UNIX PRINTING SYSTEM

CUPS Translation Guide
CUPS−TRANS−1.1

Table of Contents

1 Scope

1.1 Identification

This translation guide provides instructions for creating translations of the CUPS message catalogs and web pages for the Common UNIX Printing System ("CUPS") Version 1.1 software.

1.2 System Overview

CUPS provides a portable printing layer for UNIX®−based operating systems. It has been developed by Easy Software Products to promote a standard printing solution for all UNIX vendors and users. CUPS provides the System V and Berkeley command−line interfaces.

CUPS uses the Internet Printing Protocol ("IPP") as the basis for managing print jobs and queues. The Line Printer Daemon ("LPD") Server Message Block ("SMB"), and AppSocket (a.k.a. JetDirect) protocols are also supported with reduced functionality. CUPS adds network printer browsing and PostScript Printer Description ("PPD") based printing options to support real−world printing under UNIX.

CUPS also includes a customized version of GNU Ghostscript (currently based off GNU Ghostscript 5.50) and an image file RIP that are used to support non−PostScript printers. Sample drivers for HP and EPSON printers are included that use these filters.

1.3 Document Overview

This translation guide is organized into the following sections:

- 1 – Scope
- 2 – References
- 3 – Character Sets
- 4 – Message Catalogs
- 5 – Web Interfaces
- A – Glossary

2 References

2.1 CUPS Documentation

The following CUPS documentation is referenced by this document:

- CUPS−CMP−1.1: CUPS Configuration Management Plan
- CUPS−IDD−1.1: CUPS System Interface Design Description
- CUPS−IPP−1.1: CUPS Implementation of IPP
- CUPS−SAM−1.1.x: CUPS Software Administrators Manual
- CUPS−SDD−1.1: CUPS Software Design Description
- CUPS−SPM−1.1.x: CUPS Software Programming Manual
- CUPS−SSR−1.1: CUPS Software Security Report
- CUPS−STP−1.1: CUPS Software Test Plan
- CUPS−SUM−1.1.x: CUPS Software Users Manual
- CUPS−SVD−1.1: CUPS Software Version Description

2.2 Other Documents

The following non−CUPS documents are referenced by this document:

- Adobe PostScript Printer Description File Format Specification, Version 4.3.
- Adobe PostScript Language Reference, Third Edition.
- IPP/1.1: Implementers Guide
- RFC 1179, Line Printer Daemon Protocol
- RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax
- RFC 2567, Design Goals for an Internet Printing Protocol
- RFC 2568, Rationale for the Structure of the Model and Protocol for the Internet Printing Protocol
- RFC 2569, Mapping between LPD and IPP Protocols
- RFC 2616, Hypertext Transfer Protocol −− HTTP/1.1
- RFC 2617, HTTP Authentication: Basic and Digest Access Authentication
- RFC 2910, IPP/1.1: Encoding and Transport
- RFC 2911, IPP/1.1: Model and Semantics
- RFC 3380, IPP: Job and Printer Set Operations

3 Character Sets

CUPS uses character set files to define the mapping of local character sets to Unicode code points, as well as the fonts that should be used for different ranges of characters.

CUPS includes files for common 8–bit encodings as well as UTF–8 for Unicode text. The format of these files is described in the CUPS Interface Design Description (IDD) document. Current character sets are enumerated in the CUPS API, so in order to add a new character set you must patch the CUPS source as well as provide a new charset file.

CUPS 1.1 supports the following character sets:

- iso–8859–1
- iso–8859–2
- iso–8859–3
- iso–8859–4
- iso–8859–5
- iso–8859–6
- iso–8859–7
- iso–8859–8
- iso–8859–9
- iso–8859–10
- iso–8859–13
- iso–8859–14
- iso–8859–15
- koi8–r
- koi8–u
- us–ascii
- utf–8
- windows–874
- windows–1250
- windows–1251
- windows–1252
- windows–1253
- windows–1254
- windows–1255
- windows–1256
- windows–1257
- windows–1258

4 Message Catalogs

CUPS message catalogs are text files that identify the default character set for the locale and a list of localized message strings for the CUPS software. The format of the message catalog files is described in the CUPS IDD.

