NAME

posix_madvise – give advice about patterns of memory usage

SYNOPSIS

#include <sys/mman.h>

int posix_madvise(void **addr*, size_t *len*, int *advice*);

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

posix_madvise():
_POSIX_C_SOURCE >= 200112L

DESCRIPTION

The **posix_madvise**() function allows an application to advise the system about its expected patterns of usage of memory in the address range starting at *addr* and continuing for *len* bytes. The system is free to use this advice in order to improve the performance of memory accesses (or to ignore the advice altogether), but calling **posix_madvise**() shall not affect the semantics of access to memory in the specified range.

The *advice* argument is one of the following:

POSIX_MADV_NORMAL
The application has no special advice regarding its memory usage patterns for the specified address range. This is the default behavior.

POSIX_MADV_SEQUENTIAL
The application expects to access the specified address range sequentially, running from lower addresses to higher addresses. Hence, pages in this region can be aggressively read ahead, and may be freed soon after they are accessed.

POSIX_MADV_RANDOM
The application expects to access the specified address range randomly. Thus, read ahead may be less useful than normally.

POSIX_MADV_WILLNEED
The application expects to access the specified address range in the near future. Thus, read ahead may be beneficial.

POSIX_MADV_DONTNEED
The application expects that it will not access the specified address range in the near future.

RETURN VALUE

On success, **posix_madvise**() returns 0. On failure, it returns a positive error number.

ERRORS

EINVAL
addr is not a multiple of the system page size or *len* is negative.

EINVAL
advice is invalid.

ENOMEM
Addresses in the specified range are partially or completely outside the caller's address space.

VERSIONS

Support for **posix_madvise**() first appeared in glibc version 2.2.

CONFORMING TO

POSIX.1-2001.

NOTES

POSIX.1 permits an implementation to generate an error if *len* is 0. On Linux, specifying *len* as 0 is permitted (as a successful no-op).

In glibc, this function is implemented using **madvise**(2). However, since glibc 2.6, **POSIX_MADV_DONTNEED** is treated as a no-op, because the corresponding **madvise**(2) value,

MADV_DONTNEED, has destructive semantics.

SEE ALSO

madvise(2), **posix_fadvise**(2)

COLOPHON

This page is part of release 5.02 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/.