

**NAME**

`fsync`, `fdatasync` - synchronize a file's in-core state with storage device

**SYNOPSIS**

```
#include <unistd.h>

int fsync(int fd);

int fdatasync(int fd);
```

Feature Test Macro Requirements for glibc (see [feature\\_test\\_macros\(7\)](#)):

```
fsync(): _BSD_SOURCE || _XOPEN_SOURCE
|| /* since glibc 2.8: */ _POSIX_C_SOURCE >= 200112L
fdatasync(): _POSIX_C_SOURCE >= 199309L || _XOPEN_SOURCE >= 500
```

**DESCRIPTION**

**fsync**(*fd*) transfers (flushes) all modified in-core data of (i.e., modified buffer cache pages for) the file referred to by the file descriptor *fd* to the disk device (or other permanent storage device) so that all changed information can be retrieved even after the system crashed or was rebooted. This includes writing through or flushing a disk cache if present. The call blocks until the device reports that the transfer has completed. It also flushes metadata information associated with the file (see [stat\(2\)](#)).

Calling **fsync**(*fd*) does not necessarily ensure that the entry in the directory containing the file has also reached disk. For that an explicit **fsync**(*fd*) on a file descriptor for the directory is also needed.

**fdatasync**(*fd*) is similar to **fsync**(*fd*), but does not flush modified metadata unless that metadata is needed in order to allow a subsequent data retrieval to be correctly handled. For example, changes to *st\_atime* or *st\_mtime* (respectively, time of last access and time of last modification; see [stat\(2\)](#)) do not require flushing because they are not necessary 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 [truncate\(2\)](#)), would require a metadata flush.

The aim of **fdatasync**(*fd*) 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.

**EROFS, EINVAL**

*fd* is bound to a special file which does not support synchronization.

**CONFORMING TO**

4.3BSD, POSIX.1-2001.

**AVAILABILITY**

On POSIX systems on which **fdatasync**(*fd*) is available, `_POSIX_SYNCHRONIZED_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 descriptor.

In Linux 2.2 and earlier, **fdatasync**(*fd*) is equivalent to **fsync**(*fd*), and so has no performance advantage.

The **fsync**(*fd*) implementations in older kernels and lesser used filesystems does 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 operation.

**SEE ALSO**

[bdflush\(2\)](#), [open\(2\)](#), [sync\(2\)](#), [sync\\_file\\_range\(2\)](#), [hdparm\(8\)](#), [mount\(8\)](#), [sync\(1\)](#)

**COLOPHON**

This page is part of release 3.74 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 <http://www.kernel.org/doc/man-pages/>.