Message catalogs are named *cups_ll*, *cups_ll_CC*, or *cups_ll_CC.charset*, where "ll" is the standard 2–letter abbreviation for the language, "CC" is the standard 2–letter abbreviation for the country, and "charset" is the charset name which may differ from the list above.

Each message catalog file is stored in a subdirectory named *ll*, *ll_CC*, or *ll_CC.charset* to match the trailing portion of the message catalog filename.

When translating a new message catalog, copy the *cups_C* message catalog file to a new subdirectory; to translate the message catalog to Canadian French, you would type the following commands:

```
cd locale ENTER
mkdir fr_CA ENTER
cp C/cups_C fr_CA/cups_fr_CA ENTER
```

Alternatively, you could copy the existing *cups_fr* message catalog and then make any changes necessary.

Once you have make your copy of the file, edit it using your favorite text editor to translate the text to the desired language. Be sure to preserve any numbers starting in the first column, as they indicate a new message number – you'll see this for the HTTP status messages.

Finally, add your locale to the list of locales in the makefile and run the following command to install it:

```
make install ENTER
```

5 Web Interfaces

The CUPS scheduler provides a web interface that can be used to do many common printing and administration tasks. The built–in web server supports localization of web pages through the use of subdirectories for each locale, e.g. "fr" for French, "de" for German, "fr_ca" for French in Canada, and so forth.

5.1 Template Files

Template files are HTML files with special formatting characters in them that allow substition of variables and arrays. The CUPS CGI programs (`admin.cgi`, `classes.cgi`, `jobs.cgi`, and `printers.cgi`) use these template file to provide dynamic content for the web interface. Template files are installed in the */usr/share/cups/templates* directory by default.

Translated versions of the template files should be installed in the appropriate subdirectories under */usr/share/cups/templates*. For example, Canadian French template files should be stored in the */usr/share/cups/templates/fr_CA* directory.

5.1.1 Inserting Attributes and Values

Template files consist of HTML with variable substitutions for named inside curley braces "{name}". Variable names are generally the IPP attribute names with the hyphen ("–") replaced by the underscore ("_") character. For example, the `job-printer-uri` attribute is renamed to `job_printer_uri`.

Curley braces ("{" and "}") to indicate substitutions, and the backslash ("\") character for quoting. To insert any of these special characters as–is you need to use the HTML `&name;` mechanism or prefix each special character with the backslash ("\".)

You substitute the value of a variable using {NAME} in your template file. If the variable is undefined then the {NAME} string is output as–is.

To substitute an empty string if the variable is undefined, use {?NAME} instead.

5.1.2 Array Substitutions

The number of array elements can be inserted using {#NAME}. If the array is undefined then 0 is output. The current array element (starting at 1) is inserted with {#}.

Arrays are handled using {[NAME] at the beginning of a section and } at the end. The information between the closing bracket ("]") and closing brace ("}") is repeated for as many elements as are in the named array. For example, the following template will display a list of each job in the `job_id` array:

```
<TABLE>
<TR>
        <TH>Job ID</TH>
        <TH>Destination</TH>
        <TH>Title</TH>
</TR>

{[job_id]
<TR>
```

```
        <TD>{?job_id}</TD>
        <TD>{?job_printer_name}</TD>
        <TD>{?job_name}</TD>
    </TR>
    }
    </TABLE>
```

Arrays can be nested, however all elements within the curley braces ("{" and "}") are indexed using the innermost array.

5.1.3 Conditional Tests

Templates can also test variables against specific values and conditionally include text in the template. The format is:

```
{variable?true:false}
{variable=value?true:false}
{variable!value?true:false}
{variable<value?true:false}
{variable>value?true:false}
```

where *true* is the text that is included if the condition is true and *false* is the text that is included if the condition is false. A value of # is replaced with the current element number (starting at 1.)

