enscript

NAME

enscript - convert text files to PostScript

SYNOPSIS

enscript [-12BcCgGhjkKlmOqrRvVzZ] [-a pages] [-A align] [-b header] [-d printer] [-D key[:value]] [-e [char]] [-E [lang]] [-f font] [-F header_font] [-H [num]] [-i indent] [-I filter] [-L lines_per_page] [-M media] [-n copies] [-N newline] [-o outputfile] [-o -] [-p outputfile] [-p -] [-P printer] [-s baselineskip] [-S key[:value]] [-t title] [-T tabsize] [-u [text]] [-U num] [-X encoding] [filename ...]

DESCRIPTION

Enscript converts text files to PostScript and spools generated PostScript output to the specified printer or leaves it to file. If no input files are given, enscript processes standard input. Enscript can be extended to handle different output media and it has many options which can be used to customize printouts.

OPTIONS

-1, -2, --columns=num
Specify how many columns each page have. With long option --columns=num you can specify more than 2 columns per page.

-a pages, --pages=pages
Specify which pages are printed. Page specification pages can be given in following formats:

start-end
print pages from start to end

-end
print pages from 0 to end

start-
print pages from start to end

page
print page page

odd
print odd pages

even
print even pages

-A align, --file-align=align
Align separate input files to even align page count. This is handy for two-side and 2-up printings (--file-align=2).

-b header, --header=header
Use text header as a page header. The default page header is constructed from file's name and last modification time.

-B, --no-header
Do not print page headers.

-c, --truncate-lines
Cut lines that are too long. As a default, enscript wraps long lines so no information is lost.

See also option --slice which can be used to slice long lines to separate pages.

-C, --line-numbers
Precede each line with its line number.

-d name
Spool output to the printer name.

-D key[:value], --setpagedevice=key[:value]
Pass a page device definition to the generated PostScript output. If no value is given, key key is removed from definitions.

For example, command

enscript -DDuplex:true foo.txt

prints file foo.txt in duplex (two side) mode.

Page device operators are implementation dependant but they are standardized. See section PAGE DEVICE OPTIONS for details.

-e [char], --escapes[=char]
Enable special escapes interpretation (see section SPECIAL ESCAPES). If argument char is given, it changes the escape character to char. The default escape character is 0.

-E [lang], --pretty-print[=lang]
Pretty-print source code by creating a special input filter with the states program. Optional argument lang specifies the language to highlight, as a default states makes an educated guess.

A description of supported highlighting languages and file formats can be printed with command:

enscript --help-pretty-print

The highlighting rules are defined in the `/usr/local/share/enscript/enscript.st' file which can be edited to create highlighting definitions for new languages.

Note! You can't use your own input filters with this option.

-f name, --font=name
Select font that is used for body text. The default body font is Courier10, unless multicolumn landscape printing mode is selected, in which case the default is Courier7.

Font specification name contains two parts: font's name and font's size in points. For example "Times-Roman12" selects "Times-Roman" font with size 12pt.

The font specification name can also be given in format `name@ptsize', where font's name and point size are separated by a `@' character. This allows enscript to use fonts which contain digit characters in their names.

The font point size can also be given in format width/height where width and height specify the font's size in x- and y-directions. For example "Times-Roman@10/12" selects 10 points wide and 12 points high "Times-Roman" font.

Note! font sizes can be given as a decimal number. For example "Times-Roman10.2" selects 10.2pt "Times-Roman" font.

-F name, --header-font=name
Select font for header texts.

-g, --print-anyway
Print file even if it contains binary data. Option is here only for compatibility since enscript prints binary files anyway.

-G, --fancy-header[=name]
Print fancy page header name to top of each page. Option -G specifies the default fancy header (see section CONFIGURATION FILES to see how the default fancy header can be changed).

-h, --no-job-header
Suppress printing of the job header page.

-H [num], --highlight-bars[=num]
Specify how high highlight bars are in lines. If num is not given, the default value 2 is used. As a default, no highlight bars are printed.

