

**NAME**

`msync` - synchronize a file with a memory map

**SYNOPSIS**

```
#include <sys/mman.h>
```

```
int msync(void *addr, size_t length, int flags);
```

**DESCRIPTION**

`msync()` flushes changes made to the in-core copy of a file that was mapped into memory using [mmap\(2\)](#) back to the filesystem. Without use of this call there is no guarantee that changes are written back before [munmap\(2\)](#) is called. To be more precise, the part of the file that corresponds to the memory area starting at *addr* and having length *length* is updated.

The *flags* argument may have the bits `MS_ASYNC`, `MS_SYNC`, and `MS_INVALIDATE` set, but not both `MS_ASYNC` and `MS_SYNC`. `MS_ASYNC` specifies that an update be scheduled, but the call returns immediately. `MS_SYNC` asks for an update and waits for it to complete. `MS_INVALIDATE` asks to invalidate other mappings of the same file (so that they can be updated with the fresh values just written).

**RETURN VALUE**

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

**ERRORS****EBUSY**

`MS_INVALIDATE` was specified in *flags*, and a memory lock exists for the specified address range.

**EINVAL**

*addr* is not a multiple of `PAGESIZE`; or any bit other than `MS_ASYNC` | `MS_INVALIDATE` | `MS_SYNC` is set in *flags*; or both `MS_SYNC` and `MS_ASYNC` are set in *flags*.

**ENOMEM**

The indicated memory (or part of it) was not mapped.

**CONFORMING TO**

POSIX.1-2001.

This call was introduced in Linux 1.3.21, and then used **EFAULT** instead of **ENOMEM**. In Linux 2.4.19, this was changed to the POSIX value **ENOMEM**.

**AVAILABILITY**

On POSIX systems on which `msync()` is available, both `_POSIX_MAPPED_FILES` and `_POSIX_SYNCHRONIZED_IO` are defined in `<unistd.h>` to a value greater than 0. (See also [sysconf\(3\)](#).)

**NOTES**

According to POSIX, either `MS_SYNC` or `MS_ASYNC` must be specified in *flags*, and indeed failure to include one of these flags will cause `msync()` to fail on some systems. However, Linux permits a call to `msync()` that specifies neither of these flags, with semantics that are (currently) equivalent to specifying `MS_ASYNC`. (Since Linux 2.6.19, `MS_ASYNC` is in fact a no-op, since the kernel properly tracks dirty pages and flushes them to storage as necessary.) Notwithstanding the Linux behavior, portable, future-proof applications should ensure that they specify either `MS_SYNC` or `MS_ASYNC` in *flags*.

**SEE ALSO**

[mmap\(2\)](#)

B.O. Gallmeister, POSIX.4, O'Reilly, pp. 128-129 and 389-391.

**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/>.