NAME

pthread_mutex_consistent − make a robust mutex consistent

LIBRARY

POSIX threads library (*libpthread*, *−lpthread*)

SYNOPSIS

#include <pthread.h>

int pthread_mutex_consistent(pthread_mutex_t **mutex*);

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

pthread_mutex_consistent():
_POSIX_C_SOURCE >= 200809L

DESCRIPTION

This function makes a robust mutex consistent if it is in an inconsistent state. A mutex can be left in an inconsistent state if its owner terminates while holding the mutex, in which case the next owner who acquires the mutex will succeed and be notified by a return value of **EOWNERDEAD** from a call to **pthread_mutex_lock**().

RETURN VALUE

On success, *pthread_mutex_consistent*() returns 0. Otherwise, it returns a positive error number to indicate the error.

ERRORS

EINVAL
The mutex is either not robust or is not in an inconsistent state.

VERSIONS

pthread_mutex_consistent() was added in glibc 2.12.

STANDARDS

POSIX.1-2008.

NOTES

pthread_mutex_consistent() simply informs the implementation that the state (shared data) guarded by the mutex has been restored to a consistent state and that normal operations can now be performed with the mutex. It is the application's responsibility to ensure that the shared data has been restored to a consistent state before calling **pthread_mutex_consistent**().

Before the addition of **pthread_mutex_consistent**() to POSIX, glibc defined the following equivalent nonstandard function if **_GNU_SOURCE** was defined:

[[deprecated]]
int pthread_mutex_consistent_np(const pthread_mutex_t **mutex*);

This GNU-specific API, which first appeared in glibc 2.4, is nowadays obsolete and should not be used in new programs; since glibc 2.34 it has been marked as deprecated.

EXAMPLES

See **pthread_mutexattr_setrobust**(3).

SEE ALSO

pthread_mutex_lock(3), **pthread_mutexattr_getrobust**(3), **pthread_mutexattr_init**(3), **pthread_mutexattr_setrobust**(3), **pthreads**(7)