-i num, --indent=num
Indent every line num characters. The indentation can also be specified in other units by appending an unit specifier after the number. Possible unit specifiers and the corresponding units are:

c
centimeters

i
inches

l
characters (default)

p
PostScript points

-I filter, --filter=filter
Read all input files through input filter filter. Input filter can be a single command or a command pipeline and it can refer to the name of the input file with escape `%s'. The name of the input file stdin can be changed with option `--filter-stdin'.

For example, the following command can be used to print file `foo.c' by using only upper-case characters:

enscript --filter="cat %s | tr 'a-z' 'A-Z'" foo.c

And to highlight changes made to files since the last checkout:

enscript --filter="rcsdiff %s | diffpp %s" -e *.c

Note! To include string "%s" to the filter command, you must write it as "%%s".

-j, --borders
Print borders around columns.

-k, --page-prefeed
Enable page prefeed.

-K, --no-page-prefeed
Disable page prefeed (default).

-l, --lineprinter
Emulate lineprinter. This option is a shortcut for options: --lines-per-page=66, --no-header, --portrait, --columns=1.

-L num, --lines-per-page=num
Print only num lines per each page.

-m, --mail
Send mail notification to user after print job has been completed.

-M name, --media=name
Select output media name. Enscript's default output media is A4.

-n num, --copies=num
Print num copies of each page.

-N nl, --newline=nl
Select the newline character. Possible values for nl are: n (unix newline, 0xa hex) and r (mac newline, 0xd hex).

-o file
An alias for option -p, --output.

-O, --missing-characters
Print a listing of character codes which couldn't be printed.

-p file, --output=file
Leave output to file file. If file is `-', leave output to stdout.

-P name, --printer=name
Spool output to the printer name.

-q, --quiet, --silent
Make enscript really quiet. Only fatal error messages are printed to stderr.

-r, --landscape
Print in landscape mode; rotate page 90 degrees.

-R, --portrait
Print in portrait mode (default).

-s num, --baselineskip=num
Specify the baseline skip in PostScript points. Number num can be given as a decimal number. When enscript moves from line to line, current point y coordinate is moved (font point size + baselineskip) points down. The default baseline skip is 1.

-S key[:value], --statusdict=key[:value]
Pass a statusdict definition to the generated PostScript output. If no value is given, key key is removed from definitions.

Statusdict operators are implementation dependant; see printer's documentation for details.

For example the following command prints file foo.txt by using paper from the paper tray 1 (assuming that printer supports paper tray selection).

enscript -Ssetpapertray:1 foo.txt

-t title, --title=title
Set banner page's job title to title. Option sets also the name of the input file stdin.

-T num, --tabsize=num
Set tabulator size to num (default is 8).

-u [text], --underlay[=text]
Print string text under every page. Text's properties can be changed with options --ul-angle, --ul-font, --ul-gray, --ul-position and --ul-style.

If no text is given, no underlay is printed. This can be used to remove underlay that was specified with the `Underlay' configuration file option.

-U num, --nup=num
Print num logical pages on each output page (N-up printing).

-v, --verbose[=level]
Tell what enscript is doing.

-V, --version
Print enscript version and exit.

-X name, --encoding=name
Use input encoding name. Currently enscript supports following encodings:

latin1
ISO-8859/1 (ISO latin1) (enscript's default encoding).

latin2
ISO-8859/2 (ISO latin2)

latin3
ISO-8859/3 (ISO latin3)

latin5
ISO-8859/5 (ISO latin5)

ascii
7-bit ascii

asciiscands
7-bit ascii with some scandinavian extensions

ibmpc
IBM PC charset

mac
Mac charset

vms
VMS multinational charset

hp8
HP Roman-8 charset

koi8
Adobe Standard Cyrillic Font KOI8 charset

ps
PostScript font's default encoding

pslatin1
PostScript interpreter's `ISOLatin1Encoding'

-z, --no-formfeed
Turn off form feed character interpretation.

-Z, --pass-through
Pass through all PostScript and PCL files without any modifications. This allows that enscript can be used as a lp filter.

