX36</td>
<td>Battery amount 6</td>
</tr>
<tr>
<td>0xXX35</td>
<td>Battery amount 5</td>
</tr>
<tr>
<td>0xXX34</td>
<td>Battery amount 4</td>
</tr>
<tr>
<td>0xXX33</td>
<td>Battery amount 3</td>
</tr>
<tr>
<td>0xXX32</td>
<td>Battery amount 2</td>
</tr>
<tr>
<td>0xXX31</td>
<td>Battery amount 1 (Near end)</td>
</tr>
<tr>
<td>0xXX30</td>
<td>Battery amount 0 (Real end)</td>
</tr>
</tbody>
</table>

0 is shown when the model doesn't have a battery installed.
• Value of printjobid

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;&quot;</td>
<td>Print job ID</td>
</tr>
<tr>
<td></td>
<td>(An empty string denotes that a print job ID has not been set)</td>
</tr>
</tbody>
</table>

**Example**

To create and send a print document.
To display the print result in a message box.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">-->
function printHelloWorld() {
  var builder = new epson.ePOSBuilder();
  builder.addText('Hello, World!\n');
  builder.addCut();
  var request = builder.toString();
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.ePOSPrint(address);
  epos.onreceive = function (res) {
    var success = res.success;
    var code = res.code;
    var status = res.status;
    alert(success);
  }
  epos.send(request);
}-->
</script>
```
onerror event

This property registers the callback function and obtains a communication error event.

Syntax

Function (error)

Parameter of the callback function

Parameters: error (See “Properties of the error object” on page 148.)
Name: Communication error information
Object type: Object

Properties of the error object

<table>
<thead>
<tr>
<th>Property</th>
<th>Name</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>HTTP Status</td>
<td>Number</td>
</tr>
<tr>
<td>responseText</td>
<td>Response text</td>
<td>String</td>
</tr>
</tbody>
</table>

Example

To create and send a print document.
To display the HTTP status code in a message box when a communication error occurs.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function printHelloWorld() {
  var builder = new epson.ePOSBuilder();
  builder.addText('Hello, World!' +\n');
  builder.addCut();
  var request = builder.toString();
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.ePOSPrint(address);
  epos.onerror = function (err) {
    var status = err.status;
    var text = err.responseText;
    alert(status);
  }
  epos.send(request);
}
//-->
</script>
```
onstatuschange event

Registers a callback function to obtain a status change event.

**Syntax**

Function (status)

**Parameter of the callback function**

- Parameters: status
- Name: Status
- Object type: Number

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onstatuschange = function (status) {
    alert(status);
};
epos.open();
//-->
</script>
```

onbatterystatuschange event

Registers call back function and obtains battery status change event. (in ePOS-Print Service Ver.2.2 and later)

**Object type**

Function (battery)

**Parameter of the callback function**

- Parameters: battery
- Name: Batterystatus
- Object type: Number

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onbatterystatuschange = function (battery) {
    alert(battery);
};
epos.open();
//-->
</script>
```
**ononline event**

Registers a callback function to obtain an online event.

*Object type*
Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.ePOSPrint(address);
    epos.ononline = function () {
        alert('online');
    };
    epos.open();
</script>
```

**onoffline event**

Registers a callback function to obtain an offline event.

*Object type*
Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.ePOSPrint(address);
    epos.onoffline = function () {
        alert('offline');
    };
    epos.open();
</script>
```
onpoweroff event

 Registers a callback function to obtain a non-response event.

 Object type
 Function ()

 Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">

 var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
 var epos = new epson.ePOSPrint(address);
 epos.onpoweroff = function () {
   alert('poweroff');
   };
 epos.open();
 //</script>
```

oncoverok event

 Registers a callback function to obtain a cover close event.

 Object type
 Function ()

 Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">

 var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
 var epos = new epson.ePOSPrint(address);
 epos.oncoverok = function () {
   alert('coverok');
   };
 epos.open();
 //</script>
```
oncoveropen event

Registers a callback function to obtain a cover open event.

**Object type**

Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.ePOSPrint(address);
  epos.oncoveropen = function () {
    alert('coveropen');
  };
  epos.open();
</script>
```

onpaperok event

Registers a callback function to obtain a paper remaining event.

**Object type**

Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.ePOSPrint(address);
  epos.onpaperok = function () {
    alert('paperok');
  };
  epos.open();
</script>
```
**onpapernearend event**

Registers a callback function to obtain a paper near end event.

*Object type*

Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.ePOSPrint(address);
    epos.onpapernearend = function () {
        alert('papernearend');
    };
    epos.open();
    //-->
</script>
```

**onpaperend event**

Registers a callback function to obtain a paper end event.

*Object type*

Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.ePOSPrint(address);
    epos.onpaperend = function () {
        alert('paperend');
    };
    epos.open();
    //-->
</script>
```
ondrawerclosed event

Registers a callback function to obtain a drawer close event.

**Object type**

Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.ondrawerclosed = function () {
    alert('drawerclosed');
};
epos.open();
//--></script>
```

ondraweropen event

Registers a callback function to obtain a drawer open event.

**Object type**

Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.ondraweropen = function () {
    alert('draweropen');
};
epos.open();
//-->
</script>
```
**onbatteryok event**

Registers call back function and obtains remaining battery event. (in ePOS-Print Service Ver.2.2 and later)

*Object type*

Function ()

*Example*

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//!
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onbatteryok = function () {
  alert('batteryok');
};
epos.open();
//</script>
```

**onbatterylow event**

Registers call back function and obtains no remaining battery event. (in ePOS-Print Service Ver.2.2 and later)

*Object type*

Function ()

*Example*

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//!
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.ePOSPrint(address);
epos.onbatterylow = function () {
  alert('batterylow');
};
epos.open();
//</script>
```
This chapter describes the ePOS-Print Canvas API.

List of ePOS-Print Canvas API functions

The ePOS-Print Canvas API provides the following object:
- ePOS-Print Canvas API (window.epson.CanvasPrint) object (p. 157)

window.epson.CanvasPrint Components

<table>
<thead>
<tr>
<th>Element</th>
<th>API</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CanvasPrint</td>
<td>Initializes an ePOS-Print Canvas API object.</td>
<td>159</td>
</tr>
<tr>
<td>method</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>print</td>
<td>Prints an HTML5 Canvas image.</td>
<td>160</td>
</tr>
<tr>
<td></td>
<td>getPrintJobStatus *4</td>
<td>Acquires print job status.</td>
<td>162</td>
</tr>
<tr>
<td></td>
<td>open</td>
<td>Enables status event operation</td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>close</td>
<td>Disables status event operation</td>
<td>164</td>
</tr>
<tr>
<td></td>
<td>recover *3</td>
<td>Recovers from an error</td>
<td>165</td>
</tr>
<tr>
<td></td>
<td>reset *3</td>
<td>Resets the printer</td>
<td>165</td>
</tr>
<tr>
<td>Property</td>
<td>address</td>
<td>URL of the printer</td>
<td>166</td>
</tr>
<tr>
<td></td>
<td>enabled</td>
<td>Enabling/disabling of status event</td>
<td>167</td>
</tr>
<tr>
<td></td>
<td>interval</td>
<td>Printer status update interval</td>
<td>168</td>
</tr>
<tr>
<td></td>
<td>status</td>
<td>Status</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td>battery</td>
<td>Battery status</td>
<td>170</td>
</tr>
<tr>
<td></td>
<td>timeout</td>
<td>Connection timeout</td>
<td>171</td>
</tr>
<tr>
<td></td>
<td>halftone</td>
<td>Raster image halftone processing method</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td>brightness</td>
<td>Raster image brightness correction value</td>
<td>173</td>
</tr>
<tr>
<td></td>
<td>cut</td>
<td>Paper cut</td>
<td>174</td>
</tr>
<tr>
<td>Property</td>
<td>mode</td>
<td>Color mode</td>
<td>175</td>
</tr>
<tr>
<td></td>
<td>align</td>
<td>Position alignment</td>
<td>176</td>
</tr>
<tr>
<td></td>
<td>color</td>
<td>Printing color</td>
<td>177</td>
</tr>
<tr>
<td></td>
<td>feed *1</td>
<td>Control of label paper/black mark paper</td>
<td>178</td>
</tr>
<tr>
<td></td>
<td>paper *1</td>
<td>Type of papers</td>
<td>179</td>
</tr>
<tr>
<td></td>
<td>layout *2</td>
<td>Paper layout</td>
<td>180</td>
</tr>
<tr>
<td>Element</td>
<td>API</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------</td>
<td>--------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Event</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onreceive</td>
<td>onerror</td>
<td>Response message receipt event</td>
<td>185</td>
</tr>
<tr>
<td>onerror</td>
<td>onstatuschange</td>
<td>Communication error event</td>
<td>188</td>
</tr>
<tr>
<td>onstatuschange</td>
<td>onbatterystatuschange</td>
<td>Status change event</td>
<td>189</td>
</tr>
<tr>
<td>onbatterystatuschange</td>
<td>onbatteryok</td>
<td>Battery status change event</td>
<td>190</td>
</tr>
<tr>
<td>onbatterystatuschange</td>
<td>onbatterylow</td>
<td>Battery OK event</td>
<td>191</td>
</tr>
<tr>
<td>onbatterylow</td>
<td>ononline</td>
<td>Battery low event</td>
<td>191</td>
</tr>
<tr>
<td>ononline</td>
<td>onoffline</td>
<td>Online event</td>
<td>192</td>
</tr>
<tr>
<td>onoffline</td>
<td>onpoweroff</td>
<td>Offline event</td>
<td>192</td>
</tr>
<tr>
<td>onpoweroff</td>
<td>oncoverok</td>
<td>Non-response event</td>
<td>193</td>
</tr>
<tr>
<td>oncoverok</td>
<td>oncoveropen</td>
<td>Cover close event</td>
<td>193</td>
</tr>
<tr>
<td>oncoveropen</td>
<td>onpaperevent</td>
<td>Cover open event</td>
<td>194</td>
</tr>
<tr>
<td>onpaperevent</td>
<td>onpapernear</td>
<td>Paper remaining event</td>
<td>194</td>
</tr>
<tr>
<td>onpapernear</td>
<td>onpapernear</td>
<td>Paper near end event</td>
<td>195</td>
</tr>
<tr>
<td>onpapernear</td>
<td>onpoweroff</td>
<td>Paper end event</td>
<td>195</td>
</tr>
<tr>
<td>onpoweroff</td>
<td>ononlinelow</td>
<td>Drawer close event</td>
<td>196</td>
</tr>
<tr>
<td>ononlinelow</td>
<td>oncoveropen</td>
<td>Drawer open event</td>
<td>196</td>
</tr>
<tr>
<td>Constant</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASB_*</td>
<td>HALFTONE_*</td>
<td>Response document status</td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td>MODE_*</td>
<td>Halftone type</td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td>ALIGN_*</td>
<td>Color mode</td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td>COLOR_*</td>
<td>Position alignment</td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td>FEED_*</td>
<td>Color specification</td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td>PAPER_*</td>
<td>Paper feed position of label paper/black mark paper</td>
<td></td>
</tr>
<tr>
<td>Constant</td>
<td></td>
<td>Type of papers</td>
<td></td>
</tr>
</tbody>
</table>

*1 ePOS-Print Service Ver.2.1 or later versions supported  
*2 ePOS-Print Service Ver.2.2 or later versions supported  
*3 ePOS-Print Service Ver.3.0 or later versions supported  
*4 ePOS-Print Service Ver.4.1 or later versions supported
ePOS-Print Canvas API Object

Prints a print image rendered in HTML5 Canvas and monitors the print result or the communication status.

