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 'ln.1p' command

$ man ln.1p

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

ln ? link files

SYNOPSIS

ln [-fs] [-L|-P] source_file target_file

ln [-fs] [-L|-P] source_file... target_dir

DESCRIPTION

In the first synopsis form, the ln utility shall create a new directory entry (link) at the destination path specified by the target_file oper? and. If the -s option is specified, a symbolic link shall be created for the file specified by the source_file operand. This first synopsis form shall be assumed when the final operand does not name an existing directory; if more than two operands are specified and the final is not an existing directory, an error shall result.

In the second synopsis form, the ln utility shall create a new direc? tory entry (link), or if the -s option is specified a symbolic link, for each file specified by a source_file operand, at a destination path in the existing directory named by target_dir.

If the last operand specifies an existing file of a type not specified by the System Interfaces volume of POSIX.1?2017, the behavior is imple? mentation-defined.

The corresponding destination path for each source_file shall be the concatenation of the target directory pathname, a <slash> character if the target directory pathname did not end in a <slash>, and the last pathname component of the source_file. The second synopsis form shall be assumed when the final operand names an existing directory.

For each source_file:

1. If the destination path exists and was created by a previous step, it is unspecified whether ln shall write a diagnostic message to standard error, do nothing more with the current source_file, and go on to any remaining source_files; or will continue processing the current source_file. If the destination path exists:
   a. If the -f option is not specified, ln shall write a diagnostic message to standard error, do nothing more with the current source_file, and go on to any remaining source_files.
   b. If the destination path names the same directory entry as the current source_file ln shall write a diagnostic message to standard error, do nothing more with the current source_file, and go on to any remaining source_files.
   c. Actions shall be performed equivalent to the unlink() function defined in the System Interfaces volume of POSIX.1?2017, called using the destination path as the path argument. If this fails for any reason, ln shall write a diagnostic message to standard error, do nothing more with the current source_file, and go on to any remaining source_files.
2. If the -s option is specified, actions shall be performed equiva? lent to the symlink() function with source_file as the path1 argu? ment and the destination path as the path2 argument. The ln utility shall do nothing more with source_file and shall go on to any re? maining files.
3. If source_file is a symbolic link:

a. If the -P option is in effect, actions shall be performed equivalent to the linkat() function with source_file as the path1 argument, the destination path as the path2 argument, AT_FDCWD as the fd1 and fd2 arguments, and zero as the flag ar? gument.

b. If the -L option is in effect, actions shall be performed equivalent to the linkat() function with source_file as the path1 argument, the destination path as the path2 argument, AT_FDCWD as the fd1 and fd2 arguments, and AT_SYMLINK_FOLLOW as the flag argument.

The ln utility shall do nothing more with source_file and shall go on to any remaining files.

4. Actions shall be performed equivalent to the link() function de? fined in the System Interfaces volume of POSIX.1?2017 using source_file as the path1 argument, and the destination path as the path2 argument.

OPTIONS

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

The following options shall be supported:

-f Force existing destination pathnames to be removed to allow the link.

-L For each source_file operand that names a file of type sym? bolic link, create a (hard) link to the file referenced by the symbolic link.

-P For each source_file operand that names a file of type sym? bolic link, create a (hard) link to the symbolic link itself.

-s Create symbolic links instead of hard links. If the -s option is specified, the -L and -P options shall be silently ig? nored.

Specifying more than one of the mutually-exclusive options -L and -P shall not be considered an error. The last option specified shall de? termine the behavior of the utility (unless the -s option causes it to

be ignored).

If the -s option is not specified and neither a -L nor a -P option is specified, it is implementation-defined which of the -L and -P options will be used as the default.

OPERANDS

The following operands shall be supported:

source_file

A pathname of a file to be linked. If the -s option is speci? fied, no restrictions on the type of file or on its existence shall be made. If the -s option is not specified, whether a directory can be linked is implementation-defined.

target_file

The pathname of the new directory entry to be created.

target_dir

A pathname of an existing directory in which the new direc? tory entries are created.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of ln:

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.

STDOUT

Not used.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 All the specified files were linked successfully.

>0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

None.

EXAMPLES

None.

RATIONALE

The CONSEQUENCES OF ERRORS section does not require ln -f a b to remove b if a subsequent link operation would fail.

Some historic versions of ln (including the one specified by the SVID) unlink the destination file, if it exists, by default. If the mode does not permit writing, these versions prompt for confirmation before at? tempting the unlink. In these versions the -f option causes ln not to

attempt to prompt for confirmation.

This allows ln to succeed in creating links when the target file al? ready exists, even if the file itself is not writable (although the di? rectory must be). Early proposals specified this functionality.

This volume of POSIX.1?2017 does not allow the ln utility to unlink ex? isting destination paths by default for the following reasons:

* The ln utility has historically been used to provide locking for shell applications, a usage that is incompatible with ln unlinking the destination path by default. There was no corresponding techni? cal advantage to adding this functionality.
* This functionality gave ln the ability to destroy the link struc? ture of files, which changes the historical behavior of ln.
* This functionality is easily replicated with a combination of rm and ln.
* It is not historical practice in many systems; BSD and BSD-derived systems do not support this behavior. Unfortunately, whichever be? havior is selected can cause scripts written expecting the other behavior to fail.
* It is preferable that ln perform in the same manner as the link() function, which does not permit the target to exist already.

This volume of POSIX.1?2017 retains the -f option to provide support for shell scripts depending on the SVID semantics. It seems likely that shell scripts would not be written to handle prompting by ln and would therefore have specified the -f option.

The -f option is an undocumented feature of many historical versions of the ln utility, allowing linking to directories. These versions require modification.

Early proposals of this volume of POSIX.1?2017 also required a -i op? tion, which behaved like the -i options in cp and mv, prompting for confirmation before unlinking existing files. This was not historical practice for the ln utility and has been omitted.

The -L and -P options allow for implementing both common behaviors of the ln utility. Earlier versions of this standard did not specify these

options and required the behavior now described for the -L option. Many systems by default or as an alternative provided a non-conforming ln utility with the behavior now described for the -P option. Since appli? cations could not rely on ln following links in practice, the -L and -P options were added to specify the desired behavior for the application.

The -L and -P options are ignored when -s is specified in order to al? low an alias to be created to alter the default behavior when creating hard links (for example, alias ln='ln -L'). They serve no purpose when -s is specified, since source_file is then just a string to be used as the contents of the created symbolic link and need not exist as a file.

The specification ensures that ln a a with or without the -f option will not unlink the file a. Earlier versions of this standard were un? clear in this case.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod, find, pax, rm

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, link(), unlink()

COPYRIGHT

nel.org/doc/man-pages/reporting_bugs.html .