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 'fclose.3p' command

$ man fclose.3p

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

fclose ? close a stream

SYNOPSIS

```
#include <stdio.h>
int fclose(FILE *stream);
```

DESCRIPTION

The functionality described on this reference page is aligned with the ISO C standard. Any conflict between the requirements described here and the ISO C standard is unintentional. This volume of POSIX.1?2017 defers to the ISO C standard.

The fclose() function shall cause the stream pointed to by stream to be flushed and the associated file to be closed. Any unwritten buffered data for the stream shall be written to the file; any unread buffered data shall be discarded. Whether or not the call succeeds, the stream shall be disassociated from the file and any buffer set by the setbuf() or setvbuf() function shall be disassociated from the stream. If the associated buffer was automatically allocated, it shall be deallocated.

If the file is not already at EOF, and the file is one capable of seek?
ing, the file offset of the underlying open file description shall be
set to the file position of the stream if the stream is the active han?
dle to the underlying file description.

The fclose() function shall mark for update the last data modification
and last file status change timestamps of the underlying file, if the
stream was writable, and if buffered data remains that has not yet been
written to the file. The fclose() function shall perform the equivalent
of a close() on the file descriptor that is associated with the stream
pointed to by stream.

After the call to fclose(), any use of stream results in undefined be?
havior.

RETURN VALUE

Upon successful completion, fclose() shall return 0; otherwise, it
shall return EOF and set errno to indicate the error.

ERRORS

The fclose() function shall fail if:

EAGAIN The O_NONBLOCK flag is set for the file descriptor underlying
stream and the thread would be delayed in the write operation.

EBADF The file descriptor underlying stream is not valid.

EFBIG An attempt was made to write a file that exceeds the maximum
file size.

EFBIG An attempt was made to write a file that exceeds the file size
limit of the process.

EFBIG The file is a regular file and an attempt was made to write at
or beyond the offset maximum associated with the corresponding
stream.

EINTR The fclose() function was interrupted by a signal.

EIO The process is a member of a background process group attempting
to write to its controlling terminal, TOSTOP is set, the calling
thread is not blocking SIGTTOU, the process is not ignoring
SIGTTOU, and the process group of the process is orphaned. This
error may also be returned under implementation-defined condi?

tions.

ENOMEM The underlying stream was created by open_memstream() or open_wmemstream() and insufficient memory is available.

ENOSPC There was no free space remaining on the device containing the file or in the buffer used by the fmemopen() function.

EPIPE An attempt is made to write to a pipe or FIFO that is not open for reading by any process. A SIGPIPE signal shall also be sent to the thread.

The fclose() function may fail if:

ENXIO A request was made of a nonexistent device, or the request was outside the capabilities of the device.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

Since after the call to fclose() any use of stream results in undefined behavior, fclose() should not be used on stdin, stdout, or stderr except immediately before process termination (see the Base Definitions volume of POSIX.1?2017, Section 3.303, Process Termination), so as to avoid triggering undefined behavior in other standard interfaces that rely on these streams. If there are any atexit() handlers registered by the application, such a call to fclose() should not occur until the last handler is finishing. Once fclose() has been used to close stdin, stdout, or stderr, there is no standard way to reopen any of these streams.

Use of freopen() to change stdin, stdout, or stderr instead of closing them avoids the danger of a file unexpectedly being opened as one of the special file descriptors STDIN_FILENO, STDOUT_FILENO, or STDERR_FILENO at a later time in the application.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.5, Standard I/O Streams, atexit(), close(), fmemopen(), fopen(), freopen(), getrlimit(), open_memstream(), ulimit()

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

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Por? table Operating System Interface (POSIX), The Open Group Base Specifi? cations Issue 7, 2018 Edition, Copyright (C) 2018 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 .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.ker? nel.org/doc/man-pages/reporting_bugs.html .