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 'fseek.3p' command

$ man fseek.3p

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

fseek, fseeko ? reposition a file-position indicator in a stream

SYNOPSIS

```
#include <stdio.h>
int fseek(FILE *stream, long offset, int whence);
int fseeko(FILE *stream, off_t offset, int whence);
```

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 fseek() function shall set the file-position indicator for the stream pointed to by stream. If a read or write error occurs, the er? ror indicator for the stream shall be set and fseek() fails.

The new position, measured in bytes from the beginning of the file, shall be obtained by adding offset to the position specified by whence.

The specified point is the beginning of the file for SEEK_SET, the cur?

rent value of the file-position indicator for SEEK_CUR, or end-of-file for SEEK_END.

If the stream is to be used with wide-character input/output functions, the application shall ensure that offset is either 0 or a value re? turned by an earlier call to ftell() on the same stream and whence is SEEK_SET.

A successful call to fseek() shall clear the end-of-file indicator for the stream and undo any effects of ungetc() and ungetwc() on the same stream. After an fseek() call, the next operation on an update stream may be either input or output.

If the most recent operation, other than ftell(), on a given stream is fflush(), the file offset in the underlying open file description shall be adjusted to reflect the location specified by fseek().

The fseek() function shall allow the file-position indicator to be set beyond the end of existing data in the file. If data is later written at this point, subsequent reads of data in the gap shall return bytes with the value 0 until data is actually written into the gap.

The behavior of fseek() on devices which are incapable of seeking is implementation-defined. The value of the file offset associated with such a device is undefined.

If the stream is writable and buffered data had not been written to the underlying file, fseek() shall cause the unwritten data to be written to the file and shall mark the last data modification and last file status change timestamps of the file for update.

In a locale with state-dependent encoding, whether fseek() restores the stream's shift state is implementation-defined.

The fseeko() function shall be equivalent to the fseek() function ex? cept that the offset argument is of type off_t.

RETURN VALUE

The fseek() and fseeko() functions shall return 0 if they succeed.

Otherwise, they shall return -1 and set errno to indicate the error.

ERRORS

The fseek() and fseeko() functions shall fail if, either the stream is

unbuffered or the stream's buffer needed to be flushed, and the call to fseek() or fseeko() causes an underlying lseek() or write() to be in? voked, and:

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

EBADF The file descriptor underlying the stream file is not open for writing or the stream's buffer needed to be flushed and the file is not open.

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 write operation was terminated due to the receipt of a sig? nal, and no data was transferred.

EINVAL The whence argument is invalid. The resulting file-position in? dicator would be set to a negative value.

EIO A physical I/O error has occurred, or the process is a member of a background process group attempting to perform a 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 conditions.

ENOSPC There was no free space remaining on the device containing the file.

EOVERFLOW

For fseek(), the resulting file offset would be a value which cannot be represented correctly in an object of type long.

EOVERFLOW

For fseeko(), the resulting file offset would be a value which cannot be represented correctly in an object of type off_t.

EPIPE An attempt was 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.

ESPIPE The file descriptor underlying stream is associated with a pipe, FIFO, or socket.

The fseek() and fseeko() functions 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

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.5, Standard I/O Streams, fopen(), fsetpos(), ftell(), getr? limit(), lseek(), rewind(), ulimit(), ungetc(), write()

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

COPYRIGHT

files to man page format. To report such errors, see https://www.ker?

nel.org/doc/man-pages/reporting_bugs.html .