NAME

truncate, ftruncate − truncate a file to a specified length

SYNOPSIS

#include <unistd.h>
#include <sys/types.h>

int truncate(const char ****path*, off_t** *length***);**
int ftruncate(int *fd***, off_t** *length***);**

Feature Test Macro Requirements for glibc (see **feature_test_macros**(7)):

truncate(): _BSD_SOURCE || _XOPEN_SOURCE >= 500
ftruncate(): _BSD_SOURCE || _XOPEN_SOURCE >= 500 || _POSIX_C_SOURCE >= 200112L

DESCRIPTION

The **truncate**() and **ftruncate**() functions cause the regular file named by *path* or referenced by *fd* to be truncated to a size of precisely *length* bytes.

If the file previously was larger than this size, the extra data is lost. If the file previously was shorter, it is extended, and the extended part reads as null bytes ('\0').

The file offset is not changed.

If the size changed, then the st_ctime and st_mtime fields (respectively, time of last status change and time of last modification; see **stat**(2)) for the file are updated, and the set-user-ID and set-group-ID permission bits may be cleared.

With **ftruncate**(), the file must be open for writing; with **truncate**(), the file must be writable.

RETURN VALUE

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

ERRORS

For **truncate**():

EACCES
Search permission is denied for a component of the path prefix, or the named file is not writable by the user. (See also **path_resolution**(7).)

EFAULT
Path points outside the process's allocated address space.

EFBIG
The argument *length* is larger than the maximum file size. (XSI)

EINTR
A signal was caught during execution.

EINVAL
The argument *length* is negative or larger than the maximum file size.

EIO An I/O error occurred updating the inode.

EINTR
While blocked waiting to complete, the call was interrupted by a signal handler; see **fcntl**(2) and **signal**(7).

EISDIR
The named file is a directory.

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

ENAMETOOLONG
A component of a pathname exceeded 255 characters, or an entire pathname exceeded 1023 characters.

ENOENT
The named file does not exist.

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

EPERM
The underlying file system does not support extending a file beyond its current size.

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

ETXTBSY
The file is a pure procedure (shared text) file that is being executed.

For **ftruncate**() the same errors apply, but instead of things that can be wrong with *path*, we now have things that can be wrong with the file descriptor, *fd*:

EBADF
fd is not a valid descriptor.

EBADF or **EINVAL**
fd is not open for writing.

EINVAL
fd does not reference a regular file.

CONFORMING TO

4.4BSD, SVr4, POSIX.1-2001 (these calls first appeared in 4.2BSD).

NOTES

The above description is for XSI-compliant systems. For non-XSI-compliant systems, the POSIX standard allows two behaviors for **ftruncate**() when *length* exceeds the file length (note that **truncate**() is not specified at all in such an environment): either returning an error, or extending the file. Like most Unix implementations, Linux follows the XSI requirement when dealing with native file systems. However, some non-native file systems do not permit **truncate**() and **ftruncate**() to be used to extend a file beyond its current length: a notable example on Linux is VFAT.

SEE ALSO

open(2), **stat**(2), **path_resolution**(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/.