Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'msync.3p' command

$ man msync.3p

MSYNC(3P) POSIX Programmer's Manual MSYNC(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

msync ? synchronize memory with physical storage

SYNOPSIS

```
#include <sys/mman.h>
int msync(void *addr, size_t len, int flags);
```

DESCRIPTION

The msync() function shall write all modified data to permanent storage locations, if any, in those whole pages containing any part of the ad? dress space of the process starting at address addr and continuing for len bytes. If no such storage exists, msync() need not have any effect. If requested, the msync() function shall then invalidate cached copies of data.

The implementation may require that addr be a multiple of the page size as returned by sysconf().

For mappings to files, the msync() function shall ensure that all write operations are completed as defined for synchronized I/O data integrity completion. It is unspecified whether the implementation also writes

out other file attributes. When the msync() function is called on MAP_PRIVATE mappings, any modified data shall not be written to the un? derlying object and shall not cause such data to be made visible to other processes. It is unspecified whether data in MAP_PRIVATE mappings has any permanent storage locations. The effect of msync() on a shared memory object or a typed memory object is unspecified. The behavior of this function is unspecified if the mapping was not established by a call to mmap().

The flags argument is constructed from the bitwise-inclusive OR of one or more of the following flags defined in the <sys/mman.h> header:

Symbolic Constant	Description
MS_ASYNC	Perform asynchronous writes.
MS_SYNC	Perform synchronous writes.
MS_INVALIDATE	Invalidate cached data.

When MS_ASYNC is specified, msync() shall return immediately once all the write operations are initiated or queued for servicing; when MS_SYNC is specified, msync() shall not return until all write opera? tions are completed as defined for synchronized I/O data integrity com? pletion. Either MS_ASYNC or MS_SYNC shall be specified, but not both.

When MS_INVALIDATE is specified, msync() shall invalidate all cached copies of mapped data that are inconsistent with the permanent storage locations such that subsequent references shall obtain data that was consistent with the permanent storage locations sometime between the call to msync() and the first subsequent memory reference to the data.

If msync() causes any write to a file, the file's last data modifica? tion and last file status change timestamps shall be marked for update.

RETURN VALUE

Upon successful completion, msync() shall return 0; otherwise, it shall return -1 and set errno to indicate the error.

ERRORS

The msync() function shall fail if:

EBUSY Some or all of the addresses in the range starting at addr and continuing for len bytes are locked, and MS_INVALIDATE is speci? fied.

EINVAL The value of flags is invalid.

ENOMEM The addresses in the range starting at addr and continuing for len bytes are outside the range allowed for the address space of a process or specify one or more pages that are not mapped.

The msync() function may fail if:

EINVAL The value of addr is not a multiple of the page size as returned by sysconf().

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

The msync() function is only supported if the Synchronized Input and Output option is supported, and thus need not be available on all im? plementations.

The msync() function should be used by programs that require a memory object to be in a known state; for example, in building transaction fa? cilities.

Normal system activity can cause pages to be written to disk. There? fore, there are no guarantees that msync() is the only control over when pages are or are not written to disk.

RATIONALE

The msync() function writes out data in a mapped region to the perma? nent storage for the underlying object. The call to msync() ensures data integrity of the file.

After the data is written out, any cached data may be invalidated if the MS_INVALIDATE flag was specified. This is useful on systems that do not support read/write consistency.

FUTURE DIRECTIONS

None.

SEE ALSO

mmap(), sysconf()

The Base Definitions volume of POSIX.1?2017, <sys_mman.h>