PostScript files are recognized by looking up the `%!' magic cookie from the beginning of the file. Note! Enscript recognized also the Windoze damaged `^D%!' cookie.

PCL files are recognized by looking up the `^[E' or `^[%' magic cookies from the beginning of the file.

--download-font=fontname
Include the font description file (.pfa or .pfb file) of the font fontname to the generated output.

--filter-stdin=name
Specify how stdin is shown to the input filter. The default value is an empty string ("") but some programs require that stdin is called something else, usually "-".

--help
Print short help message and exit.

--help-pretty-print
Describe all supported --pretty-print languages and file formats.

--highlight-bar-gray=gray
Specify the gray level which is used to print highlight bars.

--list-media
List the names of all known output media and exit successfully.

--list-options
List all options and their current values. Exit successfully.

--non-printable-format=format
Specify how non-printable characters are printed. Possible values for format are:

caret
caret notation: `^@', `^A', `^B', ...

octal
octal notation: `\000', `\001', `\002', ... (default)

questionmark
replace non-printable characters with a question mark `?'

space
replace non-printable characters with a space ` '

--page-label-format=format
Set page label format to format. Page label format specifies how labels for the `%%Page:' PostScript comments are formatted. Possible values are:

short
Print current pagenumber: `%%Page: (1) 1' (default)

long
Print current filename and pagenumber: `%%Page: (main.c: 1) 1'

--printer-options=options
Pass extra options to the printer command.

--slice=num
Print vertical slice num. Slices are vertical regions of input files, new slice starts from the point where the line would otherwise be wrapped to the next line. Slice numbers start from 1.

--toc Print table of contents to the end of the print job.

--ul-angle=angle
Set underlay text's angle. As a default, angle is atan(-page_height, page_width).

--ul-font=name
Select font for the underlay text. The default underlay font is Times-Roman200.

--ul-gray=num
Print underlay text with gray value num (0 ... 1), the default gray is .8.

--ul-position=position_spec
Set underlay text's starting position according to position_spec. Position specification must be given in format: `sign xpos sign ypos', where sign must be `+' or `-'. Positive dimensions are measured from the lower left corner and negative dimensions from the upper right corner. For example, spec `+0-0' specifies the upper left corner and `-0+0' specifies the lower right corner.

--ul-style=style
Set underlay text's style to style. Possible values for style are:

outline
print outline underlay texts (default)

filled
print filled underlay texts

CONFIGURATION FILES

Enscript reads configuration information from following sources (in this order): command line options, environment variable ENSCRIPT, user's personal configuration file ($HOME/.enscriptrc), site configuration file (/usr/local/etc/enscriptsite.cfg) and system's global configuration file (/usr/local/etc/enscript.cfg).

The configuration files have the following format:

Empty lines and lines starting with `#' are comments.

All other lines are option lines and have format:

OPTION [arguments ...].

Following options can be specified:

AcceptCompositeCharacters: bool
Specify whatever PostScript font's composite characters are accepted as printable or should they be considered as non-existent. The default value is false (0).

AFMPath: path
Specifies search path for the AFM files.

AppendCtrlD: bool
Specify if the Control-D (^D) character should be appended to the end of the output. The default value is false (0).

Clean7Bit: bool
Specify how characters greater than 127 are printed. Value true (1) generates 7-bit clean code by escaping all characters greater than 127 to the backslash-octal notation (default). Value false (0) generates 8-bit PostScript code leaving all characters untouched.

DefaultEncoding: name
Select the default input encoding. Encoding name name can be one of the values of the -X, --encoding option.

DefaultFancyHeader: name
Select the default fancy header. Default header is used when option -G is specified or option --fancy-header is given without an argument. System-wide default is `enscript'.

DefaultMedia: name
Select the default output media.

DefaultOutputMethod: method
Select the default target to which generated output is send. Possible values for method are:

printer
send output to printer (default)

stdout
send output to stdout

DownloadFont: fontname
Include the font description file of the font fontname to the generated output.