**Constructor**

Constructor for an ePOS-Print Canvas API object.
Creates a new ePOS-Print Canvas API object and initializes it.

**Syntax**

```javascript
CanvasPrint(address);
```

**Parameter**

- `address`: (Optional parameter, Object type: String)
  Specifies the address property (URL of printer to be used for printing).
  The URL is as follows:

```javascript
http://[ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=[device ID of printer to be used for printing]&timeout=[timeout time]
```

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function printCanvas() {
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
}
//-->
</script>
```
print method

Prints an image rendered in HTML5 Canvas. Converts the specified range in a RGBA full-color image of HTML5 Canvas into raster image data according to the settings of the halftone and brightness properties. One pixel in an image equals to one printer dot. When an image contains any transparent color, the background color of the image is assumed to be white.

| Warning | If an HTML5 Canvas image contains images downloaded from different domains, you cannot print the image. In this case, a security error occurs due to violation of the same origin policy of JavaScript. |

Syntax

- `print(canvas);`

- For acquiring the job ID
  ```javascript
  print(canvas, printjobid);
  ```

  The syntax used to specify job IDs is only compatible with ePOS-Print Service Ver.4.1 or later. For details, refer to Specifying the Print Job ID from the Application (p.22).

- `print(canvas, cut, mode);`

  This syntax is compatible version.

Parameter

- `canvas` : (Required parameter, Object type : canvas)
  Specify the HTML5 Canvas object to be printed.

- `printjobid` : (Optional parameter, Object type : String)
  Specify the print job ID. (ePOS-Print Service Ver.4.1 or later versions)

  1 to 30 alphanumeric characters, including underscores, hyphens and periods, may be used.

- `cut` : (Optional parameter, Object type : Boolean)
  Sets whether to cut paper.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true or 1</td>
<td>Cuts the paper after printing</td>
</tr>
<tr>
<td>false or 0</td>
<td>Does not cut the paper after printing</td>
</tr>
<tr>
<td>undefined</td>
<td>Does not cut the paper after printing</td>
</tr>
</tbody>
</table>

- `mode` : (Optional parameter, Object type : String)
  Specifies the color mode.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_MONO</td>
<td>Monochrome (two-tone)</td>
</tr>
<tr>
<td>MODE_GRAY16</td>
<td>Multiple tones (16-tone)</td>
</tr>
<tr>
<td>undefined</td>
<td>Monochrome (two-tone)</td>
</tr>
</tbody>
</table>
**Exception**

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot; is invalid</td>
<td>Error</td>
</tr>
<tr>
<td>XMLHttpRequest is not supported</td>
<td>Error</td>
</tr>
<tr>
<td>Canvas is not supported</td>
<td>Error</td>
</tr>
</tbody>
</table>

**Example**

To print Canvas(ID='myCanvas'):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function printCanvas() {
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var canvas = document.getElementById('myCanvas');
    var epos = new epson.CanvasPrint(address);
    epos.onreceive = function (res) { alert(res.success); }
    epos.onerror = function (err) { alert(err.status); }
    epos.print(canvas);
}
</script>
```

To specify a print job ID from the application:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function printCanvas() {
    var printjobid = 'ABC123';
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var canvas = document.getElementById('myCanvas');
    var epos = new epson.CanvasPrint(address);
    epos.onreceive = function (res) { alert(res.success); }
    epos.onerror = function (err) { alert(err.status); }
    epos.print(canvas, printjobid);
}
</script>
```
getPrintJobStatus method

Acquires print job status. (ePOS-Print Service Ver.4.1 or later versions)

- The results of executing this API will be returned to an onreceive event (p.185).
- The latest print job status will be obtained when executing multiple printing processes with the same job ID.

Syntax

```plaintext
getPrintJobStatus(printjobid);
```

Parameter

- `printjobid` : (Required parameter, Object type : String)
  Specifies the print job ID to be verified.
  1 to 30 alphanumeric characters, including underscores, hyphens and periods, may be used.

Exception

<table>
<thead>
<tr>
<th>Exception</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter &quot;...&quot;) is invalid</td>
<td>Error</td>
</tr>
<tr>
<td>XMLHttpRequest is not supported</td>
<td>Error</td>
</tr>
</tbody>
</table>

Example

```javascript
var printjobid = '12345';
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi';
var epos = new epson.ePOSPrint(address);
epos.onreceive = function (res) { alert(res.success); };
epos.onerror = function (err) { alert(err.status); };
epos.getPrintJobStatus(printjobid);
```
open method

Enables status event operation.
Sends the status of the printer specified by the address property using an event.
Updates the status at the interval specified by the interval property.

Syntax

    open();

Example

```
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.onCoverOpen = function () {
        alert('coveropen');
    };

    function startMonitor() {
        epos.open();
    }

    function stopMonitor() {
        epos.close();
    }
    //</script>
```
close method

Disables status event operation.

**Syntax**

```
close();
```

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.CanvasPrint(address);
  epos.onCoverOpen = function () {
    alert('coveropen');
  };

  function startMonitor() {
    epos.open();
  }

  function stopMonitor() {
    epos.close();
  }
  //-->
</script>
```
recover method

Recovers from an error. (In ePOS-Print Service Ver.3.0 and later)
Recovers from errors that can be recovered from and clears the buffer.

Syntax

```
recover();
```

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.CanvasPrint(address);
epos.oncoveropen = function () {
    alert('coveropen');
};

function recover() {
    epos.recover();
}
//-->
</script>
```

reset method

Resets the printer. (In ePOS-Print Service Ver.3.0 and later)

Syntax

```
reset();
```

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.CanvasPrint(address);
epos.oncoveropen = function () {
    alert('coveropen');
};

function reset() {
    epos.reset();
}
//-->
</script>
```
## address property

URL of the printer.

**Object type**

String

**Description**

The URL of the printer to be used for printing is specified.

The URL is shown as follows:

```
http://[IP address of ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=[device ID of printer to be used for printing]&timeout=[timeout time]
```

The default value is the address specified by the constructor.

To acquire print job status, it is not necessary to specify "devid" and "timeout".

(For ePOS-Print Service Ver.4.1 or later and with the spooler enabled)

**Example**

```xml
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var epos = new epson.CanvasPrint();
epos.address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
epos.oncoveropen = function () { alert('coveropen'); };
epos.open();
//-->
</script>
```
**enabled property**

Retains the enabled/disabled setting for status event operation.

*Object type*
Boolean

*Description*
The enabled/disabled setting for status event operation is retained using a logical value. This is read-only. The default value is false.

*Example*

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.CanvasPrint(address);
  epos.oncoveropen = function () { alert('coveropen'); };
  epos.open();
  alert(epos.enabled);
  //-->
</script>
```
interval property

Specifies the interval of upgrading the status.

Object type
Number

Description

The interval of upgrading the status is specified in milliseconds.
Default value: 3000 (three seconds)
Minimum value: 1000 (one second or longer)
When an invalid value is specified, it is assumed to be 3000.

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
 <!--
 var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
 var epos = new epson.CanvasPrint(address);
 epos.interval = 1000;
 epos.oncoveropen = function () { alert('coveropen'); }; //-->
 epos.open();
</script>
```
status property

Status of the printer.

Object type

Number

Description

This is the status last obtained from the printer. This is read-only.
Default value: 0

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.CanvasPrint(address);
  epos.onoffline = function () {
    alert(epos.status);
  };
  epos.open();
//-->
</script>
```
## battery property

Battery status of the printer.

**Object type**
Number

### Description

Battery status obtained from the last printer status. This is read-only.
Default value: 0

### Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.CanvasPrint(address);
epos.onbatterylow = function () {
    alert(epos.battery);
};
epos.open();
//-->
</script>
```
timeout property

Specifies connection timeout.

Object type
Number

Description
Specifies connection timeout with ePOS-Print supported printer in milliseconds. When the transmission of print document by print method times out, onerror even is generated. Default value: 300000 (5 minutes)

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');

  var epos = new epson.CanvasPrint(address);
  epos.timeout = 60000;
  epos.onreceive = function (res) { alert(res.success); };
  epos.onerror = function (err) { alert(err.status); };
  epos.print(canvas);
</script>
```
## halftone property

Halftone processing method.

**Object type**

String

**Description**

The halftone processing method to be applied to monochrome (two-tone) printing is specified. The default value is HALFTONE_DITHER.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HALFTONE_DITHER</td>
<td>Dithering, suitable for printing graphics only.</td>
</tr>
<tr>
<td>HALFTONE_ERROR_DIFFUSION</td>
<td>Error diffusion, suitable for printing text and graphics together.</td>
</tr>
<tr>
<td>HALFTONE_THRESHOLD</td>
<td>Threshold, suitable for printing text only.</td>
</tr>
</tbody>
</table>

**Example**

To set the halftone type as error diffusion:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');

  var epos = new epson.CanvasPrint(address);
  epos.halftone = epos.HALFTONE_ERROR_DIFFUSION;
  epos.print(canvas);
  //-->
</script>
```
**brightness property**

Brightness correction value.

*Object type*

Number

*Description*

A gamma value in the range 0.1-10.0 is specified for the brightness correction value. The default value is 1.0.

*Example*

To set brightness as 2.2:

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
//
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var canvas = document.getElementById('myCanvas');

var epos = new epson.CanvasPrint(address);
epos.brightness = 2.2;
epos.print(canvas);
//-->
</script>
```
**cut property**

It sets with or without paper cut.

**Object type**

Boolean

**Description**

It specifies with or without paper cut.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true/1</td>
<td>Cut paper after printing</td>
</tr>
<tr>
<td>false/0 (Default)</td>
<td>Do not cut paper</td>
</tr>
</tbody>
</table>

**Example**

It sets paper cut after printing.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  //var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');
  var epos = new epson.CanvasPrint(address);
  epos.cut = true;
  epos.print(canvas);
  //-->
</script>
```
**mode property**

It sets the color mode.

