## NAME

fmtmsg - print formatted error messages

## SYNOPSIS

**#include <fmtmsg.h>**

**int fmtmsg(long** *classification*, **const char ***_label_,
**int** *severity*, **const char ***_text_,
**const char ***_action_, **const char ***_tag_**);**

## DESCRIPTION

This function displays a message described by its arguments on the device(s) specified in the *classification* argument. For messages written to *stderr*, the format depends on the **MSGVERB** environment variable.

The *label* argument identifies the source of the message. The string must consist of two colon separated parts where the first part has not more than 10 and the second part not more than 14 characters.

The *text* argument describes the condition of the error.

The *action* argument describes possible steps to recover from the error. If it is printed, it is prefixed by TO FIX: .

The *tag* argument is a reference to the online documentation where more information can be found. It should contain the *label* value and a unique identification number.

### Dummy arguments

Each of the arguments can have a dummy value. The dummy classification value **MM_NULLMC** (0L) does not specify any output, so nothing is printed. The dummy severity value **NO_SEV** (0) says that no severity is supplied. The values **MM_NULLLBL**, **MM_NULLTXT**, **MM_NULLACT**, **MM_NULLTAG** are synonyms for *((char *) 0)*, the empty string, and **MM_NULLSEV** is a synonym for **NO_SEV**.

### The classification argument

The *classification* argument is the sum of values describing 4 types of information.

The first value defines the output channel.

**MM_PRINT**
Output to *stderr*.

**MM_CONSOLE**
Output to the system console.

**MM_PRINT | MM_CONSOLE**
Output to both.

The second value is the source of the error:

**MM_HARD**
A hardware error occurred.

**MM_FIRM**
A firmware error occurred.

**MM_SOFT**
A software error occurred.

The third value encodes the detector of the problem:

**MM_APPL**
It is detected by an application.

**MM_UTIL** It is detected by a utility.

**MM_OPSYS**
It is detected by the operating system.

The fourth value shows the severity of the incident:

**MM_RECOVER**
It is a recoverable error.

**MM_NRECOV**
It is a nonrecoverable error.

**The severity argument**
The *severity* argument can take one of the following values:

**MM_NOSEV**
No severity is printed.

**MM_HALT**
This value is printed as HALT.

**MM_ERROR**
This value is printed as ERROR.

**MM_WARNING**
This value is printed as WARNING.

**MM_INFO**
This value is printed as INFO.

The numeric values are between 0 and 4. Using addseverity(3) or the environment variable **SEV_LEVEL** you can add more levels and strings to print.

## RETURN VALUE

The function can return 4 values:

**MM_OK** Everything went smooth.

**MM_NOTOK**
Complete failure.

**MM_NOMSG**
Error writing to *stderr*.

**MM_NOCON**
Error writing to the console.

## ENVIRONMENT

The environment variable **MSGVERB** (message verbosity) can be used to suppress parts of the output to *stderr*. (It does not influence output to the console.) When this variable is defined, is non-NULL, and is a colon-separated list of valid keywords, then only the parts of the message corresponding to these keywords is printed. Valid keywords are label, severity, text, action and tag.

The environment variable **SEV_LEVEL** can be used to introduce new severity levels. By default, only the five severity levels described above are available. Any other numeric value would make **fmtmsg**() print nothing. If the user puts **SEV_LEVEL** with a format like

SEV_LEVEL=[description[:description[:...]]]

in the environment of the process before the first call to **fmtmsg**(), where each description is of the form

severity-keyword,level,printstring

then **fmtmsg**() will also accept the indicated values for the level (in addition to the standard levels 0-4), and use the indicated printstring when such a level occurs.

The severity-keyword part is not used by **fmtmsg**() but it has to be present. The level part is a

string representation of a number. The numeric value must be a number greater than 4. This value must be used in the severity argument of **fmtmsg**() to select this class. It is not possible to overwrite any of the predefined classes. The printstring is the string printed when a message of this class is processed by **fmtmsg**().

## VERSIONS

**fmtmsg**() is provided in glibc since version 2.1.

## ATTRIBUTES

### Multithreading (see pthreads(7))

Before glibc 2.16, the **fmtmsg**() function uses a static variable that is not protected, so it is not thread-safe.

Since glibc 2.16, the **fmtmsg**() function uses a lock to protect the static variable, so it is thread-safe.

## CONFORMING TO

The functions **fmtmsg**() and addseverity(3), and environment variables **MSGVERB** and **SEV_LEVEL** come from System V. The function **fmtmsg**() and the environment variable **MSGVERB** are described in POSIX.1-2001.

## NOTES

System V and UnixWare man pages tell us that these functions have been replaced by pfmt() and addsev() or by pfmt(), vpfmt(), lfmt(), and vlfmt(), and will be removed later.

## EXAMPLE

```
#include <stdio.h>
#include <stdlib.h>
#include <fmtmsg.h>

int
main(void)
{
long class = MM_PRINT | MM_SOFT | MM_OPSYS | MM_RECOVER;
int err;

err = fmtmsg(class, util-linux:mount, MM_ERROR,
unknown mount option, See mount(8),
util-linux:mount:017);
switch (err) {
case MM_OK:
break;
case MM_NOTOK:
printf(Nothing printedn);
break;
case MM_NOMSG:
printf(Nothing printed to stderrn);
break;
case MM_NOCON:
printf(No console outputn);
break;
default:
printf(Unknown error from fmtmsg()n);
}
exit(EXIT_SUCCESS);
}
```

The output should be:

```
util-linux:mount: ERROR: unknown mount option
```

TO FIX: See mount(8)
util-linux:mount:017

and after

MSGVERB=text:action; export MSGVERB

the output becomes:

unknown mount option
TO FIX: See mount(8)

## SEE ALSO

addseverity(3), perror(3)

## COLOPHON

This page is part of release 3.74 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 http://www.kernel.org/doc/man-pages/.