EscapeChar: num
Specify the escape character for special escapes. The default value is 0.

FormFeedType: type
Specify what to do when a formfeed character is encountered from the input. Possible values for type are:

column
move to the beginning of the next column (default)

page
move to the beginning of the next page

HighlightBarGray: gray
Specify the gray level which is used to print highlight bars.

HighlightBars: num
Specify how high highlight bars are in lines. The default value is 0 so no highlight bars are printed.

LibraryPath: path
Specifies enscript's library path that is used to lookup various resources. Default path is: `/usr/local/share/enscript:home/.enscript'. Where home is the user's home directory.

Media: name width height llx lly urx ury
Add a new output media with name name. Media's physical dimensions are width and height. Media's bounding box is specified by points (llx, lly) and (urx, ury). **Enscript** prints all graphics inside media's bounding box.

User can select this media by giving option -M name.

NoJobHeaderSwitch: switch
Specify the spooler option to suppress the print job header. This option is passed to the printer spooler when enscript's option -h, --no-job-header is selected.

NonPrintableFormat: format
Specify how non-printable characters are printed. Possible values for format are the same which can be given to the --non-printable-format option.

OutputFirstLine: line
Set PostScript output's first line to line, the default value is PS-Adobe-3.0. Since some printers do not like DSC levels greater than 2.0, this option can be used to change the output first line to something more suitable like %!PS-Adobe-2.0 or %!.

PageLabelFormat: format
Set page label format to format. Possible values for format are the same which can be given to the --page-label-format option.

PagePrefeed: bool
Enable / disable page prefeed. The default is false (0).

Printer: name
Names the printer to spool to.

QueueParam: name
The spooler command switch for the printer queue, e.g. -P in lpr -Pps. This option can also be used to pass other flags to the spooler command but they must be given before the queue switch.

SetPageDevice: key[:value]
Pass a page device definition to the generated PostScript output.

Spooler: name
Names printer spooler command. Enscript pipes generated PostScript to command name.

StatesColorModel: model
Set the pretty-printing color model to model. Possible values are blackwhite and emacs.

StatesConfigFile: file
Read pretty-printer states configuration from file file. The default config file is `/usr/local/share/enscript/enscript.st'.

StatesHighlightLevel: level
Set the pretty-printing highlight level to level. Possible values are none, light and heavy.

StatusDict: key[:value]
Pass a statusdict definition to the generated PostScript output.

TOCFormat: format
Format table of contents entries with format string format. Format string format can contain the same escapes which are used to format header strings with the `%Format' special comment.

Underlay: text
Print string text under every page.

UnderlayAngle: num
Set underlay text's angle to num.

UnderlayFont: fontspec
Select font for the underlay text.

UnderlayGray: num
Print underlay text with gray value num.

UnderlayPosition: position_spec
Set underlay text's starting position according to position_spec.

UnderlayStyle: style
Set underlay text's style to style.

CUSTOMIZATION

Users can create their own fancy headers by creating a header description file and placing it in a directory which is in enscript's library path. The name of the header file must be in format: `headername.hdr'. Header can be selected by giving option: --fancy-header=headername.

Header description file contains PostScript code that paints the header. Description file must provide procedure do_header which is called by enscript at the beginning of every page.

Header description file contains two parts: comments and code.

The parts are separated by a line containing text:

% -- code follows this line --

Enscript copies only the code part of description file to the generated PostScript output. The comments part can contain any data, it is not copied. If separator line is missing, no data is copied to output.