**Object type**

String

**Description**

It specifies the color mode.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_MONO (Default)</td>
<td>Monochrome (2-tone)</td>
</tr>
<tr>
<td>MODE_GRAY16</td>
<td>Multiple tones (16-tone)</td>
</tr>
</tbody>
</table>

**Example**

Prints with multiple tones.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var canvas = document.getElementById('myCanvas');

var epos = new epson.CanvasPrint(address);
epos.mode = epos.MODE_GRAY16;
epos.print(canvas);
//-->
</script>
```
align property

It sets the position alignment.

Object type
String

Description

It specifies the position alignment.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALIGN_LEFT (Default)</td>
<td>Alignment to the left</td>
</tr>
<tr>
<td>ALIGN_CENTER</td>
<td>Alignment to the center</td>
</tr>
<tr>
<td>ALIGN_RIGHT</td>
<td>Alignment to the right</td>
</tr>
</tbody>
</table>

Example

Prints with center alignment.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');

  var epos = new epson.CanvasPrint(address);
  epos.align = epos.ALIGN_CENTER;
  epos.print(canvas);
  //</-->
</script>```


**color property**

It sets printing color.

**Object type**

String

**Description**

It specifies printing color.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLOR_NONE</td>
<td>No printing</td>
</tr>
<tr>
<td>COLOR_1</td>
<td>1st color</td>
</tr>
<tr>
<td>COLOR_2</td>
<td>2nd color</td>
</tr>
<tr>
<td>COLOR_3</td>
<td>3rd color</td>
</tr>
<tr>
<td>COLOR_4</td>
<td>4th color</td>
</tr>
</tbody>
</table>

**Example**

Prints with the 2nd color.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
var address = 'http://192.168.168.168/cgi-bin/epos/service.cgi?devid=local_printer';
var canvas = document.getElementById('myCanvas');

var epos = new epson.CanvasPrint(address);
epos.color = epos.COLOR_2;
epos.print(canvas);
//-->
</script>
```
**feed property**

It sets paper feed of label paper/black mark paper. (in ePOS-Print Service Ver.2.1 and later)

*Object type*

String

*Description*

Paper feed position of label paper/black mark paper.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEED_PEELED</td>
<td>Feeds to the peeling position.</td>
</tr>
<tr>
<td>FEED_CUTTING</td>
<td>Feeds to the cutting position.</td>
</tr>
<tr>
<td>FEED_CURRENT_TOF (Default)</td>
<td>Feeds to the top of the current label.</td>
</tr>
<tr>
<td>FEED_NEXT_TOF</td>
<td>Feeds to the top of the next label.</td>
</tr>
</tbody>
</table>

*Example*

After printing a label, it feeds paper to the peeling position.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
 <!--
 var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
 var canvas = document.getElementById('myCanvas');

 var epos = new epson.CanvasPrint(address);
 epos.paper = epos.PAPER_LABEL;
 epos.feed = epos.FEED_PEELED;
 epos.print(canvas);
 //-->

</script>
```
paper property

It sets paper type. (in ePOS-Print Service Ver.2.1 and later)

Object type
String

Description
It specifies paper type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPER_RECEIPT (Default)</td>
<td>Receipt (without black mark)</td>
</tr>
<tr>
<td>PAPER_RECEIPT_BM</td>
<td>Receipt (with black mark)</td>
</tr>
<tr>
<td>PAPER_LABEL</td>
<td>Die-cut label (without black mark)</td>
</tr>
<tr>
<td>PAPER_LABEL_BM</td>
<td>Die-cut label (with black mark)</td>
</tr>
</tbody>
</table>

Example
Prints a label.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var canvas = document.getElementById('myCanvas');

var epos = new epson.CanvasPrint(address);
epos.paper = epos.PAPER_LABEL;
epos.feed = epos.FEED_PEELING;
epos.print(canvas);
//-->
</script>```
layout property

It sets paper layout. (In ePOS-Print Service Ver.2.2 and later)

Object type
Object

Description
It specifies paper layout.

Property of layout being setup

<table>
<thead>
<tr>
<th>Property</th>
<th>Name</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Paper width</td>
<td>Number</td>
</tr>
<tr>
<td>height</td>
<td>Paper height</td>
<td>Number</td>
</tr>
<tr>
<td>margin_top</td>
<td>Top margin</td>
<td>Number</td>
</tr>
<tr>
<td>margin_bottom</td>
<td>Bottom margin</td>
<td>Number</td>
</tr>
<tr>
<td>offset_cut</td>
<td>Cutting position</td>
<td>Number</td>
</tr>
<tr>
<td>offset_label</td>
<td>Bottom position of label</td>
<td>Number</td>
</tr>
</tbody>
</table>

- Value of width (Object type : Number, When not specified : 580)
  Specifies paper width (in units of 0.1 mm). Specifies an integer from 290 to 600. *
- Value of height (Object type : Number, When not specified : 0)
  Specifies paper height (in units of 0.1 mm).

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>0 (auto)</td>
<td>Distance from the top of black mark to the top of next black mark</td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>284 to 1550 (manual)</td>
<td>Distance from the top of label to the top of next label</td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td></td>
<td>Distance from the bottom of black mark to the bottom of next black mark</td>
</tr>
</tbody>
</table>
- Value of margin_top (Object type : Number, When not specified : 0)
  Specifies top margin (in units of 0.1 mm).

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>-150 to 1500 *</td>
<td>Distance from the top of black mark</td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>0 to 1500 *</td>
<td>Distance from the top of label</td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td>-15 to 1500 *</td>
<td>Distance from the bottom of black mark</td>
</tr>
</tbody>
</table>

- Value of margin_bottom (Object type : Number, When not specified : 0)
  Specifies bottom margin (in units of 0.1 mm).

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>-15 to 0 *</td>
<td>Distance from the bottom of label (paper feed direction is a positive number)</td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td>-15 to 15 *</td>
<td>Distance from the top of black mark (paper feed direction is a positive number)</td>
</tr>
</tbody>
</table>

- Value of offset_cut (Object type : Number, When not specified : 0)
  Specifies cut position (in units of 0.1 mm).
  In case of die cut label paper, it is a distance from the bottom of label.
  When a paper has black mark, it is a distance from the beginning of black mark.

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>-290 to 50 *</td>
<td>Distance from the top of black mark to the top of next black mark</td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>0 to 50 *</td>
<td>Distance from the bottom of label to the cutting position</td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td>0 to 50 *</td>
<td>Distance from the top of black mark to the cutting position</td>
</tr>
</tbody>
</table>
- Value of offset_label (Object type: Number, When not specified: 0)
  Specifies label bottom position (sd) per 0.1 mm unit.

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Valid value range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receipt (without black mark)</td>
<td>0</td>
<td>Setup not necessary</td>
</tr>
<tr>
<td>Receipt (with black mark)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Die-cut label (without black mark)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Die-cut label (with black mark)</td>
<td>0 to 15 *</td>
<td>Distance from the top of black mark to the bottom of label</td>
</tr>
</tbody>
</table>

*: Valid value of range is depending on the printer model. For detail, refer to "Appendix - Printer Specifications".

Layout property positions that can be designated for each type of paper

![Diagram showing paper types and parameters](image)

<table>
<thead>
<tr>
<th>Mark</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>sf</td>
<td>width</td>
</tr>
<tr>
<td>sa</td>
<td>height</td>
</tr>
<tr>
<td>sb</td>
<td>margin_top</td>
</tr>
<tr>
<td>se</td>
<td>margin_bottom</td>
</tr>
<tr>
<td>sc</td>
<td>offset_cut</td>
</tr>
<tr>
<td>sd</td>
<td>offset_label</td>
</tr>
</tbody>
</table>
Example

To set 58 mm receipt (without black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');

  var epos = new epson.CanvasPrint(address);
  epos.paper = epos.PAPER_RECEIPT;
  epos.layout = { width: 580, height: 0, margin_top: 15, offset_cut: 0 };
  epos.cut = true;
  epos.print(canvas);
  //-->
</script>
```

To set 58 mm receipt (with black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');

  var epos = new epson.CanvasPrint(address);
  epos.paper = epos.PAPER_RECEIPT_BM;
  epos.layout = { width: 580, height: 0, margin_top: 15, offset_cut: 0 };
  epos.cut = true;
  epos.print(canvas);
  //-->
</script>
```

To set 58 mm die-cut label (without black mark):

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  <!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var canvas = document.getElementById('myCanvas');

  var epos = new epson.CanvasPrint(address);
  epos.paper = epos.PAPER_LABEL;
  epos.layout = { width: 580, height: 0, margin_top: 15, margin_bottom: -15, offset_cut: 25 };
  epos.cut = true;
  epos.print(canvas);
  //-->
</script>
```
To set 58 mm die-cut label (with black mark):

```javascript
<script type="text/javascript" src="/home/epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var canvas = document.getElementById('myCanvas');

    var epos = new epson.CanvasPrint(address);
    epos.paper = epos.PAPER_LABEL_BM;
    epos.layout = { width: 580, height: 0, margin_top: 15, margin_bottom: -15,
                   offset_cut: 25, offset_label: 15 };
    epos.cut = true;
    epos.print(canvas);
    //-->
</script>
```
onreceive event

This property registers the callback function and obtains a response message receipt event.

Syntax

Function (response)

Parameter of the callback function

Parameter: response (See “Properties of the response object” on page 185.)
Name: Response message
Object type: Object

Properties of the response object

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Name</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>success (p. 185)</td>
<td>Print result</td>
<td>Boolean</td>
</tr>
<tr>
<td>code (p. 185)</td>
<td>Error code</td>
<td>String</td>
</tr>
<tr>
<td>status (p. 186)</td>
<td>Status</td>
<td>Number</td>
</tr>
<tr>
<td>battery (p. 186)</td>
<td>Battery status</td>
<td>Number</td>
</tr>
<tr>
<td>printjobid (p. 186)</td>
<td>Print job ID</td>
<td>String</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported

• Value of success

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| true or 1      | • Printing succeeded  
 |                | • Processing succeeded (When the spooler function is enabled) |
| false or 0     | • Printing Failed  
 |                | • Processing Failed (When the spooler function is enabled) |

• Value of code

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'EPTR_AUTOMATICAL'</td>
<td>An automatically recoverable error occurred</td>
</tr>
<tr>
<td>'EPTR_BATTERY_LOW'</td>
<td>No remaining battery</td>
</tr>
<tr>
<td>'EPTR_COVER_OPEN'</td>
<td>A cover open error occurred</td>
</tr>
<tr>
<td>'EPTR_CUTTER'</td>
<td>An autocutter error occurred</td>
</tr>
<tr>
<td>'EPTR_MECHANICAL'</td>
<td>A mechanical error occurred</td>
</tr>
<tr>
<td>'EPTR_REC_EMPTY'</td>
<td>No paper in roll paper end sensor</td>
</tr>
<tr>
<td>'EPTR_UNRECOVERABLE'</td>
<td>An unrecoverable error occurred</td>
</tr>
<tr>
<td>'SchemaError'</td>
<td>The request document contains a syntax error</td>
</tr>
<tr>
<td>'DeviceNotFound'</td>
<td>The printer with the specified device ID does not exist</td>
</tr>
<tr>
<td>'PrintSystemError'</td>
<td>An error occurred on the printing system</td>
</tr>
<tr>
<td>'EX_BADPORT'</td>
<td>An error was detected on the communication port</td>
</tr>
<tr>
<td>'EX_TIMEOUT'</td>
<td>A print timeout occurred</td>
</tr>
<tr>
<td>'EX_SPOOLER'</td>
<td>There is not enough space available in the printing queue</td>
</tr>
<tr>
<td>'JobNotFound'</td>
<td>The specified job ID does not exist</td>
</tr>
<tr>
<td>'Printing'</td>
<td>Print job now printing</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported
- **Value of status**

<table>
<thead>
<tr>
<th>Constant (status)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASB_NO_RESPONSE</td>
<td>No response from the TM printer</td>
</tr>
<tr>
<td>ASB_PRINT_SUCCESS</td>
<td>Printing is successfully completed</td>
</tr>
<tr>
<td>ASB_DRAWER_KICK</td>
<td>Status of the 3rd pin of the drawer kick-out connector = &quot;H&quot;</td>
</tr>
<tr>
<td>ASB_BATTERY_OFFLINE</td>
<td>Off line status from remaining battery (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_OFF_LINE</td>
<td>Offline</td>
</tr>
<tr>
<td>ASBCOVER_OPEN</td>
<td>The cover is open</td>
</tr>
<tr>
<td>ASB_PAPER_FEED</td>
<td>Paper is being fed by a paper feed switch operation</td>
</tr>
<tr>
<td>ASB_WAIT_ON_LINE</td>
<td>Waiting to be brought back online</td>
</tr>
<tr>
<td>ASB_PANEL_SWITCH</td>
<td>The paper feed switch is being pressed (ON)</td>
</tr>
<tr>
<td>ASB_MECHANICAL_ERR</td>
<td>A mechanical error occurred</td>
</tr>
<tr>
<td>ASB_AUTOCUTTER_ERR</td>
<td>An autocutter error occurred</td>
</tr>
<tr>
<td>ASB_UNRECOVER_ERR</td>
<td>An unrecoverable error occurred</td>
</tr>
<tr>
<td>ASB_AUROCOVER_ERR</td>
<td>An automatically recoverable error occurred</td>
</tr>
<tr>
<td>ASB_RECEIPT_NEAR_END</td>
<td>No paper in roll paper near end sensor</td>
</tr>
<tr>
<td>ASB_RECEIPT_END</td>
<td>No paper in roll paper end sensor</td>
</tr>
<tr>
<td>ASB_BUZZER</td>
<td>A buzzer is on (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_WAIT_REMOVE_LABEL</td>
<td>Waiting period for removal of label (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_NO_LABEL</td>
<td>No paper in label peeling sensor (only for applicable devices)</td>
</tr>
<tr>
<td>ASB_SPOOLER_IS_STOPPED*</td>
<td>The spooler has stopped</td>
</tr>
</tbody>
</table>

* ePOS-Print Service Ver.4.1 or later versions supported

- **Value of battery**

0 is shown when the model doesn't have a battery installed.

