Red Hat Enterprise Linux Release 9.2 Manual Pages on 'fdatasync.2' command

$ man fdatasync.2

FSYNC(2)   Linux Programmer's Manual   FSYNC(2)

NAME

    fsync, fdatasync - synchronize a file's in-core state with storage de?

    vice

SYNOPSIS

    #include <unistd.h>

    int fsync(int fd);

    int fdatasync(int fd);

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

    fsync():

      Glibc 2.16 and later:

        No feature test macros need be defined

      Glibc up to and including 2.15:

        _BSD_SOURCE || _XOPEN_SOURCE

          || /* since glibc 2.8: */ _POSIX_C_SOURCE >= 200112L

    fdatasync():

      _POSIX_C_SOURCE >= 199309L || _XOPEN_SOURCE >= 500

DESCRIPTION

    fsync() transfers ("flushes") all modified in-core data of (i.e., modi?

    fied buffer cache pages for) the file referred to by the file descrip?

    tor fd to the disk device (or other permanent storage device) so that

    all changed information can be retrieved even if the system crashes or

    is rebooted. This includes writing through or flushing a disk cache if

present. The call blocks until the device reports that the transfer has completed.

As well as flushing the file data, fsync() also flushes the metadata information associated with the file (see inode(7)).

Calling fsync() does not necessarily ensure that the entry in the di‐ rectory containing the file has also reached disk.   For that an ex‐ plicit fsync() on a file descriptor for the directory is also needed.

fdatasync() is similar to fsync(), but does not flush modified metadata unless that metadata is needed in order to allow a subsequent data re‐ trieval to be correctly handled. For example, changes to st_atime or st_mtime (respectively, time of last access and time of last modifica‐ tion; see inode(7)) do not require flushing because they are not neces‐ sary for a subsequent data read to be handled correctly. On the other hand, a change to the file size (st_size, as made by say ftruncate(2)), would require a metadata flush.

The aim of fdatasync() is to reduce disk activity for applications that do not require all metadata to be synchronized with the disk.

RETURN VALUE

On success, these system calls return zero. On error, -1 is returned, and errno is set appropriately.

ERRORS

EBADF fd is not a valid open file descriptor.

EIO   An error occurred during synchronization. This error may relate to data written to some other file descriptor on the same file. Since Linux 4.13, errors from write-back will be reported to all file descriptors that might have written the data which trig‐ gered the error. Some filesystems (e.g., NFS) keep close track of which data came through which file descriptor, and give more precise reporting. Other filesystems (e.g., most local filesys‐ tems) will report errors to all file descriptors that were open on the file when the error was recorded.

ENOSPC Disk space was exhausted while synchronizing.

EROFS, EINVAL

fd is bound to a special file (e.g., a pipe, FIFO, or socket)

which does not support synchronization.

ENOSPC, EDQUOT

fd is bound to a file on NFS or another filesystem which does

not allocate space at the time of a write(2) system call, and

some previous write failed due to insufficient storage space.

CONFORMING TO

POSIX.1-2001, POSIX.1-2008, 4.3BSD.

On POSIX systems on which fdatasync() is available, _POSIX_SYNCHRO?

NIZED_IO is defined in <unistd.h> to a value greater than 0. (See also

sysconf(3).)

NOTES

On some UNIX systems (but not Linux), fd must be a writable file de?

scriptor.

In Linux 2.2 and earlier, fdatasync() is equivalent to fsync(), and so

has no performance advantage.

The fsync() implementations in older kernels and lesser used filesys?

tems do not know how to flush disk caches. In these cases disk caches

need to be disabled using hdparm(8) or sdparm(8) to guarantee safe op?

eration.

SEE ALSO

sync(1), bdflush(2), open(2), posix_fadvise(2), pwritev(2), sync(2),

sync_file_range(2), fflush(3), fileno(3), hdparm(8), mount(8)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A

description of the project, information about reporting bugs, and the

latest   version   of   this   page,   can   be   found   at

https://www.kernel.org/doc/man-pages/.

Linux   2020-06-09   FSYNC(2)