The character after the variable name specifies the condition to test:

Char	Condition
?	True if *variable* exists.
=	True if *variable* is equal to *value*.
!	True if *variable* is not equal to *value*.
<	True if *variable* is less than *value*.
>	True if *variable* is greater than *value*.

5.1.4 Template File List

The following template files are used by the web interface:

add−class.tmpl
      This is the initial form that is shown to add a new printer class.
add−printer.tmpl
      This is the initial form that is shown to add a new printer.
admin−op.tmpl
      This is the template that is used to display an error message when the admin interface sees an undefined operation name.
admin.tmpl
      This is the template that shows the initial menu of operations (add a class, manage classes, etc.)
choose−device.tmpl
      This is the form that shows the list of available devices.
choose−make.tmpl
      This is the form that shows the list of available manufacturers.
choose−members.tmpl

This is the form that shows the list of available printers that can be added to a class.

choose−model.tmpl

This is the form that shows the list of available printer models/drivers.

choose−serial.tmpl

This is the form that allows the user to choose a serial port and any options.

choose−uri.tmpl

This is the form that allows the user to enter a device URI for network printers.

class−added.tmpl

This template shows the "class added" message.

class−confirm.tmpl

This is the template used to confirm the deletion of a class.

class−deleted.tmpl

This template shows the "class deleted" message.

classes.tmpl

This template shows one or more printer classes.

class−modified.tmpl

This template shows the "class modified" message.

config−printer.tmpl

This template starts the printer configuration form.

config−printer2.tmpl

This template ends the printer configuration form.

error.tmpl

This template displays a generic error message.

header.tmpl

This template is used as the standard header on all dynamic content.

job−cancel.tmpl

This template shows "job cancelled".

job−hold.tmpl

This template shows "job held".

job−op.tmpl

This is the template that is used to display an error message when the job interface sees an undefined operation name.

job−release.tmpl

This template shows "job released".

job−restart.tmpl

This template shows "job restarted".

jobs.tmpl

This template is used to list the print jobs on a server, class, or printer.

modify−class.tmpl

This template is used as the first form when modifying a class.

modify−printer.tmpl

This template is used as the first form when modifying a printer.

option−boolean.tmpl

This template is used to select a boolean PPD option.

option−header.tmpl

This template is used to start a PPD option group.

option−pickmany.tmpl

This template is used to select a multi−valued PPD option.

option−pickone.tmpl

This template is used to select a single−valued PPD option.

option−trailer.tmpl

This template is used to end a PPD option group.

printer−accept.tmpl
>   This template shows "printer now accepting jobs".

printer−added.tmpl
>   This template shows "printer added".

printer−configured.tmpl
>   This template shows "printer configured".

printer−confirm.tmpl
>   This template asks the user to confirm the deletion of a printer.

printer−deleted.tmpl
>   This template shows "printer deleted".

printer−modified.tmpl
>   This template shows "printer modified".

printer−purge.tmpl
>   This template shows "printer has been purged of all jobs".

printer−reject.tmpl
>   This template shows "printer now rejecting jobs".

printer−start.tmpl
>   This template shows "printer started".

printers.tmpl
>   This template is used to list information on one or more printers.

printer−stop.tmpl
>   This template shows "printer stopped".

test−page.tmpl
>   This template shows "test page printed".

trailer.tmpl
>   This template is used as the standard trailer on all dynamic content.

5.2 CGI Programs

CUPS uses four CGI programs to manage the dynamic web interfaces:

- `admin.cgi`
- `classes.cgi`
- `jobs.cgi`
- `printers.cgi`

5.2.1 admin.cgi

The `admin.cgi` program handles all of the printer and class administration functions and is run for all direct accesses to the */admin* resource. For most operations it uses the `PRINTER_NAME` and `OP` form variables to specify the action requested.

The following `OP` values are supported:

accept−jobs
>   Accepts jobs on the named destination.

add−class
>   Adds a new printer class. This operation also adds several other form variables:
>   *MEMBER_URIS*

Sets the members of the class. Multiple MEMBER_URIS values can be provided.

