NAME

unlink − delete a name and possibly the file it refers to

SYNOPSIS

#include <unistd.h>

int unlink(const char * *pathname***);**

DESCRIPTION

unlink() deletes a name from the file system. If that name was the last link to a file and no processes have the file open the file is deleted and the space it was using is made available for reuse.

If the name was the last link to a file but any processes still have the file open the file will remain in existence until the last file descriptor referring to it is closed.

If the name referred to a symbolic link the link is removed.

If the name referred to a socket, fifo or device the name for it is removed but processes which have the object open may continue to use it.

RETURN VALUE

On success, zero is returned. On error, −1 is returned, and *errno* is set appropriately.

ERRORS

EACCES
Write access to the directory containing *pathname* is not allowed for the process's effective UID, or one of the directories in *pathname* did not allow search permission. (See also **path_resolution**(7).)

EBUSY (not on Linux)
The file *pathname* cannot be unlinked because it is being used by the system or another process and the implementation considers this an error.

EFAULT
pathname points outside your accessible address space.

EIO An I/O error occurred.

EISDIR
pathname refers to a directory. (This is the non-POSIX value returned by Linux since 2.1.132.)

ELOOP
Too many symbolic links were encountered in translating *pathname*.

ENAMETOOLONG
pathname was too long.

ENOENT
A component in *pathname* does not exist or is a dangling symbolic link, or *pathname* is empty.

ENOMEM
Insufficient kernel memory was available.

ENOTDIR
A component used as a directory in *pathname* is not, in fact, a directory.

EPERM
The system does not allow unlinking of directories, or unlinking of directories requires privileges that the calling process doesn't have. (This is the POSIX prescribed error return; as noted above, Linux returns **EISDIR** for this case.)

EPERM (Linux only)
The file system does not allow unlinking of files.

EPERM or **EACCES**
The directory containing *pathname* has the sticky bit (**S_ISVTX**) set and the process's effective UID is neither the UID of the file to be deleted nor that of the directory containing it, and the process is not privileged (Linux: does not have the **CAP_FOWNER** capability).

EROFS
pathname refers to a file on a read-only file system.

CONFORMING TO
SVr4, 4.3BSD, POSIX.1-2001.

BUGS
Infelicities in the protocol underlying NFS can cause the unexpected disappearance of files which are still being used.

SEE ALSO
rm(1), **chmod**(2), **link**(2), **mknod**(2), **open**(2), **rename**(2), **rmdir**(2), **unlinkat**(2), **mkfifo**(3), **remove**(3), **path_resolution**(7), **symlink**(7)

COLOPHON
This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.