**NAME**

ln – make hard or symbolic links to files

**SYNOPSIS**

**/usr/bin/ln [ –fns ]** *source_file* **[** *target* **]**
**/usr/bin/ln [ –fns ]** *source_file . . . target*

**/usr/xpg4/bin/ln [ –fs ]** *source_file* **[** *target* **]**
**/usr/xpg/bin/ln [ –fs ]** *source_file . . . target*

**AVAILABILITY**

**/usr/bin/ln**

SUNWcsu

**/usr/xpg4/bin/ln**

SUNWxcu4

**DESCRIPTION**

In the first synopsis form, the **ln** utility will create a new directory entry (link) for the file specified by *source_file*, at the destination path specified by *target*. If *target* is not specified, the link is made in the current directory. This first synopsis form is 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 will result.

In the second synopsis form, the **ln** utility will create a new directory entry for each file specified by a *source_file* operand, at a destination path in the existing directory named by *target*.

The **ln** utility may be used to create both hard links and symbolic links. A hard link is a pointer to a file and is indistinguishable from the original directory entry. Any changes to a file are effective independent of the name used to reference the file. Hard links may not span file systems and may not refer to directories.

**ln** by default creates hard links. *source_file* is linked to *target*. If *target* is a directory, another file named *source_file* is created in *target* and linked to the original *source_file*.

**/usr/bin/ln**

If *target* is a file, its contents are overwritten. If **ln** determines that the mode of *target* forbids writing, it will print the mode (see **chmod**(1)), ask for a response, and read the standard input for one line. If the line begins with **y**, the link occurs, if permissible; otherwise, the command exits.

**/usr/xpg4/bin/ln**

If *target* is a file and the –**f** option is not specified, **ln** will write a diagnostic message to standard error, do nothing more with the current *source_file*, and go on to any remaining *source_file*s.

A symbolic link is an indirect pointer to a file; its directory entry contains the name of the file to which it is linked. Symbolic links may span file systems and may refer to directories.

**OPTIONS**

The following options are supported:

–**f** Link files without questioning the user, even if the mode of *target* forbids writing. This is the default if the standard input is not a terminal.

**/usr/bin/ln**

**−n** If the link is an existing file, do not overwrite the contents of the file. The **−f** option overrides this option. This is the default behavior for **/usr/xpg4/bin/ln**, and is silently ignored.

**−s** Create a symbolic link.

If the **−s** option is used with two arguments, *target* may be an existing directory or a non-existent file. If *target* already exists and is not a directory, an error is returned. *source_file* may be any path name and need not exist. If it exists, it may be a file or directory and may reside on a different file system from *target*. If *target* is an existing directory, a file is created in directory *target* whose name is *source_file* or the last component of *source_file*. This file is a symbolic link that references *source_file*. If *target* does not exist, a file with name *target* is created and it is a symbolic link that references *source_file*.

If the **−s** option is used with more than two arguments, *target* must be an existing directory or an error will be returned. For each *source_file*, a link is created in *target* whose name is the last component of *source_file*; each new *source_file* is a symbolic link to the original *source_file*. The files and *target* may reside on different file systems.

File permissions for *target* may be different from those displayed with a **−l** listing of the **ls**(1) command. To display the permissions of *target* use **ls −lL**. See **stat**(2) for more information.

## OPERANDS

The following operands are supported:

*source_file* A path name of a file to be linked. This can be either a regular or special file. If the **−s** option is specified, *source_file* can also be a directory.

*target* The path name of the new directory entry to be created, or of an existing directory in which the new directory entries are to be created.

## ENVIRONMENT

See **environ**(5) for descriptions of the following environment variables that affect the execution of **ln**: **LC_CTYPE**, **LC_MESSAGES**, and **NLSPATH**.

## EXIT STATUS

The following exit values are returned:

**0** All the specified files were linked successfully

**>0** An error occurred.

## SEE ALSO

**chmod**(1), **ls**(1), **stat**(2), **environ**(5)

## NOTES

A symbolic link to a directory behaves differently than you might expect in certain cases. While an **ls**(1) on such a link displays the files in the pointed-to directory, an '**ls −l**' displays information about the link itself:

```
example% ln −s dir link
example% ls link
file1 file2 file3 file4
example% ls −l link
```

```
lrwxrwxrwx  1 user          7 Jan 11 23:27 link -> dir
```

When you **cd**(1) to a directory through a symbolic link, you wind up in the pointed-to location within the file system. This means that the parent of the new working directory is not the parent of the symbolic link, but rather, the parent of the pointed-to directory. For instance, in the following case the final working directory is **/usr** and not **/home/user/linktest**.

```
example% pwd
/home/user/linktest
example% ln –s /usr/tmp symlink
example% cd symlink
example% cd . .
example% pwd
/usr
```

C shell user's can avoid any resulting navigation problems by using the **pushd** and **popd** built-in commands instead of **cd**.