Enscript defines following constants which can be used in header description files:

d_page_w
page width
d_page_h
page height
d_header_x
header lower left x coordinate
d_header_y
header lower left y coordinate
d_header_w
header width
d_header_h
header height
d_footer_x
footer lower left x coordinate
d_footer_y
footer lower left y coordinate
d_footer_w
footer width
d_footer_h
footer height
d_output_w
width of the text output area
d_output_h
height of the text output area
user_header_p
predicate which tells if user has defined his/her own header string: true/false
user_header_str
if user_header_p is true, this is the user supplied header string.
HF standard header font (from -F, --header-font option). This can be selected simply by invoking command: `HF setfont'.
pagenum
the number of the current page
fname
the full name of the printed file (/foo/bar.c)
fdir the directory part of the file name (/foo)
ftail file name without the directory part (bar.c)
gs_languagelevel
PostScript interpreter's language level (currently 1 or 2)

You can also use the following special comments to customize your headers and to specify some extra options. Special comments are like DSC comments but they start with a single `%' character; special comments start from the beginning of the line and they have the following syntax:

%commentname: options

Currently enscript support the following special comments:

%Format: name format

Define a new string constant name according to the format string format. Format string start from the first non-space character and it ends to the end of the line. Format string can contain general `%' escapes and input file related `$' escapes. Currently following escapes are supported:

%%
character `%'

$$
character `$'

$%
current page number

$=
number of pages in the current file

$(VAR)
value of the environment variable VAR.

%c
trailing component of the current working directory

%C ($C)
current time (file modification time) in `hh:mm:ss' format

%d
current working directory

%D ($D)
current date (file modification date) in `yy-mm-dd' format

%D{string} ($D{string})
format string string with the strftime(3) function. `%D{}' refers to the current date and `$D{}' to the input file's last modification date.

%E ($E)
current date (file modification date) in `yy/mm/dd' format

%F ($F)
current date (file modification date) in `dd.mm.yyyy' format

%H
document title

$L
number of lines in the current input file. This is valid only for the toc entries, it can't be used in header strings.

%m
the hostname up to the first `.' character

%M
the full hostname

%n
the user login name

$n
input file name without the directory part
%N
the user's pw_gecos field up to the first `,' character
$N
the full input file name
%t ($t)
current time (file modification time) in 12-hour am/pm format
%T ($T)
current time (file modification time) in 24-hour format `hh:mm'
%* ($*)
current time (file modification time) in 24-hour format with seconds `hh:mm:ss'
$v
the sequence number of the current input file
$V
the sequence number of the current input file in the `Table of Contents' format: if the --toc option is given, escape expands to `num-'; if the --toc is not given, escape expands to an empty string.
%W ($W)
current date (file modification date) in `mm/dd/yy' format

All format directives except `$=' can also be given in format

escape width directive

where width specifies the width of the column to which the escape is printed. For example, escape "$5%" will expand to something like " 12". If the width is negative, the value will be printed left-justified.

For example, the `emacs.hdr' defines its date string with the following format comment:

%Format: eurdatestr %E

which expands to:

/eurdatestr (96/01/08) def

%HeaderHeight: height
Allocate height points space for the page header. The default header height is 36 points.
%FooterHeight: height
Allocate height points space for the page footer. The default footer height is 0 points.

According to Adobe's Document Structuring Conventions (DSC), all resources needed by a document must be listed in document's prolog.

Since user's can create their own headers, enscript don't know what resources those headers use. That's why all headers must contain a standard DSC comment that lists all needed resources.

For example, used fonts can be listed with following comment:

%%DocumentNeededResources: font fontname1 fontname2

Comment can be continued to the next line with the standard continuation comment:

%%+ font fontname3

SPECIAL ESCAPES

Enscript supports special escape sequences which can be used to add some page formatting commands to ASCII documents. As a default, special escapes interpretation is off, so all ASCII files print out as everyone expects. Special escapes interpretation is activated by giving option -e, --escapes to enscript.

All special escapes start with the escape character. The default escape character is ^@ (octal 000); escape character can be changed with option -e, --escapes. Escape character is followed by escape's name and optional options and arguments.

Currently enscript supports following escapes:

color change the text color. Escape's syntax is:

^@color{red green blue}

where color components red, green and blue are given as a decimal numbers between 0 and 1.

comment

comment the rest of the line including the newline character. Escape's syntax is:

^@comment text newline_character

epsf inline EPS file to the document. Escape's syntax is:

^@epsf[options]{filename}

where options is an optional sequence of option characters and values enclosed with brackets and filename is the name of the EPS file.