PRINTER_INFO

Sets the printer−info attribute for the printer class, which is usually the printer description.

PRINTER_LOCATION

Sets the printer−location attribute for the printer class.

add−printer

Adds a new printer. This operation also adds several other form variables:

BAUDRATE

Sets the baud rate for serial devices.

BITS

Sets the number of data bits for serial devices.

DEVICE_URI

Sets the device URI for the printer.

FLOW

Sets the flow control for serial devices.

PARITY

Sets the parity checking for serial devices.

PPD_NAME

Sets the driver name for the printer ("raw" for a raw queue.)

PRINTER_INFO

Sets the printer−info attribute for the printer, which is usually the printer description.

PRINTER_LOCATION

Sets the printer−location attribute for the printer.

config−printer

Configures an existing printer. This operation uses form variables of the same name as the options in the printer's PPD file.

delete−class

Deletes a printer class. The form variable CONFIRM may be set to any value to bypass the confirmation page.

delete−printer

Deletes a printer. The form variable CONFIRM may be set to any value to bypass the confirmation page.

modify−class

Modifies a printer class. See the add−class operation for a list of form variables.

modify−printer

Modifies a printer. See the add−printer operation for a list of form variables.

purge−jobs

Purges all jobs on the named destination.

reject−jobs

Rejects new jobs on the named destination.

start−printer

Starts the named destination.

stop−printer

Stops the named destination.

5.2.2 classes.cgi

The classes.cgi program is responsible for listing class information, including jobs destined for that class. It is for all direct accesses to the */classes* resource and supports the optional form variables OP and WHICH_JOBS. If no form variables are supplied then the CGI lists all or a specific class and the active jobs

on each class.

The following WHICH_JOBS values are supported:

completed
    Show only the completed jobs.
not−completed
    Show only the active jobs.

The following OP values are supported:

print−test−page
    Print a PostScript test page.

5.2.3 jobs.cgi

The jobs.cgi program handles all of the job functions and is run for all direct accesses to the */jobs* resource. For most operations it uses the JOB_ID, OP, and WHICH_JOBS form variables to specify the action requested.

The following WHICH_JOBS values are supported:

completed
    Show only the completed jobs.
not−completed
    Show only the active jobs.

The following OP values are supported:

job−cancel
    Cancels a job.
job−hold
    Holds a job indefinitely.
job−release
    Releases a job for printing.
job−restart
    Restarts a stopped, cancelled, completed, or aborted print job.

5.2.4 printers.cgi

The printers.cgi program is responsible for listing printer information, including jobs destined for that printer. It is for all direct accesses to the */printers* resource and supports the optional form variables OP and WHICH_JOBS. If no form variables are supplied then the CGI lists all or a specific printer and the active jobs on each printer.

The following WHICH_JOBS values are supported:

completed
    Show only the completed jobs.
not−completed

Show only the active jobs.

The following `OP` values are supported:

print−test−page
Print a PostScript test page.

A Glossary

A.1 Terms

C
> A computer language.

parallel
> Sending or receiving data more than 1 bit at a time.

pipe
> A one−way communications channel between two programs.

serial
> Sending or receiving data 1 bit at a time.

socket
> A two−way network communications channel.

A.2 Acronyms

ASCII
> American Standard Code for Information Interchange

CUPS
> Common UNIX Printing System

ESC/P
> EPSON Standard Code for Printers

FTP
> File Transfer Protocol

HP−GL
> Hewlett−Packard Graphics Language

HP−PCL
> Hewlett−Packard Page Control Language

HP−PJL
> Hewlett−Packard Printer Job Language

IETF
> Internet Engineering Task Force

IPP
> Internet Printing Protocol

ISO
> International Standards Organization

LPD
> Line Printer Daemon

MIME
> Multimedia Internet Mail Exchange

PPD
> PostScript Printer Description

SMB
> Server Message Block

TFTP
> Trivial File Transfer Protocol