NAME

mq_notify - notify process that a message is available (REALTIME)

LIBRARY

POSIX Real-time Library (librt, -lrt)

SYNOPSIS

#include <mqueue.h>

int
mq_notify(*mqd_t mqdes*, *const struct sigevent *notification*);

DESCRIPTION

If the argument notification is not NULL, this system call will register the calling process to be notified of message arrival at an empty message queue associated with the specified message queue descriptor, *mqdes*. The notification specified by the *notification* argument will be sent to the process when the message queue transitions from empty to non-empty. At any time, only one process may be registered for notification by a message queue. If the calling process or any other process has already registered for notification of message arrival at the specified message queue, subsequent attempts to register for that message queue will fail.

The *notification* argument points to a *sigevent* structure that defines how the calling process will be notified. If *notification->sigev_notify* is SIGEV_NONE, then no signal will be posted, but the error status and the return status for the operation will be set appropriately. For SIGEV_SIGNO and SIGEV_THREAD_ID notifications, the signal specified in *notification->sigev_signo* will be sent to the calling process (SIGEV_SIGNO) or to the thread whose LWP ID is *notification->sigev_notify_thread_id* (SIGEV_THREAD_ID). The information for the queued signal will include:

Member	**Value**
si_code	SI_MESGQ
si_value	the value stored in *notification->sigev_value*
si_mqd	*mqdes*

If *notification* is NULL and the process is currently registered for notification by the specified message queue, the existing registration will be removed.

When the notification is sent to the registered process, its registration is removed. The message queue then is available for registration.

If a process has registered for notification of message arrival at a message queue and some thread is blocked in **mq_receive**() waiting to receive a message when a message arrives at the queue, the arriving message will satisfy the appropriate **mq_receive**(). The resulting behavior is as if the message queue remains empty, and no notification will be sent.

RETURN VALUES

Upon successful completion, the value 0 is returned; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

ERRORS

The **mq_notify**() system call will fail if:

[EBADF]	The *mqdes* argument is not a valid message queue descriptor.
[EBUSY]	Process is already registered for notification by the message queue.
[EINVAL]	The asynchronous notification method in *notification->sigev_notify* is invalid or not supported.

SEE ALSO

mq_open(2), mq_send(2), mq_timedsend(2), sigevent(3), siginfo(3)

STANDARDS

The **mq_notify**() system call conforms to IEEE Std 1003.1-2004 ("POSIX.1").

HISTORY

Support for POSIX message queues first appeared in FreeBSD 7.0.

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2004 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2004 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.