If filename ends to the `|' character, then filename is assumed to name a command that prints EPS data to its standard output. In this case, enscript opens a pipe to the specified command and reads EPS data from pipe.
The following options can be given to the epsf escape:

c

print image centered

r

print image right justified

n

do not update current point. Following output is printed to that position where the current point was just before the epsf escape

nx
do not update current point x coordinate
ny
do not update current point y coordinate
xnum
move image's top left x coordinate num characters from current point x coordinate (relative position)
xnuma
set image's top left x coordinate to column num (absolute position)
ynum
move image's top left y coordinate num lines from current line (relative position)
ynuma
set image's top left y coordinate to line num (absolute position)
hnum
set image's height to num lines
snum
scale image with factor num
sxnum
scale image in x direction with factor num
synum
scale image in y direction with factor num

As a default, all dimensions are given in lines (vertical) and characters (horizontal). You can also specify other units by appending an unit specifier after number. Possible unit specifiers and the corresponding units are:

c centimeters
i inches
l lines or characters (default)
p PostScript points

For example to print an image one inch high, you can specify height by following options: h1i (1 inch), h2.54c (2.54 cm), h72p (72 points).

font select current font. Escape's syntax is:

^@font{fontname}

where fontname is a standard font specification. Special font specification default can be used to select the default body font (enscript's default or the one specified by the command line option -f, --font).

ps include raw PostScript code to the output. Escape's syntax is:

^@ps{code}

shade
highlight regions of text by changing the text background color. Escape's syntax is:

^@shade{gray}

where gray is the new text background gray value. The default value is 1.0 (white) which disables highlighting.

PAGE DEVICE OPTIONS

Page device is a PostScript level 2 feature that offers an uniform interface to control printer's output device. Enscript protects all page device options inside an if block so they have no effect in level 1 interpreters. Although all level 2 interpreters support page device, they do not have to support all page device options. For example some printers can print in duplex mode and some can not. Refer to the documentation of your printer for supported options.

Here are some usable page device options which can be selected with the -D, --setpagedevice option. For a complete listing, see PostScript Language Reference Manual: section 4.11 Device Setup.

Collate boolean
how output is organized when printing multiple copies
Duplex boolean
duplex (two side) printing
ManualFeed boolean
manual feed paper tray
OutputFaceUp boolean
print output `face up' or `face down'
Tumble boolean
how opposite sides are positioned in duplex printing

PRINTING EXAMPLES

Following printing examples assume that enscript uses the default configuration. If default actions have been changed from the configuration files, some examples will behave differently.

enscript foo.txt
Print file foo.txt to the default printer.
enscript -Possu foo.txt
Print file foo.txt to printer ossu.
enscript -pfoo.ps foo.txt
Print file foo.txt, but leave PostScript output to file foo.ps.
enscript -2 foo.txt
Print file foo.txt to two columns.
enscript -2r foo.txt
Print file to two columns and rotate output 90 degrees (landscape).
enscript -DDuplex:true foo.txt
Print file in duplex (two side) mode (printer dependant).
enscript -G2rE -U2 foo.c
My default code printing command:
gaudy header, two columns, landscape,
code highlighting, 2-up printing.

ENVIRONMENT VARIABLES

The environment variable ENSCRIPT can be used to pass default options for enscript. For example, to select the default body font to be Times-Roman 7pt, set the following value to the ENSCRIPT environment variable:

-fTimes-Roman7

The value of the ENSCRIPT variable is processed before the command line options, so command line options can be used to overwrite these defaults.

Variable ENSCRIPT_LIBRARY specifies the enscript's library directory. It can be used to overwrite the build-in default `/usr/local/share/enscript'.