- **Status of power**

<table>
<thead>
<tr>
<th>Value (battery)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x30XX</td>
<td>AC adapter is connected</td>
</tr>
<tr>
<td>0x31XX</td>
<td>AC adapter is not connected</td>
</tr>
</tbody>
</table>

- **Remaining battery**

<table>
<thead>
<tr>
<th>Value (battery)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xx36</td>
<td>Remaining battery 6</td>
</tr>
<tr>
<td>0xx35</td>
<td>Remaining battery 5</td>
</tr>
<tr>
<td>0xx34</td>
<td>Remaining battery 4</td>
</tr>
<tr>
<td>0xx33</td>
<td>Remaining battery 3</td>
</tr>
<tr>
<td>0xx32</td>
<td>Remaining battery 2</td>
</tr>
<tr>
<td>0xx31</td>
<td>Remaining battery 1 (Near end)</td>
</tr>
<tr>
<td>0xx30</td>
<td>Remaining battery 0 (Real end)</td>
</tr>
</tbody>
</table>

- **Value of printjobid**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;...&quot;</td>
<td>Print job ID (An empty string denotes that a print job ID has not been set)</td>
</tr>
</tbody>
</table>
Example

To print Canvas(ID=myCanvas):
To display the print result in a message box.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
function printCanvas() {
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var canvas = document.getElementById('myCanvas');

    var epos = new epson.CanvasPrint(address);
    epos.onreceive = function (res) {
        var success = res.success;
        var code = res.code;
        var status = res.status;
        alert(success);
    };
    epos.print(canvas);
}
</script>
```
**onerror event**

This property registers the callback function and obtains a communication error event.

**Syntax**

```
Function (error)
```

**Parameter of the callback function**

- **Parameter:** error (See “Properties of the error object” on page 188.)
- **Name:** Communication error information
- **Object type:** Object

**Properties of the error object**

<table>
<thead>
<tr>
<th>property</th>
<th>Name</th>
<th>Object type</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>HTTP status</td>
<td>Number</td>
</tr>
<tr>
<td>responseText</td>
<td>Response text</td>
<td>String</td>
</tr>
</tbody>
</table>

**Example**

To print Canvas(ID=myCanvas):
To display the HTTP status code in a message box when a communication error occurs.

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
function printCanvas() {
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var canvas = document.getElementById('myCanvas');

var epos = new epson.CanvasPrint(address);
epos.onerror = function (err) {
  var status = err.status;
  var text = err.responseText;
  alert(status);
};
epos.print(canvas);
//-->
</script>
```
onstatuschange event

Registers a callback function to obtain a status change event.

Syntax

Function (status)

Parameter of the callback function

Parameters: status
Name: Status
Object type: Number

Example

```javascript
<!--
var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
var epos = new epson.CanvasPrint(address);
epos.onstatuschange = function (status) {
    alert(status);
};
epos.open();
//-->
</script>
```
onbatterystatuschange event

Registers call back function and obtains battery status change event. (in ePOS-Print Service Ver.2.2 and later)

**Syntax**

Function (battery)

**Parameter of the callback function**

- **Parameters:** battery
- **Name:** Battery status
- **Object type:** Number

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.CanvasPrint(address);
  epos.onbatterystatuschange = function (battery) {
    alert(battery);
  };
  epos.open();
  //-->
</script>
```
onbatteryok event

Registers call back function and obtains remaining battery event. (in ePOS-Print Service Ver.2.2 and later)

**Syntax**

```javascript
Function ()
```

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
   var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
   var epos = new epson.CanvasPrint(address);
   epos.onbatteryok = function () {
      alert('batteryok');
   };
   epos.open();
   //-->
</script>
```

onbatterylow event

Registers call back function and obtains no remaining battery event. (in ePOS-Print Service Ver.2.2 and later)

**Syntax**

```javascript
Function ()
```

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
   var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
   var epos = new epson.CanvasPrint(address);
   epos.onbatterylow = function () {
      alert('batterylow');
   };
   epos.open();
   //-->
</script>
```
## ononline event

Registers a callback function to obtain an online event.

**Object type**

Function

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.CanvasPrint(address);
  epos.ononline = function () {
      alert('online');
  }
  epos.open();
//--></script>
```

## onoffline event

Registers a callback function to obtain an offline event.

**Object type**

Function

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
  var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
  var epos = new epson.CanvasPrint(address);
  epos.onoffline = function () {
      alert('offline');
  }
  epos.open();
//--></script>
```
**onpoweroff event**

Registers a callback function to obtain a non-response event.

*Object type*

Function ()

*Example*

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.onpoweroff = function () {
        alert('poweroff');
    };
    epos.open();
    //-->
</script>
```

**oncoverok event**

Registers a callback function to obtain a cover close event.

*Object type*

Function ()

*Example*

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.oncoverok = function () {
        alert('coverok');
    };
    epos.open();
    //-->
</script>
```
oncoveropen event

Registers a callback function to obtain a cover open event.

**Object type**
Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.oncoveropen = function () {
        alert('coveropen');
    };
    epos.open();
    //-->
</script>
```

onpaperok event

Registers a callback function to obtain a paper remaining event.

**Object type**
Function ()

**Example**

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.onpaperok = function () {
        alert('paperok');
    };
    epos.open();
    //-->
</script>
```
onpapernearend event

Registers a callback function to obtain a paper near end event.

Object type
Function ()

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.onpapernearend = function () {
        alert('papernearend');
    };
    epos.open();
//--> </script>
```

onpaperend event

Registers a callback function to obtain a paper end event.

Object type
Function ()

Example

```javascript
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
<!--
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.onpaperend = function () {
        alert('paperend');
    };
    epos.open();
//--> </script>
```
**ondrawerclosed event**

Registers a callback function to obtain a drawer close event.

*Object type*

Function ()

*Example*

```html
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.ondrawerclosed = function () {
        alert('drawerclosed');
    };
    epos.open();
</script>
```

**ondraweropen event**

Registers a callback function to obtain a drawer open event.

*Object type*

Function ()

*Example*

```html
<script type="text/javascript" src="epos-print-4.x.x.js"></script>
<script type="text/javascript">
    var address = 'http://192.168.192.168/cgi-bin/epos/service.cgi?devid=local_printer';
    var epos = new epson.CanvasPrint(address);
    epos.ondraweropen = function () {
        alert('draweropen');
    };
    epos.open();
</script>
```
ePOS-Print Editor

This section describes how to use ePOS-Print Editor included in the contents in the package. This tool allows you to create an ePOS-Print API (p. 61) sample code as you like. Use this tool for your Web application development.

ePOS-Print Editor Operating Environment

- Web Browser
  - Windows Internet Explorer 9 or later
  - Mozilla Firefox 13 or later
  - Google Chrome 19 or later
  - Apple Safari 5.1.7 or later
  - iPad Safari in iOS 5.1 or later

Displaying ePOS-Print Editor

- If opening a page with ePOS-Print Editor's HTML file placed on the local disk, some functionality does not operate due to your Web browser's security policy. Place the HTML file of ePOS Editor to a folder under Web server.
- In Google Chrome, when a preview image including pictures is displayed, a "SECURITY_ERR: DOM Exception 18" error occurs.
- In Windows Internet Explorer 9, when printing is performed, a "SCRIPT5: Access is denied." error occurs.

1. Open the following URL page using the Web browser. http://[Web server IP address]/editor/index.html

2. ePOS-Print Editor appears.
Perform the print setting and the preview setting. Select the (Setting) tab to display the Setting screen.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model, Paper width (Liner width)</td>
<td>Select printer model to be used and paper width (for label paper, the mount width). The Preview screen resizes according to the paper width set to the model.</td>
</tr>
<tr>
<td>IP address of ePOS-Print supported TM printer</td>
<td>Specify the IP address of the printer. Be sure to specify this item.</td>
</tr>
<tr>
<td>Device ID of the target printer</td>
<td>Specify the device ID of the printer. Be sure to specify this item.</td>
</tr>
<tr>
<td>Print timeout (milliseconds)</td>
<td>Specify the print timeout time in milliseconds. The maximum value is 60000 (60 seconds).</td>
</tr>
<tr>
<td>Monitor the status</td>
<td>When this checkbox is checked, the printer's status is monitored.</td>
</tr>
<tr>
<td>Use print job ID</td>
<td>Check this for a print job ID to be assigned when printing.</td>
</tr>
</tbody>
</table>
Creating a Sample Code

Select the [Edit] tab to display the Edit screen. Create an ePOS-Print API sample code in the Edit screen.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu area</td>
<td>Displays the available functions. Click an icon to add it to the bottom of the edit area, and drag an icon to insert it anywhere in the edit area.</td>
</tr>
<tr>
<td>Edit area</td>
<td>Displays the functions selected in the menu area. Drag an element to change its position. An element can be deleted using the x button located on its right side.</td>
</tr>
<tr>
<td>Import</td>
<td>Using ePOS-Print XML, ePOS-Print Editor can import XML data stored in the past. For details, refer to Import (p.202).</td>
</tr>
<tr>
<td>Clear</td>
<td>Deletes the edited details.</td>
</tr>
<tr>
<td>Forced Transmission</td>
<td>Sets forced transmission mode.</td>
</tr>
</tbody>
</table>
Create a sample code as follows:

1. Click an icon in the menu area to add an element in the edit area. The position of the added function can be changed by dragging.

2. Configure the added element. Example: When the NV logo is added, set the key code.

3. Select the (Preview) tab to check the preview image. When a printer is connected, you can also check the image by printing. For details, refer to Print (p.201).

   - Logo printing, barcode printing, 2D code printing, ESC command, buzzer sound, drawer kick, and paper cut are displayed as icons.
   - The layout may change depending on the preview settings. (For details, refer to Setting (p.198)).

4. Select the (API) tab. The ePOS-Print API sample code appears. Use it by copying.

   The ePOS-Print XML print document is used for importing. If necessary, select the XML tab and save the content of the ePOS-Print XML print document by copying.
Print

Using the printer, print the print document according to the printer's settings to perform test printing. (For details on the printer settings, refer to Setting (p.198)).

Perform printing as follows:

1. Select the (Print) tab.

2. Check the content in the (Data to be sent (ePOS-Print XML)) box and press the (Send) button.
   The ePOS-Print XML print document created using the (Edit) tab page is displayed in "Data to be sent (ePOS-Print XML)."

