NAME

chown, **fchown**, **lchown**, **fchownat** - change owner and group of a file

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <unistd.h>

int
chown(*const char *path, uid_t owner, gid_t group*);

int
fchown(*int fd, uid_t owner, gid_t group*);

int
lchown(*const char *path, uid_t owner, gid_t group*);

int
fchownat(*int fd, const char *path, uid_t owner, gid_t group, int flag*);

DESCRIPTION

The owner ID and group ID of the file named by *path* or referenced by *fd* is changed as specified by the arguments *owner* and *group*. The owner of a file may change the *group* to a group of which he or she is a member, but the change *owner* capability is restricted to the super-user.

The **chown**() system call clears the set-user-id and set-group-id bits on the file to prevent accidental or mischievous creation of set-user-id and set-group-id programs if not executed by the super-user. The **chown**() system call follows symbolic links to operate on the target of the link rather than the link itself.

The **fchown**() system call is particularly useful when used in conjunction with the file locking primitives (see flock(2)).

The **lchown**() system call is similar to **chown**() but does not follow symbolic links.

The **fchownat**() system call is equivalent to the **chown**() and **lchown**() except in the case where *path* specifies a relative path. In this case the file to be changed is determined relative to the directory associated with the file descriptor *fd* instead of the current working directory.

Values for *flag* are constructed by a bitwise-inclusive OR of flags from the following list, defined in

<*fcntl.h*>:

AT_SYMLINK_NOFOLLOW
If *path* names a symbolic link, ownership of the symbolic link is changed.

AT_RESOLVE_BENEATH
Only walk paths below the directory specified by the *fd* descriptor. See the description of the O_RESOLVE_BENEATH flag in the open(2) manual page.

AT_EMPTY_PATH
If the *path* argument is an empty string, operate on the file or directory referenced by the descriptor *fd*. If *fd* is equal to AT_FDCWD, operate on the current working directory.

If **fchownat**() 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 **chown**() or **lchown**() respectively, depending on whether or not the AT_SYMLINK_NOFOLLOW bit is set in the *flag* argument.

One of the owner or group id's may be left unchanged by specifying it as -1.

RETURN VALUES

Upon successful completion, the value 0 is returned; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

ERRORS

The **chown**() and **lchown**() will fail and the file will be unchanged if:

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

[ENAMETOOLONG]
A component of a pathname exceeded 255 characters, or an entire path name exceeded 1023 characters.

[ENOENT] The named file does not exist.

[EACCES] Search permission is denied for a component of the path prefix.

[ELOOP] Too many symbolic links were encountered in translating the pathname.

[EPERM] The operation would change the ownership, but the effective user ID is not the super-user.

[EPERM] The named file has its immutable or append-only flag set, see the chflags(2) manual page for more information.

[EROFS] The named file resides on a read-only file system.

[EFAULT] The *path* argument points outside the process's allocated address space.

[EIO] An I/O error occurred while reading from or writing to the file system.

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

The **fchown**() system call will fail if:

[EBADF] The *fd* argument does not refer to a valid descriptor.

[EINVAL] The *fd* argument refers to a socket, not a file.

[EPERM] The effective user ID is not the super-user.

[EROFS] The named file resides on a read-only file system.

[EIO] An I/O error occurred while reading from or writing to the file system.

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

In addition to the errors specified for **chown**() and **lchown**(), the **fchownat**() system call may fail if:

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

[EINVAL] The value of the *flag* argument is not valid.

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

[ENOTCAPABLE] *path* is an absolute path, or contained a ".." component leading to a directory outside of the directory hierarchy specified by *fd*, and the process is in capability mode or the AT_RESOLVE_BENEATH flag was specified.

SEE ALSO

chgrp(1), chflags(2), chmod(2), flock(2), chown(8)

STANDARDS

The **chown**() system call is expected to conform to IEEE Std 1003.1-1990 ("POSIX.1"). The **fchownat**() system call follows The Open Group Extended API Set 2 specification.

HISTORY

The **chown**() function appeared in Version 1 AT&T UNIX. The **fchown**() system call appeared in 4.2BSD.

The **chown**() system call was changed to follow symbolic links in 4.4BSD. The **lchown**() system call was added in FreeBSD 3.0 to compensate for the loss of functionality.

The **fchownat**() system call appeared in FreeBSD 8.0.