FILES

```
/usr/local/share/enscript/*.hdr         header files
/usr/local/share/enscript/*.enc         input encoding vectors
/usr/local/share/enscript/enscript.pro  PostScript prolog
/usr/local/share/enscript/*.afm         AFM files for PostScript fonts
/usr/local/share/enscript/font.map      index for the AFM files
/usr/local/share/enscript/enscript.st   states definition file
/usr/local/etc/enscript.cfg             system-wide configuration file
/usr/local/etc/enscriptsite.cfg         site configuration file
~/.enscriptrc                           personal configuration file
~/.enscript/                            personal resource directory
```

SEE ALSO

diffpp(1), ghostview(1), gs(1), lpq(1), lpr(1), lprm(1), states(1)

states

NAME

states - awk alike text processing tool

SYNOPSIS

states [-hV] [-D var=val] [-f file] [-o outputfile] [-s startstate] [-W level] [filename ...]

DESCRIPTION

States is an awk-alike text processing tool with some state machine extensions. It is designed for program source code highlighting and to similar tasks where state information helps input processing.

At a single point of time, States is in one state, each quite similar to awk's work environment, they have regular expressions which are matched from the input and actions which are executed when a match is found. From the action blocks, states can perform state transitions; it can move to another state from which the processing is continued. State transitions are recorded so states can return to the calling state once the current state has finished.

The biggest difference between states and awk, besides state machine extensions, is that states is not line-oriented. It matches regular expressions tokens from the input and once a match is processed, it continues processing from the current position, not from the beginning of the next input line.

OPTIONS

-D var=val, --define=var=val
: Define variable var to have string value val. Command line definitions overwrite variable definitions found from the config file.

-f file, --file=file
: Read state definitions from file file. As a default, states tries to read state definitions from file states.st in the current working directory.

-h, --help
: Print short help message and exit.

-o file, --output=file
: Save output to file file instead of printing it to stdout.

-s state, --state=state
: Start execution from state state. This definition overwrites start state resolved from the start block.

-V, --version
: Print states version and exit.

-W level, --warning=level
: Set the warning level to level.

Possible values for level are:

light
: light warnings (default)

all
: all warnings

PRIMITIVE FUNCTIONS

call (symbol)
: Move to state symbol and continue input file processing from that state. Function returns whatever the symbol state's terminating return statement returned.

check_namerules ()
: Try to resolve start state from namerules rules. Function returns 1 if start state was resolved or 0 otherwise.

check_startrules ()
: Try to resolve start state from startrules rules. Function returns 1 if start state was resolved or 0 otherwise.

concat (str, ...)
: Concanate argument strings and return result as a new string.

getenv (str)
: Get value of environment variable str. Returns an empty string if variable var is undefined.

int (any)
: Convert argument to an integer number.

length (item, ...)
: Count the length of argument strings or lists.

list (any, ...)
: Create a new list which contains items any, ...

panic (any, ...)
: Report a non-recoverable error and exit with status 1. Function never returns.

print (any, ...)
: Convert arguments to strings and print them to the output.

regmatch (string, regexp)
: Check if string string matches regular expression regexp. Functions returns a boolean success status and sets sub-expression registers $n.

regsub (string, regexp, subst)
: Search regular expression regexp from string string and replace the matching substring with string subst. Returns the resulting string.

regsuball (string, regexp, subst)
: Like regsub but replace all matches of regular expression regexp from string string with string subst.

sprintf (fmt, ...)
: Format arguments according to fmt and return result as a string.

strcmp (str1, str2)
: Perform a case-sensitive comparision for strings str1 and str2. Function returns a value which is:

-1 string str1 is less than str2
0 strings are equal
1 string str1 is greater than str2

string (any)
Convert argument to string.
strncmp (str1, str2, num)
Perform a casee-sensitive comparision for strings str1 and str2 comparing at maximum num characters.
substring (str, start, end)
Return a substring of string str starting from position start (inclusively) to end (exclusively).

BUILTIN VARIABLES

$n the nth parenthesized regular expression sub-expression from the latest state regular expression or from the regmatch primitive
filename
name of the current input file
program
name of the program (usually states)
version
program version string

FILES

```
/usr/local/share/enscript/enscript.st   enscript's states definitions
```

SEE ALSO

awk(1), enscript(1)