3. The print document is printed to ePOS-Print supported TM printer. The acquired status is displayed in the Information box.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data to be sent</td>
<td>The ePOS-Print XML document is displayed.</td>
</tr>
<tr>
<td>(ePOS-Print XML)</td>
<td></td>
</tr>
<tr>
<td>Send</td>
<td>Sends data to the printer and performs printing.</td>
</tr>
<tr>
<td>Information</td>
<td>Displays the print status.</td>
</tr>
<tr>
<td>Clear</td>
<td>Deletes the content in the (Information) box.</td>
</tr>
</tbody>
</table>
ePOS-Print Editor can import and re-edit the ePOS-Print XML print document once created.

Note that you cannot perform import operation using ePOS-Print API source code. Perform import operation using ePOS-Print XML print data.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data to be import (ePOS-Print XML)</td>
<td>Paste and check the ePOS-Print XML print document to be imported.</td>
</tr>
<tr>
<td>Apply</td>
<td>Imports the ePOS-Print XML print document.</td>
</tr>
<tr>
<td>Information</td>
<td>Displays the import information.</td>
</tr>
<tr>
<td>Close</td>
<td>Closes the import screen.</td>
</tr>
</tbody>
</table>

ePOS-Print Editor can import an ePOS-Print XML print document as follows:

1. Select the (Edit) tab and click the (Import) button.
2. The "Import" screen appears. Paste the ePOS-Print XML print document in the (Data to be import (ePOS-Print XML)) box.
3. Click the (Apply) button.
4. The "Confirmation" screen appears. Click the (Yes) button.
## Appendix

### Printer specifications

<table>
<thead>
<tr>
<th></th>
<th>Receipt</th>
<th>Die-cut label</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface</strong></td>
<td>Ethernet, Wireless LAN</td>
<td></td>
</tr>
<tr>
<td><strong>Resolution</strong></td>
<td>203 dpi x 203 dpi (W x H)</td>
<td></td>
</tr>
<tr>
<td><strong>Print Width</strong></td>
<td>256 dots (38 mm) to 576 dots (80 mm)</td>
<td>224 dots (38 mm) to 560 dots (80 mm)</td>
</tr>
<tr>
<td><strong>Font</strong></td>
<td>Font A, Font B, Font C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td><strong>Characters in a Line</strong></td>
<td>Font A</td>
<td>ANK: 48 characters,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ANK: 46 characters</td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>ANK: 57 characters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ANK: 56 characters</td>
</tr>
<tr>
<td></td>
<td>Font C</td>
<td>ANK: 72 characters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ANK: 70 characters</td>
</tr>
<tr>
<td><strong>Character Size</strong></td>
<td>Font A</td>
<td>ANK: 12 dots x 24 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ANK: 10 dots x 24 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>Font C</td>
<td>ANK: 8 dots x 16 dots (W x H)</td>
</tr>
<tr>
<td><strong>Character Baseline</strong></td>
<td>Font A</td>
<td>At the 21st dot from the top of the character</td>
</tr>
<tr>
<td></td>
<td></td>
<td>At the 21st dot from the top of the character</td>
</tr>
<tr>
<td></td>
<td>Font C</td>
<td>At the 15st dot from the top of the character</td>
</tr>
<tr>
<td><strong>Default Line Feed Space</strong></td>
<td>30 dots</td>
<td></td>
</tr>
<tr>
<td><strong>Color Specification</strong></td>
<td>First color</td>
<td></td>
</tr>
<tr>
<td></td>
<td>First color, Second color (when two-color printing is set)</td>
<td></td>
</tr>
<tr>
<td><strong>Raster Image</strong></td>
<td>Monochrome image, Two color image</td>
<td></td>
</tr>
<tr>
<td><strong>Logo</strong></td>
<td>Monochrome image, Two color image</td>
<td>(To perform two-color printing, change the settings of the printer using the memory switch setting utility.)</td>
</tr>
<tr>
<td><strong>Bar Code</strong></td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128</td>
<td></td>
</tr>
<tr>
<td><strong>Two-Dimensional Code</strong></td>
<td>PDF417, QR Code, MaxiCode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Receipt</td>
<td>Die-cut label</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>576 dots x 738 dots (W x H)</td>
<td>560 dots x 738 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>576 dots x 369 dots (W x H)</td>
<td>560 dots x 369 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>576 dots x 1476 dots (W x H)</td>
<td>560 dots x 1476 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>576 dots x 738 dots (W x H)</td>
<td>560 dots x 738 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Rectangle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
<td></td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported (With automatic setup mode)</td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
</tr>
</tbody>
</table>
## TM-T70-i/ TM-T70

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>80 mm</strong></td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
</tr>
<tr>
<td>Resolution</td>
<td>180 dpi x 180 dpi (W x H)</td>
</tr>
<tr>
<td>Print Width</td>
<td>512 dots</td>
</tr>
</tbody>
</table>
| Font                                         | Font A, Font B  
For more information about what character codes can be printed, refer to the user's manual that came with the printer. |
| Characters in a Line                         | Font A ANK: 42 characters  
Font B ANK: 56 characters                                                      |
| Character Size                               | Font A ANK: 12 dots x 24 dots (W x H)  
Font B ANK: 9 dots x 17 dots (W x H)                                           |
| Character Baseline                           | Font A At the 21st dot from the top of the character  
Font B At the 15th dot from the top of the character                           |
| Default Line Feed Space                      | 30 dots                                                                      |
| Color Specification                          | First color                                                                  |
| Page Mode Default Area                       | 512 dots x 1662 dots (W x H)                                                  |
| Page Mode Maximum Area                       | 512 dots x 1662 dots (W x H)                                                  |
| Raster image                                 | Monochrome image                                                             |
| Logo                                         | Monochrome image                                                             |
| Bar Code                                      | UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128 |
| Two-Dimensional Code                         | PDF417, QR Code                                                              |
| Ruled Line                                   | Not supported                                                                |
| Paper Cut                                    | Cut, Feed cut                                                                |
| Specification of the paper feed position for | Not supported                                                                |
| labels / black mark paper                    |                                                                              |
| Drawer Kick-Out                              | Supported                                                                    |
| Buzzer                                       | Not supported                                                                |
| Paper Layout Settings                        | Not supported                                                                |
| Forced transmission mode                     | Supported                                                                    |
| Recovery from an error                       | Supported                                                                    |
| Reset                                        | Supported                                                                    |
| Command                                      | Supported                                                                    |
## TM-T70-i (Multi-language model)

<table>
<thead>
<tr>
<th>Feature</th>
<th>80 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
</tr>
<tr>
<td>Resolution</td>
<td>203 dpi x 203 dpi (W x H)</td>
</tr>
<tr>
<td>Print Width</td>
<td>576 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B</td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>Font A: ANK: 48 characters</td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 64 characters</td>
</tr>
<tr>
<td>Character Size</td>
<td>Font A: ANK: 12 dots x 24 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 9 dots x 17 dots (W x H)</td>
</tr>
<tr>
<td>Character Baseline</td>
<td>Font A: At the 21st dot from the top of the character</td>
</tr>
<tr>
<td></td>
<td>Font B: At the 15th dot from the top of the character</td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td>30 dots</td>
</tr>
<tr>
<td>Color Specification</td>
<td>First color</td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>576 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>576 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td>Raster image</td>
<td>Monochrome image</td>
</tr>
<tr>
<td>Logo</td>
<td>Monochrome image</td>
</tr>
<tr>
<td>Bar Code</td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128</td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td>PDF417, QR Code</td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td>Not supported</td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
</tr>
<tr>
<td>Buzzer</td>
<td>Not supported</td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
</tr>
</tbody>
</table>
## TM-T88V-i/ TM-T88V

<table>
<thead>
<tr>
<th></th>
<th>80 mm</th>
<th>58 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>180 dpi x 180 dpi (W x H)</td>
<td></td>
</tr>
<tr>
<td>Print Width</td>
<td>512 dots</td>
<td>360 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>Font A: ANK: 42 characters</td>
<td>ANK: 30 characters</td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 56 characters</td>
<td>ANK: 40 characters</td>
</tr>
<tr>
<td>Character Size</td>
<td>Font A: ANK: 12 dots x 24 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 9 dots x 17 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td>Character Baseline</td>
<td>Font A: At the 21st dot from the top of the character</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B: At the 16th dot from the top of the character</td>
<td></td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td>30 dots</td>
<td></td>
</tr>
<tr>
<td>Color Specification</td>
<td>First color</td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>512 dots x 831 dots (W x H)</td>
<td>360 dots x 831 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>512 dots x 1662 dots (W x H)</td>
<td>360 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td>Raster image</td>
<td>Monochrome image, two-color image</td>
<td></td>
</tr>
<tr>
<td>Logo</td>
<td>Monochrome image, two-color image</td>
<td></td>
</tr>
<tr>
<td>Bar Code</td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, IIF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded</td>
<td></td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td>PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked (Composite Symbology not supported)</td>
<td></td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
<td></td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Optional (Pattern A ~ Pattern E, Error, No paper, Stop)</td>
<td></td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>80 mm</td>
<td>58 mm</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
</tr>
</tbody>
</table>
# Appendix

## TM-P60II

<table>
<thead>
<tr>
<th>feature</th>
<th>Receipt 58 mm</th>
<th>Receipt 60 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Wireless LAN</td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>203 dpi x 203 dpi (W x H)</td>
<td></td>
</tr>
<tr>
<td>Print Width</td>
<td>420 dots</td>
<td>432 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B, Font C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>Font A: ANK: 35 characters,</td>
<td>ANK: 36 characters</td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 42 characters</td>
<td>ANK: 43 characters</td>
</tr>
<tr>
<td></td>
<td>Font C: ANK: 52 characters</td>
<td>ANK: 54 characters</td>
</tr>
<tr>
<td>Character Size</td>
<td>Font A: ANK: 12 dots x 24 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 10 dots x 24 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font C: ANK: 8 dots x 16 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td>Character Baseline</td>
<td>Font A: At the 21st dot from the top of the character</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B: At the 21st dot from the top of the character</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font C: At the 15th dot from the top of the character</td>
<td></td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td>30 dots</td>
<td></td>
</tr>
<tr>
<td>Color Specification</td>
<td>First color</td>
<td></td>
</tr>
<tr>
<td>Raster Image</td>
<td>Monochrome image</td>
<td></td>
</tr>
<tr>
<td>Logo</td>
<td>Monochrome image</td>
<td></td>
</tr>
<tr>
<td>Bar Code</td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded</td>
<td></td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td>PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, DataMatrix (Composit Symbology: Not supported)</td>
<td></td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>420 dots x 1624 dots (W x H)</td>
<td>432 dots x 1624 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>420 dots x 1624 dots (W x H)</td>
<td>432 dots x 1624 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Line</td>
<td>Supported (Only solid line)</td>
<td></td>
</tr>
<tr>
<td>Page Mode Rectangle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
<td></td>
</tr>
<tr>
<td>Specification</td>
<td>Receipt 58 mm</td>
<td>Receipt 60 mm</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Support (Pattern 1 ~ Pattern 10, Stop)</td>
<td></td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
</tr>
</tbody>
</table>
## TM-P60II with Peeler

