Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'closelog.3p' command

$ man closelog.3p

CLOSELOG(3P) POSIX Programmer's Manual CLOSELOG(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

closelog, openlog, setlogmask, syslog ? control system log

SYNOPSIS

```
#include <syslog.h>
void closelog(void);
void openlog(const char *ident, int logopt, int facility);
int setlogmask(int maskpri);
void syslog(int priority, const char *message, ... /* arguments */);
```

DESCRIPTION

The syslog() function shall send a message to an implementation-defined logging facility, which may log it in an implementation-defined system log, write it to the system console, forward it to a list of users, or forward it to the logging facility on another host over the network.

The logged message shall include a message header and a message body. The message header contains at least a timestamp and a tag string.

The message body is generated from the message and following arguments in the same manner as if these were arguments to printf(), except that

the additional conversion specification %m shall be recognized; it shall convert no arguments, shall cause the output of the error message string associated with the value of errno on entry to syslog(), and may be mixed with argument specifications of the "%n$" form. If a complete conversion specification with the m conversion specifier character is not just %m, the behavior is undefined. A trailing <newline> may be added if needed.

Values of the priority argument are formed by OR'ing together a sever? ity-level value and an optional facility value. If no facility value is specified, the current default facility value is used.

Possible values of severity level include:

LOG_EMERG A panic condition.

LOG_ALERT A condition that should be corrected immediately, such as a corrupted system database.

LOG_CRIT Critical conditions, such as hard device errors.

LOG_ERR Errors.

LOG_WARNING

Warning messages.

LOG_NOTICE Conditions that are not error conditions, but that may re? quire special handling.

LOG_INFO Informational messages.

LOG_DEBUG Messages that contain information normally of use only when debugging a program.

The facility indicates the application or system component generating the message. Possible facility values include:

LOG_USER Messages generated by arbitrary processes. This is the de? fault facility identifier if none is specified.

LOG_LOCAL0 Reserved for local use.

LOG_LOCAL1 Reserved for local use.

LOG_LOCAL2 Reserved for local use.

LOG_LOCAL3 Reserved for local use.

LOG_LOCAL4 Reserved for local use.

LOG_LOCAL5 Reserved for local use.

LOG_LOCAL6 Reserved for local use.

LOG_LOCAL7 Reserved for local use.

The openlog() function shall set process attributes that affect subse? quent calls to syslog(). The ident argument is a string that is prepended to every message. The logopt argument indicates logging op? tions. Values for logopt are constructed by a bitwise-inclusive OR of zero or more of the following:

LOG_PID Log the process ID with each message. This is useful for identifying specific processes.

LOG_CONS Write messages to the system console if they cannot be sent to the logging facility. The syslog() function ensures that the process does not acquire the console as a controlling terminal in the process of writing the message.

LOG_NDELAY Open the connection to the logging facility immediately. Normally the open is delayed until the first message is logged. This is useful for programs that need to manage the order in which file descriptors are allocated.

LOG_ODELAY Delay open until syslog() is called.

LOG_NOWAIT Do not wait for child processes that may have been created during the course of logging the message. This option should be used by processes that enable notification of child termination using SIGCHLD, since syslog() may other? wise block waiting for a child whose exit status has al? ready been collected.

The facility argument encodes a default facility to be assigned to all messages that do not have an explicit facility already encoded. The initial default facility is LOG_USER.

The openlog() and syslog() functions may allocate a file descriptor. It is not necessary to call openlog() prior to calling syslog().

The closelog() function shall close any open file descriptors allocated by previous calls to openlog() or syslog().

The setlogmask() function shall set the log priority mask for the cur? rent process to maskpri and return the previous mask. If the maskpri

argument is 0, the current log mask is not modified. Calls by the cur?
rent process to syslog() with a priority not set in maskpri shall be
rejected. The default log mask allows all priorities to be logged. A
call to openlog() is not required prior to calling setlogmask().
Symbolic constants for use as values of the logopt, facility, priority,
and maskpri arguments are defined in the <syslog.h> header.

RETURN VALUE

The setlogmask() function shall return the previous log priority mask.
The closelog(), openlog(), and syslog() functions shall not return a
value.

ERRORS

No errors are defined.
The following sections are informative.

EXAMPLES

Using openlog()

The following example causes subsequent calls to syslog() to log the
process ID with each message, and to write messages to the system con?
sole if they cannot be sent to the logging facility.

```
#include <syslog.h>
char *ident = "Process demo";
int logopt = LOG_PID | LOG_CONS;
int facility = LOG_USER;
...
openlog(ident, logopt, facility);
```

Using setlogmask()

The following example causes subsequent calls to syslog() to accept er?
ror messages, and to reject all other messages.

```
#include <syslog.h>
int result;
int mask = LOG_MASK (LOG_ERR);
...
result = setlogmask(mask);
```

Using syslog

The following example sends the message "Thisisamessage" to the default logging facility, marking the message as an error message generated by random processes.

```
#include <syslog.h>
char *message = "This is a message";
int priority = LOG_ERR | LOG_USER;
...
syslog(priority, message);
```

APPLICATION USAGE

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

fprintf()

The Base Definitions volume of POSIX.1?2017, <syslog.h>

COPYRIGHT