Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'tee.1p' command

$ man tee.1p

TEE(1P) POSIX Programmer's Manual TEE(1P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

tee ? duplicate standard input

SYNOPSIS

tee [-ai] [file...]

DESCRIPTION

The tee utility shall copy standard input to standard output, making a copy in zero or more files. The tee utility shall not buffer output.

If the -a option is not specified, output files shall be written (see Section 1.1.1.4, File Read, Write, and Creation.

OPTIONS

The tee utility shall conform to the Base Definitions volume of POSIX.1?2017, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-a Append the output to the files.

-i Ignore the SIGINT signal.

OPERANDS

The following operands shall be supported:

file A pathname of an output file. If a file operand is '-', it shall refer to a file named -; implementations shall not treat it as meaning standard output. Processing of at least 13 file operands shall be supported.

STDIN

The standard input can be of any type.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of tee:

LANG Provide a default value for the internationalization vari? ables that are unset or null. (See the Base Definitions vol? ume of POSIX.1?2017, Section 8.2, Internationalization Vari? ables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard er? ror.

NLSPATH Determine the location of message catalogs for the processing of LC_MESSAGES.

ASYNCHRONOUS EVENTS

Default, except that if the -i option was specified, SIGINT shall be ignored.

STDOUT

The standard output shall be a copy of the standard input.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

If any file operands are specified, the standard input shall be copied to each named file.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 The standard input was successfully copied to all output files.

>0 An error occurred.

CONSEQUENCES OF ERRORS

If a write to any successfully opened file operand fails, writes to other successfully opened file operands and standard output shall con? tinue, but the exit status shall be non-zero. Otherwise, the default actions specified in Section 1.4, Utility Description Defaults apply.

The following sections are informative.

APPLICATION USAGE

The tee utility is usually used in a pipeline, to make a copy of the output of some utility.

The file operand is technically optional, but tee is no more useful than cat when none is specified.

EXAMPLES

Save an unsorted intermediate form of the data in a pipeline:

```
... | tee unsorted | sort > sorted
```

RATIONALE

The buffering requirement means that tee is not allowed to use ISO C standard fully buffered or line-buffered writes. It does not mean that tee has to do 1-byte reads followed by 1-byte writes.

It should be noted that early versions of BSD ignore any invalid op? tions and accept a single '-' as an alternative to -i. They also print a message if unable to open a file:

```
"tee: cannot access %s\n", <pathname>
```

Historical implementations ignore write errors. This is explicitly not permitted by this volume of POSIX.1?2017.

Some historical implementations use O_APPEND when providing append mode; others use the lseek() function to seek to the end-of-file after opening the file without O_APPEND. This volume of POSIX.1?2017 requires functionality equivalent to using O_APPEND; see Section 1.1.1.4, File Read, Write, and Creation.

FUTURE DIRECTIONS

None.

SEE ALSO

Chapter 1, Introduction, cat

The Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables, Section 12.2, Utility Syntax Guidelines

The System Interfaces volume of POSIX.1?2017, lseek()