<table>
<thead>
<tr>
<th></th>
<th>Receipt 58 mm</th>
<th>Receipt 60 mm</th>
<th>Die-cut label</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface</strong></td>
<td>Wireless LAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Resolution</strong></td>
<td>203 dpi x 203 dpi (W x H)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Print Width</strong></td>
<td>420 dots</td>
<td>432 dots</td>
<td>160 dots – 400 dots</td>
</tr>
<tr>
<td><strong>Font</strong></td>
<td>Font A, Font B, Font C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For more information about what character codes can be printed, refer to the user’s manual that came with the printer.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Characters in a Line</strong></td>
<td>Font A ANK: 35 characters, ANK: 36 characters ANK: 33 characters</td>
<td>Font B ANK: 42 characters ANK: 43 characters ANK: 40 characters</td>
<td>Font C ANK: 52 characters ANK: 54 characters ANK: 50 characters</td>
</tr>
<tr>
<td><strong>Character Size</strong></td>
<td>Font A ANK: 12 dots x 24 dots (W x H)</td>
<td>Font B ANK: 10 dots x 24 dots (W x H)</td>
<td>Font C ANK: 8 dots x 16 dots (W x H)</td>
</tr>
<tr>
<td><strong>Character Baseline</strong></td>
<td>Font A At the 21st dot from the top of the character</td>
<td>Font B At the 21st dot from the top of the character</td>
<td>Font C At the 15th dot from the top of the character</td>
</tr>
<tr>
<td><strong>Default Line Feed Space</strong></td>
<td>30 dots</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Color Specification</strong></td>
<td>First color</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Raster Image</strong></td>
<td>Monochrome image</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Logo</strong></td>
<td>Monochrome image</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bar Code</strong></td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Two-Dimensional Code</strong></td>
<td>PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, DataMatrix (Composit Symbology : Not supported)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Ruled Line</strong></td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Page Mode Default Area</strong></td>
<td>420 dots x 1624 dots (W x H)</td>
<td>432 dots x 1624 dots (W x H)</td>
<td>400 dots x 1624 dots (W x H)</td>
</tr>
<tr>
<td><strong>Page Mode Maximum Area</strong></td>
<td>420 dots x 1624 dots (W x H)</td>
<td>432 dots x 1624 dots (W x H)</td>
<td>400 dots x 1624 dots (W x H)</td>
</tr>
<tr>
<td><strong>Page Mode Line</strong></td>
<td>Supported (Only solid line)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Paper Layout

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Receipt 58 mm</th>
<th>Receipt 60 mm</th>
<th>Die-cut label paper (without black mark)</th>
<th>Die-cut label paper (with black mark)</th>
</tr>
</thead>
<tbody>
<tr>
<td>width (sf)</td>
<td>290 to 600</td>
<td>290 to 600</td>
<td>290 to 600</td>
<td>290 to 600</td>
</tr>
<tr>
<td>height (sa)</td>
<td>0</td>
<td>0, 284 to 1550</td>
<td>0, 284 to 1550</td>
<td>0, 284 to 1550</td>
</tr>
<tr>
<td>margin_top (sb)</td>
<td>0</td>
<td>-130 to 1500</td>
<td>0 to 1500</td>
<td>-15 to 1500</td>
</tr>
<tr>
<td>margin_bottom (se)</td>
<td>0</td>
<td>0</td>
<td>-15 to 0</td>
<td>-15 to 15</td>
</tr>
<tr>
<td>offset_cut (sc)</td>
<td>0</td>
<td>-256 to 50</td>
<td>0 to 50</td>
<td>0 to 50</td>
</tr>
<tr>
<td>offset_label (sd)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 to 15</td>
</tr>
</tbody>
</table>
## TM-P80

<table>
<thead>
<tr>
<th><strong>Receipt 80 mm</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface</strong></td>
<td>Wireless LAN</td>
</tr>
<tr>
<td><strong>Resolution</strong></td>
<td>203 dpi x 203 dpi (W x H)</td>
</tr>
<tr>
<td><strong>Print Width</strong></td>
<td>576 dots, 546 dots (42 column mode)</td>
</tr>
<tr>
<td><strong>Font</strong></td>
<td>Font A, Font B</td>
</tr>
<tr>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td><strong>Characters in a Line</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Font A</strong></td>
<td>ANK: 48 characters,</td>
</tr>
<tr>
<td><strong>Font B</strong></td>
<td>ANK: 64 characters</td>
</tr>
<tr>
<td><strong>Font A</strong> (42 column mode)</td>
<td>ANK: 42 characters</td>
</tr>
<tr>
<td><strong>Font B</strong> (42 column mode)</td>
<td>ANK: 60 characters</td>
</tr>
<tr>
<td><strong>Character Size</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Font A</strong></td>
<td>ANK: 12 dots x 24 dots (W x H)</td>
</tr>
<tr>
<td><strong>Font B</strong></td>
<td>ANK: 9 dots x 17 dots (W x H)</td>
</tr>
<tr>
<td><strong>Font A</strong> (42 column mode)</td>
<td>ANK: 13 dots x 24 dots (W x H)</td>
</tr>
<tr>
<td><strong>Font B</strong> (42 column mode)</td>
<td>ANK: 9 dots x 17 dots (W x H)</td>
</tr>
<tr>
<td><strong>Character Baseline</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Font A</strong></td>
<td>At the 21st dot from the top of the character</td>
</tr>
<tr>
<td><strong>Font B</strong></td>
<td>At the 15th dot from the top of the character</td>
</tr>
<tr>
<td><strong>Default Line Feed Space</strong></td>
<td>3.75 mm (0.15&quot;)</td>
</tr>
<tr>
<td><strong>Color Specification</strong></td>
<td>First color</td>
</tr>
<tr>
<td><strong>Raster Image</strong></td>
<td>Monochrome image</td>
</tr>
<tr>
<td><strong>Logo</strong></td>
<td>Monochrome image</td>
</tr>
<tr>
<td><strong>Bar Code</strong></td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded</td>
</tr>
</tbody>
</table>
Two-Dimensional Code | PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, DataMatrix
---|---
Ruled Line | Not supported
Page Mode Default Area | 576 dots x 1662 dots (W x H), 546 dots x 1662 dots (W x H) (42 column mode)
Page Mode Maximum Area | 576 dots x 1662 dots (W x H), 546 dots x 1662 dots (W x H) (42 column mode)
Page Mode | Line Supported (Only solid line)
Rectangle | Supported
Paper Cut | Feed cut (Feeds paper to cutting position)
Specification of the paper feed position for labels / black mark paper | Supported (Only for black mark paper)
Drawer Kick-Out | Not supported
Buzzer | Support (Pattern 1 ~ Pattern 10, Stop)
Paper Layout Settings | Supported (Only for receipt paper)
Forced transmission mode | Supported
Recovery from an error | Supported
Reset | Supported
Command | Supported

**Paper Layout**

<table>
<thead>
<tr>
<th>Paper type</th>
<th>Receipt paper (without black mark)</th>
<th>Receipt paper (with black mark)</th>
</tr>
</thead>
<tbody>
<tr>
<td>width (sf)</td>
<td>800</td>
<td>800</td>
</tr>
<tr>
<td>height (sa)</td>
<td>0</td>
<td>0, 284 to 3100</td>
</tr>
<tr>
<td>margin_top (sb)</td>
<td>0</td>
<td>-98 to 3100</td>
</tr>
<tr>
<td>margin_bottom (se)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>offset_cut (sc)</td>
<td>0</td>
<td>-173 to 50</td>
</tr>
<tr>
<td>offset_label (sd)</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
## TM-T20

<table>
<thead>
<tr>
<th>Feature</th>
<th>58 mm</th>
<th>80 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface</strong></td>
<td>Ethernet</td>
<td></td>
</tr>
<tr>
<td><strong>Resolution</strong></td>
<td>203 dpi x 203 dpi (W x H)</td>
<td></td>
</tr>
<tr>
<td><strong>Print Width</strong></td>
<td>420 dots</td>
<td>576 dots</td>
</tr>
<tr>
<td><strong>Font</strong></td>
<td>Font A, Font B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td><strong>Characters in a Line</strong></td>
<td>Font A</td>
<td>ANK: 35 characters,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ANK: 48 characters</td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>ANK: 46 characters,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ANK: 64 characters</td>
</tr>
<tr>
<td><strong>Character Size</strong></td>
<td>Font A</td>
<td>ANK: 12 dots x 24 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>ANK: 9 dots x 17 dots (W x H)</td>
</tr>
<tr>
<td><strong>Character Baseline</strong></td>
<td>Font A</td>
<td>At the 21st dot from the top of the character</td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>At the 16th dot from the top of the character</td>
</tr>
<tr>
<td><strong>Default Line Feed Space</strong></td>
<td></td>
<td>30 dots</td>
</tr>
<tr>
<td><strong>Color Specification</strong></td>
<td></td>
<td>First color</td>
</tr>
<tr>
<td><strong>Raster Image</strong></td>
<td></td>
<td>Monochrome image</td>
</tr>
<tr>
<td><strong>Logo</strong></td>
<td></td>
<td>Monochrome image</td>
</tr>
<tr>
<td><strong>Bar Code</strong></td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded</td>
<td></td>
</tr>
<tr>
<td><strong>Two-Dimensional Code</strong></td>
<td>PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked (Composit Symbology : Not supported)</td>
<td></td>
</tr>
<tr>
<td><strong>Ruled Line</strong></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td><strong>Page Mode Default Area</strong></td>
<td>420 dots x 831 dots (W x H)</td>
<td>576 dots x 831 dots (W x H)</td>
</tr>
<tr>
<td><strong>Page Mode Maximum Area</strong></td>
<td>420 dots x 1662 dots (W x H)</td>
<td>576 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td><strong>Page Mode</strong></td>
<td>Line</td>
<td>Not supported</td>
</tr>
<tr>
<td></td>
<td>Rectangle</td>
<td></td>
</tr>
<tr>
<td><strong>Paper Cut</strong></td>
<td></td>
<td>Cut, Feed cut</td>
</tr>
<tr>
<td><strong>Specification of the paper feed position for labels / black mark paper</strong></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>58 mm</td>
<td>80 mm</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Optional (Pattern A – Pattern E, Error, No paper, Stop)</td>
<td></td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
</tr>
</tbody>
</table>
## TM-T70 (Multi-language model)

<table>
<thead>
<tr>
<th></th>
<th>80 mm</th>
<th>58 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>203 dpi x 203 dpi (W x H)</td>
<td></td>
</tr>
<tr>
<td>Print Width</td>
<td>576 dots</td>
<td>416 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>Font A: ANK: 42 characters</td>
<td>ANK: 34 characters</td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 56 characters</td>
<td>ANK: 46 characters</td>
</tr>
<tr>
<td>Character Size</td>
<td>Font A: ANK: 12 dots x 24 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B: ANK: 9 dots x 17 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td>Character Baseline</td>
<td>Font A: At the 21st dot from the top of the character</td>
<td>Font B: At the 15th dot from the top of the character</td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td>30 dots</td>
<td></td>
</tr>
<tr>
<td>Color Specification</td>
<td>First color</td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>576 dots x 1662 dots (W x H)</td>
<td>416 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>576 dots x 1662 dots (W x H)</td>
<td>416 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td>Raster image</td>
<td>Monochrome image</td>
<td></td>
</tr>
<tr>
<td>Logo</td>
<td>Monochrome image</td>
<td></td>
</tr>
<tr>
<td>Bar Code</td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128</td>
<td></td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td>PDF417, QR Code</td>
<td></td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
<td></td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
</tr>
</tbody>
</table>
## TM-T88IV

