



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 'munmap.3p' command

\$ man munmap.3p

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

munmap ? unmap pages of memory

SYNOPSIS

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

int munmap(void *addr, size_t len);
```

DESCRIPTION

The munmap() function shall remove any mappings for those entire pages containing any part of the address space of the process starting at addr and continuing for len bytes. Further references to these pages shall result in the generation of a SIGSEGV signal to the process. If there are no mappings in the specified address range, then munmap() has no effect.

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

If a mapping to be removed was private, any modifications made in this address range shall be discarded.

Any memory locks (see mlock() and mlockall()) associated with this ad?

dress range shall be removed, as if by an appropriate call to `munlock()`.

If a mapping removed from a typed memory object causes the corresponding address range of the memory pool to be inaccessible by any process in the system except through allocatable mappings (that is, mappings of typed memory objects opened with the `POSIX_TYPED_MEM_MAP_ALLOCATABLE` flag), then that range of the memory pool shall become deallocated and may become available to satisfy future typed memory allocation requests.

A mapping removed from a typed memory object opened with the `POSIX_TYPED_MEM_MAP_ALLOCATABLE` flag shall not affect in any way the availability of that typed memory for allocation.

The behavior of this function is unspecified if the mapping was established by a call to `mmap()`.

RETURN VALUE

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

ERRORS

The `munmap()` function shall fail if:

`EINVAL` Addresses in the range `[addr,addr+len)` are outside the valid range for the address space of a process.

`EINVAL` The `len` argument is 0.

The `munmap()` function may fail if:

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

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

None.

RATIONALE

The `munmap()` function corresponds to SVR4, just as the `mmap()` function does.

It is possible that an application has applied process memory locking to a region that contains shared memory. If this has occurred, the `mmap()` call ignores those locks and, if necessary, causes those locks to be removed.

Most implementations require that `addr` is a multiple of the page size as returned by `sysconf()`.

FUTURE DIRECTIONS

None.

SEE ALSO

`mlock()`, `mlockall()`, `mmap()`, `posix_typed_mem_open()`, `sysconf()`

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

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html>.

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.kernel.org/doc/man-pages/reporting_bugs.html.