



Linux Ubuntu 26.04 Manual Pages on command 'mq_setattr.3'

\$ man mq_setattr.3

mq_getattr(3) Library Functions Manual mq_getattr(3)

NAME

mq_getattr, mq_setattr - get/set message queue attributes

LIBRARY

Real-time library (librt, -lrt)

SYNOPSIS

```
#include <mqueue.h>

int mq_getattr(mqd_t mqdes, struct mq_attr *attr);

int mq_setattr(mqd_t mqdes, const struct mq_attr *restrict newattr,
               struct mq_attr *restrict oldattr);
```

DESCRIPTION

mq_getattr() and mq_setattr() respectively retrieve and modify attributes of the message queue referred to by the message queue descriptor mqdes.

mq_getattr() returns an mq_attr structure in the buffer pointed by attr. This structure is defined as:

```
struct mq_attr {
    long mq_flags;    /* Flags: 0 or O_NONBLOCK */
    long mq_maxmsg;   /* Max. # of messages on queue */
    long mq_msgsize;  /* Max. message size (bytes) */
    long mq_curmsgs;  /* # of messages currently in queue */
};
```

The mq_flags field contains flags associated with the open message queue description. This field is initialized when the queue is created by mq_open(3). The only flag that can appear

The `mq_maxmsg` and `mq_msgsize` fields are set when the message queue is created by `mq_open(3)`. The `mq_maxmsg` field is an upper limit on the number of messages that may be placed on the queue using `mq_send(3)`. The `mq_msgsize` field is an upper limit on the size of messages that may be placed on the queue. Both of these fields must have a value greater than zero. Two `/proc` files that place ceilings on the values for these fields are described in `mq_overview(7)`.

The `mq_curmsgs` field returns the number of messages currently held in the queue.

`mq_setattr()` sets message queue attributes using information supplied in the `mq_attr` structure pointed to by `newattr`. The only attribute that can be modified is the setting of the `O_NONBLOCK` flag in `mq_flags`. The other fields in `newattr` are ignored. If the `oldattr` field is not `NULL`, then the buffer that it points to is used to return an `mq_attr` structure that contains the same information that is returned by `mq_getattr()`.

On success `mq_getattr()` and `mq_setattr()` return 0; on error, -1 is returned, with `errno` set to indicate the error.

EBADF The message queue descriptor specified in mqdes is invalid.

EINVAL newattr->mq_flags contained set bits other than O_NONBLOCK.

For an explanation of the terms used in this section, see `attributes(7)`.

????????????????????	?
? Interface	? Attribute ? Value ?
????????????????????	
? mq_getattr(), mq_setattr()	? Thread safety ? MT-Safe ?
????????????????????	

On Linux, `mq_getattr()` and `mq_setattr()` are library functions layered on top of the `mq_getattr(2)` system call.

POSIX.1-2008.

POSIX.1-2001.

EXAMPLES

The program below can be used to show the default `mq_maxmsg` and `mq_msgsize` values that are assigned to a message queue that is created with a call to `mq_open(3)` in which the attribute argument is `NULL`. Here is an example run of the program:

```
$ ./a.out /testq;

Maximum # of messages on queue: 10

Maximum message size:      8192
```

Since Linux 3.5, the following `/proc` files (described in `mq_overview(7)`) can be used to control the defaults:

```
$ uname -sr;

Linux 3.8.0

$ cat /proc/sys/fs/mqueue/msg_default;

10

$ cat /proc/sys/fs/mqueue/msgsize_default;

8192
```

Program source

```
#include <err.h>

#include <fcntl.h>

#include <mqueue.h>

#include <stdio.h>

#include <stdlib.h>

#include <sys/stat.h>

#include <unistd.h>

int

main(int argc, char *argv[])

{

    mqd_t mqd;

    struct mq_attr attr;

    if (argc != 2) {

        fprintf(stderr, "Usage: %s mq-name\n", argv[0]);

        exit(EXIT_FAILURE);

    }

    mqd = mq_open(argv[1], O_CREAT | O_EXCL, 0600, NULL);
```

```
if (mqd == (mqd_t) -1)
    err(EXIT_FAILURE, "mq_open");
if (mq_getattr(mqd, &attr) == -1)
    err(EXIT_FAILURE, "mq_getattr");
printf("Maximum # of messages on queue: %ld\n", attr.mq_maxmsg);
printf("Maximum message size: %ld\n", attr.mq_msgsize);
if (mq_unlink(argv[1]) == -1)
    err(EXIT_FAILURE, "mq_unlink");
exit(EXIT_SUCCESS);
}
```

SEE ALSO

mq_close(3), mq_notify(3), mq_open(3), mq_receive(3), mq_send(3), mq_unlink(3),
mq_overview(7)