<table>
<thead>
<tr>
<th></th>
<th>80 mm</th>
<th>58 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>180 dpi x 180 dpi (W x H)</td>
<td></td>
</tr>
<tr>
<td>Print Width</td>
<td>512 dots</td>
<td>360 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>Font A ANK: 42 characters ANK: 30 characters</td>
<td>Font B ANK: 56 characters ANK: 40 characters</td>
</tr>
<tr>
<td>Character Size</td>
<td>Font A ANK: 12 dots x 24 dots (W x H)</td>
<td>Font B ANK: 9 dots x 17 dots (W x H)</td>
</tr>
<tr>
<td>Character Baseline</td>
<td>Font A At the 21st dot from the top of the character</td>
<td>Font B At the 16th dot from the top of the character</td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td>30 dots</td>
<td></td>
</tr>
<tr>
<td>Color Specification</td>
<td>First color, First color, Second color (when two-color printing is set)</td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>512 dots x 831 dots (W x H)</td>
<td>360 dots x 831 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>when two-color printing is set 512 dots x 415 dots (W x H)</td>
<td>360 dots x 415 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>512 dots x 1662 dots (W x H)</td>
<td>360 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>when two-color printing is set 512 dots x 831 dots (W x H)</td>
<td>360 dots x 831 dots (W x H)</td>
</tr>
<tr>
<td>Raster image</td>
<td>Monochrome image, two-color image</td>
<td></td>
</tr>
<tr>
<td>Logo</td>
<td>Monochrome image, two-color image</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(To perform two-color printing, change the settings of the printer using the memory switch setting utility.)</td>
<td></td>
</tr>
<tr>
<td>Bar Code</td>
<td>UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128</td>
<td></td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td>PDF417, QR Code</td>
<td></td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
<td></td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>80 mm</td>
<td>58 mm</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
</tr>
</tbody>
</table>
## TM-T90

<table>
<thead>
<tr>
<th>Feature</th>
<th>58 mm</th>
<th>60 mm</th>
<th>80 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>180 dpi x 180 dpi (W x H)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print Width</td>
<td>360 dots</td>
<td>384 dots</td>
<td>512 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>ANK: 30 characters</td>
<td>ANK: 32 characters</td>
<td>ANK: 42 characters</td>
</tr>
<tr>
<td>Font A</td>
<td>ANK: 40 characters</td>
<td>ANK: 42 characters</td>
<td>ANK: 56 characters</td>
</tr>
<tr>
<td>Font B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Character Size</td>
<td>ANK: 12 dots x 24 dots (W x H)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Character Baseline</td>
<td>ANK: 9 dots x 17 dots (W x H)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Font A</td>
<td>At the 21st dot from the top of the character</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Font B</td>
<td>At the 16th dot from the top of the character</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td>30 dots</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Color Specification</td>
<td>First color</td>
<td>First color, Second color (when two-color printing is set)</td>
<td></td>
</tr>
<tr>
<td>Raster Image</td>
<td>Monochrome image, Two color image</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logo</td>
<td>Monochrome image, Two color image</td>
<td>To perform two-color printing, change the settings of the printer using the memory switch setting utility.</td>
<td></td>
</tr>
<tr>
<td>Bar Code</td>
<td>UPC-A, UPC-E, EAN13, JAN13, JAN8, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td>PDF417</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ruled Line</td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td>360 dots x 831 dots (W x H)</td>
<td>384 dots x 831 dots (W x H)</td>
<td>512 dots x 831 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>when two-color printing is set</td>
<td>360 dots x 415 dots (W x H)</td>
<td>384 dots x 415 dots (W x H)</td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td>360 dots x 1662 dots (W x H)</td>
<td>384 dots x 1662 dots (W x H)</td>
<td>512 dots x 1662 dots (W x H)</td>
</tr>
<tr>
<td></td>
<td>when two-color printing is set</td>
<td>360 dots x 831 dots (W x H)</td>
<td>384 dots x 831 dots (W x H)</td>
</tr>
<tr>
<td>Feature</td>
<td>58 mm</td>
<td>60 mm</td>
<td>80 mm</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Page Mode</td>
<td>Line</td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rectangle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specification of the paper feed position for</td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>labels / black mark paper</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td>Cut, Feed cut</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td>Supported via Drawer Kick-Out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## TM-U220

<table>
<thead>
<tr>
<th></th>
<th>76 mm</th>
<th>70 mm</th>
<th>58 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Ethernet, Wireless LAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>80 dpi x 72 dpi (W x H)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print Width</td>
<td>200 dots</td>
<td>180 dots</td>
<td>150 dots</td>
</tr>
<tr>
<td>Font</td>
<td>Font A, Font B</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about what character codes can be printed, refer to the user's manual that came with the printer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Characters in a Line</td>
<td>Font A</td>
<td>ANK: 33 characters,</td>
<td>ANK: 30 characters</td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>ANK: 40 characters</td>
<td>ANK: 36 characters</td>
</tr>
<tr>
<td>Character Size</td>
<td>Font A</td>
<td>ANK: 4.5 dots x 9 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>ANK: 3.5 dots x 9 dots (W x H)</td>
<td></td>
</tr>
<tr>
<td>Character Baseline</td>
<td>Font A</td>
<td>Bottom of the characters</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Font B</td>
<td>Bottom of the characters</td>
<td></td>
</tr>
<tr>
<td>Default Line Feed Space</td>
<td></td>
<td>12 dots</td>
<td></td>
</tr>
<tr>
<td>Color Specification</td>
<td></td>
<td>First color</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>First color, Second color (When using a two-color ribbon cassette)</td>
<td></td>
</tr>
<tr>
<td>Raster Image</td>
<td></td>
<td>Monochrome image</td>
<td></td>
</tr>
<tr>
<td>Logo</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Bar Code</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Two-Dimensional Code</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Ruled Line</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Page Mode Default Area</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Page Mode Maximum Area</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Page Mode Line</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rectangle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Cut</td>
<td></td>
<td>Cut, Feed cut</td>
<td></td>
</tr>
<tr>
<td>Specification of the paper feed position for labels / black mark paper</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td>Drawer Kick-Out</td>
<td></td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Buzzer</td>
<td></td>
<td>Not supported</td>
<td></td>
</tr>
<tr>
<td></td>
<td>76 mm</td>
<td>70 mm</td>
<td>58 mm</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>Paper Layout Settings</td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forced transmission mode</td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recovery from an error</td>
<td>Not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Rendering in HTML5 Canvas

This section describes how to use Web pages using the ePOS-Print Canvas API in the package. You can try how to render images in HTML5 Canvas and see what images can be rendered. The following Web pages are available:

- Rendering Text (canvas-print-text.html) (p.224)
- Rendering Images (canvas-print-image.html) (p.226)
- Rendering Graphics (canvas-print-graph.html) (p.228)
- Rendering Handwritten Images (canvas-print-hand.html) (p.230)
- Rendering Barcode (canvas-print-barcode.html) (p.232)
- Rendering Barcode (canvas-print-barcode.html) (p.232)

The Web pages introduced here are embedded into the sample program. For the details about how to place them, refer to Environment Settings (p.36).

Rendering Text (canvas-print-text.html)

Print text in HTML5 Canvas and perform a test print.

1. Open the following URL page using the Web browser.
   http://[Web server IP address]/canvas/canvas-print-text.html

2. Hello, World!
3. Hello, World!
4. Hello, World!

Open the following URL page using the Web browser.
http://[Web server IP address]/canvas/canvas-print-text.html
2 “EPSON ePOS-Print Sample Program” appears. Set items on the right of the page. The following items can be set:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Specify the rendering coordinates</td>
</tr>
<tr>
<td>Text</td>
<td>Specify the text to be printed</td>
</tr>
<tr>
<td>Style</td>
<td>Specify the text style</td>
</tr>
<tr>
<td>Clear</td>
<td>Clears the image drawn in the Canvas</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears the image drawn in the Canvas. In addition, the settings are reset to their default values.</td>
</tr>
</tbody>
</table>

3 Click the (Enter) button. The text is printed on Canvas on the left of the page according to the settings made on the right of the page.

4 Set the following and click the (Print) button.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Enter the following URL: http://[IP address of ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=[device ID of printer to be used for printing]&amp;timeout=[timeout time]</td>
</tr>
<tr>
<td>Mode</td>
<td>Set the color mode (Monochrome, Grayscale).</td>
</tr>
<tr>
<td>Brightness</td>
<td>Adjust the brightness. (Gamma value in the range 0.1-10.0)</td>
</tr>
<tr>
<td>Halftone</td>
<td>Set the halftone processing method for monochrome printing (two-tone).</td>
</tr>
<tr>
<td>Cut Paper</td>
<td>When this item is selected, feed cut is performed after printing.</td>
</tr>
<tr>
<td>Alignment</td>
<td>Specify the printing position alignment.</td>
</tr>
<tr>
<td>Color(Monochrome)</td>
<td>Specify the printing color in 2-tone.</td>
</tr>
</tbody>
</table>

5 The print result is displayed.
Rendering Images (canvas-print-image.html)

Draw an image in HTML5 Canvas and perform a test print.

1. Open the following URL page using the Web browser.
   http://[Web server IP address]/canvas/canvas-print-image.html

2. “EPSON ePOS-Print Sample Program” appears.
   Set items on the right of the page. The following items can be set:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Specify the rendering coordinates</td>
</tr>
<tr>
<td>Size</td>
<td>Specify the width and height of the image.</td>
</tr>
<tr>
<td>Angle</td>
<td>Specify the rotation angle of the image. The rotation angle is counted clockwise from the top left corner.</td>
</tr>
<tr>
<td>Image File (in the same server)</td>
<td>Specify the path to the image file. In this Web page, specify the name of an image file placed under the same directory as this Web page.</td>
</tr>
<tr>
<td>Clear</td>
<td>Clears the image drawn in the Canvas.</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears the image drawn in the Canvas. In addition, the settings are reset to their default values.</td>
</tr>
</tbody>
</table>

4. Property:
   - [ ] Mode: [ ] Monochrome [ ] Brightness: [ ] [ ] [ ]
   - [ ] Alignment: [ ] [ ] [ ] [ ] [ ] [ ]
   - [ ] Color: [ ] [ ] [ ] [ ] [ ] [ ]
   - [ ] Cut Paper

Print
3 Click the (Draw) button. The image is drawn on Canvas on the left of the page according to the settings made on the right of the page.

4 Set the following and click the (Print) button.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Enter the following URL: http://[IP address of ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=[device ID of printer to be used for printing]&amp;timeout=[timeout time]</td>
</tr>
<tr>
<td>Mode</td>
<td>Set the color mode (Monochrome, Grayscale).</td>
</tr>
<tr>
<td>Brightness</td>
<td>Adjust the brightness. (Gamma value in the range 0.1-10.0)</td>
</tr>
<tr>
<td>Halftone</td>
<td>Set the halftone processing method for monochrome printing (two-tone).</td>
</tr>
<tr>
<td>Cut Paper</td>
<td>When this item is selected, feed cut is performed after printing.</td>
</tr>
<tr>
<td>Alignment</td>
<td>Specify the printing position alignment.</td>
</tr>
<tr>
<td>Color(Monochrome)</td>
<td>Specify the printing color in 2-tone.</td>
</tr>
</tbody>
</table>

5 The print result is displayed.
Rendering Graphics (canvas-print-graph.html)

Draw an image in HTML5 Canvas and perform a test print.

