NAME

symlink, **symlinkat** - make symbolic link to a file

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <unistd.h>

int
symlink(*const char *name1, const char *name2*);

int
symlinkat(*const char *name1, int fd, const char *name2*);

DESCRIPTION

A symbolic link *name2* is created to *name1* (*name2* is the name of the file created, *name1* is the string used in creating the symbolic link). Either name may be an arbitrary path name; the files need not be on the same file system.

The **symlinkat**() system call is equivalent to **symlink**() except in the case where *name2* specifies a relative path. In this case the symbolic link is created relative to the directory associated with the file descriptor *fd* instead of the current working directory. If **symlinkat**() is passed the special value AT_FDCWD in the *fd* parameter, the current working directory is used and the behavior is identical to a call to **symlink**().

RETURN VALUES

The **symlink**() function returns the value 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

ERRORS

The symbolic link succeeds unless:

[ENOTDIR] A component of the *name2* path prefix is not a directory.

[ENAMETOOLONG]
A component of the *name2* pathname exceeded 255 characters, or the entire length of either path name exceeded 1023 characters.

[ENOENT] A component of the *name2* path prefix does not exist.

[EACCES] A component of the *name2* path prefix denies search permission, or write permission is denied on the parent directory of the file to be created.

[ELOOP] Too many symbolic links were encountered in translating the *name2* path name.

[EEXIST] The path name pointed at by the *name2* argument already exists.

[EPERM] The parent directory of the file named by *name2* has its immutable flag set, see the chflags(2) manual page for more information.

[EIO] An I/O error occurred while making the directory entry for *name2*, allocating the inode for *name2*, or writing out the link contents of *name2*.

[EROFS] The file *name2* would reside on a read-only file system.

[ENOSPC] The directory in which the entry for the new symbolic link is being placed cannot be extended because there is no space left on the file system containing the directory.

[ENOSPC] The new symbolic link cannot be created because there is no space left on the file system that will contain the symbolic link.

[ENOSPC] There are no free inodes on the file system on which the symbolic link is being created.

[EDQUOT] The directory in which the entry for the new symbolic link is being placed cannot be extended because the user’s quota of disk blocks on the file system containing the directory has been exhausted.

[EDQUOT] The new symbolic link cannot be created because the user’s quota of disk blocks on the file system that will contain the symbolic link has been exhausted.

[EDQUOT] The user’s quota of inodes on the file system on which the symbolic link is being created has been exhausted.

[EINTEGRITY] Corrupted data was detected while reading from the file system.

[EFAULT] The *name1* or *name2* argument points outside the process’s allocated address space.

In addition to the errors returned by the **symlink**(), the **symlinkat**() may fail if:

[EBADF] The *name2* argument does not specify an absolute path and the *fd* argument is neither AT_FDCWD nor a valid file descriptor open for searching.

[ENOTDIR] The *name2* argument is not an absolute path and *fd* is neither AT_FDCWD nor a file descriptor associated with a directory.

SEE ALSO

ln(1), chflags(2), link(2), lstat(2), readlink(2), unlink(2), symlink(7)

STANDARDS

The **symlinkat**() system call follows The Open Group Extended API Set 2 specification.

HISTORY

The **symlink**() system call appeared in 4.2BSD. The **symlinkat**() system call appeared in FreeBSD 8.0.