1. Open the following URL page using the Web browser.
   http://[Web server IP address]/canvas/canvas-print-graph.html

2. "EPSON ePOS-Print Sample Program" appears.
   Set items on the right of the page. The following items can be set:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fill Settings</td>
<td>Specify the fill type and color</td>
</tr>
<tr>
<td>Rectangle</td>
<td>Specify the start coordinates, width and height.</td>
</tr>
<tr>
<td>Circle</td>
<td>Specify the central coordinates and radius.</td>
</tr>
<tr>
<td>Clear</td>
<td>Clears the image drawn in the Canvas</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears the image drawn in the Canvas. In addition, the settings are reset to their default values.</td>
</tr>
</tbody>
</table>

3. Click the (Draw) button.
   The image is drawn on Canvas on the left of the page according to the settings made on the right of the page.
Set the following and click the (Print) button.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Enter the following URL: http://([IP address of ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=([device ID of printer to be used for printing])&amp;timeout=([timeout time]).</td>
</tr>
<tr>
<td>Mode</td>
<td>Set the color mode (Monochrome, Grayscale).</td>
</tr>
<tr>
<td>Brightness</td>
<td>Adjust the brightness. (Gamma value in the range 0.1-10.0)</td>
</tr>
<tr>
<td>Halftone</td>
<td>Set the halftone processing method for monochrome printing (two-tone).</td>
</tr>
<tr>
<td>Cut Paper</td>
<td>When this item is selected, feed cut is performed after printing.</td>
</tr>
<tr>
<td>Alignment</td>
<td>Specify the printing position alignment.</td>
</tr>
<tr>
<td>Color(Monochrome)</td>
<td>Specify the printing color in 2-tone.</td>
</tr>
</tbody>
</table>

The print result is displayed.
Rendering Handwritten Images (canvas-print-hand.html)

Draw a handwritten image and perform a test print.

1. Open the following URL page using the Web browser.
   http://[Web server IP address]/canvas/canvas-print-hand.html

2. “EPSON ePOS-Print Sample Program” appears. Set the size of the pen on the right of the page.

3. Draw a freehand line on Canvas on the left of the page. For the mouse, drag it to draw a line; for the touch screen monitor, draw a line on the touch screen.

   To erase the drawn image, click the [Clear] button.
4 Set the following and click the (Print) button.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Enter the following URL: http://[IP address of ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=[device ID of printer to be used for printing]&amp;timeout=[timeout time]</td>
</tr>
<tr>
<td>Mode</td>
<td>Set the color mode (Monochrome, Grayscale).</td>
</tr>
<tr>
<td>Brightness</td>
<td>Adjust the brightness. (Gamma value in the range 0.1-10.0)</td>
</tr>
<tr>
<td>Halftone</td>
<td>Set the halftone processing method for monochrome printing (two-tone).</td>
</tr>
<tr>
<td>Cut Paper</td>
<td>When this item is selected, feed cut is performed after printing.</td>
</tr>
<tr>
<td>Alignment</td>
<td>Specify the printing position alignment.</td>
</tr>
<tr>
<td>Color(Monochrome)</td>
<td>Specify the printing color in 2-tone.</td>
</tr>
</tbody>
</table>

5 The print result is displayed.
Rendering Barcode (canvas-print-barcode.html)

Draw a barcode in HTML5 Canvas and perform a test print.
In the following example, an EAN13, JAN13 or UPC-A is drawn.

1. Open the following URL page using the Web browser.
   http://[Web server IP address]/canvas/canvas-print-barcode.html

2. "EPSON ePOS-Print Sample Program" appears.
   Set items on the right of the page. The following items can be set:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Specify the rendering coordinates.</td>
</tr>
<tr>
<td>Module Size</td>
<td>Specify the width and height of the bars.</td>
</tr>
<tr>
<td>Data</td>
<td>Specify EAN13 (JAN13) data.</td>
</tr>
<tr>
<td></td>
<td>For 12-digit numerical data, calculate and add the check digit.</td>
</tr>
<tr>
<td></td>
<td>For 13-digit numerical data, verify the check digit.</td>
</tr>
<tr>
<td></td>
<td>For UPC-A data, add 0 at the start of the string to make it 12-or 13-digit data.</td>
</tr>
<tr>
<td>Clear</td>
<td>Clears the image drawn in the Canvas.</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears the image drawn in the Canvas.</td>
</tr>
<tr>
<td></td>
<td>In addition, the settings are reset to their default values.</td>
</tr>
</tbody>
</table>
3 Click the (Draw) button.
The image is drawn on Canvas on the left of the page according to the settings
made on the right of the page.

4 Set the following and click the (Print) button.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Enter the following URL: http://[IP address of ePOS-Print supported TM printer]/cgi-bin/epos/service.cgi?devid=[device ID of printer to be used for printing]&amp;timeout=[timeout time]</td>
</tr>
<tr>
<td>Mode</td>
<td>Set the color mode (Monochrome, Grayscale).</td>
</tr>
<tr>
<td>Brightness</td>
<td>Adjust the brightness. (Gamma value in the range 0.1-10.0)</td>
</tr>
<tr>
<td>Halftone</td>
<td>Set the halftone processing method for monochrome printing (two-tone).</td>
</tr>
<tr>
<td>Cut Paper</td>
<td>When this item is selected, feed cut is performed after printing.</td>
</tr>
<tr>
<td>Alignment</td>
<td>Specify the printing position alignment.</td>
</tr>
<tr>
<td>Color(Monochrome)</td>
<td>Specify the printing color in 2-tone.</td>
</tr>
</tbody>
</table>

5 The print result is displayed.
Rendering Label (canvas-print-label.html)

Draw a label in HTML5 Canvas and perform a test print.

1. Open the following URL page using the Web browser.
   
   http://[Web server IP address]/canvas/canvas-print-label.html

2. "EPSON ePOS-Print Sample Program" appears.
   
   Set items on the right of the page. The following items can be set:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Specifies print data in the name field of label.</td>
</tr>
<tr>
<td>Description</td>
<td>Specifies print data in the description field of label.</td>
</tr>
<tr>
<td>Code</td>
<td>Prints barcode corresponding to the value.</td>
</tr>
<tr>
<td></td>
<td>- EAN13(JAN13)</td>
</tr>
<tr>
<td></td>
<td>- In case of 12 digits, check digit is added.</td>
</tr>
<tr>
<td></td>
<td>- In case of 13 digits, check digit is added.</td>
</tr>
<tr>
<td></td>
<td>- UPC-A</td>
</tr>
<tr>
<td></td>
<td>- Add 0 at the beginning and adjust to 12 to 13 digits.</td>
</tr>
<tr>
<td>Price</td>
<td>Specifies print data in the price field of label.</td>
</tr>
<tr>
<td>Clear</td>
<td>Clears the image drawn in the Canvas.</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears the image drawn in the Canvas.</td>
</tr>
<tr>
<td></td>
<td>In addition, the settings are reset to their default values.</td>
</tr>
</tbody>
</table>
3 Click the (Draw) button. The image is drawn on Canvas on the left of the page according to the settings made on the right of the page.

4 Set the following and click the (Print) button.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Enter the following URL: http://(IP address of ePOS-Print supported TM printer)/cgi-bin/epos/service.cgi?devid=(device ID of printer to be used for printing)&amp;timeout=(timeout time)</td>
</tr>
<tr>
<td>Mode</td>
<td>Set the color mode (Monochrome, Grayscale).</td>
</tr>
<tr>
<td>Brightness</td>
<td>Adjust the brightness. (Gamma value in the range 0.1-10.0)</td>
</tr>
<tr>
<td>Halftone(Monochrome)</td>
<td>Set the halftone processing method for monochrome printing (Two-tone).</td>
</tr>
<tr>
<td>Alignment</td>
<td>Specify the printing position alignment.</td>
</tr>
<tr>
<td>Color(Monochrome)</td>
<td>Specify the printing color in 2-tone.</td>
</tr>
<tr>
<td>Paper</td>
<td>Specify the paper type.</td>
</tr>
<tr>
<td>Feed to</td>
<td>Specify the paper feeding position.</td>
</tr>
<tr>
<td>Cut Paper</td>
<td>When this item is selected, feed cut is performed after printing.</td>
</tr>
<tr>
<td>Set the paper layout (only for TM-P60II/TM-P80)</td>
<td>Check when printing labels with paper layout specified.</td>
</tr>
<tr>
<td>Layout</td>
<td>Specify the label paper layout. Setting become effective when (Set the paper layout) is checked.</td>
</tr>
</tbody>
</table>

5 The print result is displayed.
Windows Store Apps

In this section, how to use the sample program with Windows store apps is described.

Sample Program Screen

Display the (Sample) screen. Executes printing. It can print the following.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)Samples</td>
<td>Display the (Sample) screen. Executes printing. It can print the following.</td>
<td>238</td>
</tr>
<tr>
<td></td>
<td>• Queue Ticket</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Coupon</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Label</td>
<td></td>
</tr>
<tr>
<td>(2)Settings</td>
<td>Display the (Settings) screen. Set up the following.</td>
<td>239</td>
</tr>
<tr>
<td></td>
<td>• Specifies the IP address of the ePOS-Print supported printer. (Default value: 192.168.192.168)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Specifies the Device ID of the printer to print queue ticket numbers and coupons. (Default value: local_printer)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Specifies the timeout time. (default : 60000)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Prints coupons in gray scale. (Only for supported models) (Default: No)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Specifies paper layout and print. (Only for TM-P60II Peeler) (Default: No)</td>
<td></td>
</tr>
</tbody>
</table>
Your Number
(ePOS-Print API)

Sample Shop

Your Number: 0001

Please wait until your ticket number is called.
Mon Aug 01 2011 16:18:00

Coupon
(ePOS-Print Canvas API)

Name Item A
Color Red
Code *2012001*

*: Die cut label: mount width 58 mm or above
   Label size: width 54 mm x height 25.4 mm or above
Environment of Sample Program

The environment of sample program is shown below.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development environment</td>
<td>Microsoft Visual Studio 2012</td>
</tr>
<tr>
<td>Required than</td>
<td>Windows 8</td>
</tr>
<tr>
<td>Sample program file name</td>
<td>win8/ePOS-Print Demo.zip</td>
</tr>
</tbody>
</table>

Environment setting Procedure

1. Configure your computer and ePOS-Print supported TM printer so that they can connect to the network.
2. Uncompress the sample program into any folder.
3. Open the sample program solution file using Visual Studio.
4. Start debugging.
Sample Program Settings

The screen is used to set the following:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
</table>
| IP address of the ePOS-Print supported printer | Specifies the IP address of the ePOS-Print supported printer. (Default value:  
* TM-i: DHCP  
  (If an address fails to be assigned via DHCP, the value becomes "192.168.192.168").  
* TM Printer: 192.168.192.168) |
| Device ID of the target printer           | Specifies the Device ID of the printer to print queue ticket numbers and coupons. (Default value: local_printer)  |
| Print timeout (milliseconds)               | Specifies the timeout time. (default: 60000)                                                                                                 |
| Print in grayscale (Coupon) (Only for supported models) | Prints coupons in gray scale. (Default: No)                                                                                              |
| Set the paper layout (Label) (Only for TM-P60II/TM-P80) | Prints a label in a specified layout. (Default: No)                                                                                     |
It executes printing.

Run the program according to the following procedure:

1. Select a type of printing from the left screen. There are following printing types.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
</table>
| Queue Ticket | Prints queue ticket numbers.  
This is a sample program using the ePOS-Print API. |
| Coupon  | Prints coupons.  
This is a sample program using the ePOS-Print Canvas API. |
| Label   | Prints labels.  
This is a sample program using the ePOS-Print API. |

2. Press (Print).