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 'fopen.3p' command

$ man fopen.3p

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

fopen ? open a stream

SYNOPSIS

#include <stdio.h>

FILE *fopen(const char *restrict pathname, const char *restrict mode);

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 fopen() function shall open the file whose pathname is the string pointed to by pathname, and associates a stream with it.

The mode argument points to a string. If the string is one of the fol? lowing, the file shall be opened in the indicated mode. Otherwise, the behavior is undefined.

r or rb Open file for reading.

w or wb Truncate to zero length or create file for writing.

a or ab Append; open or create file for writing at end-of-file.

r+ or rb+ or r+b

Open file for update (reading and writing).

w+ or wb+ or w+b

Truncate to zero length or create file for update.

a+ or ab+ or a+b

Append; open or create file for update, writing at end-of-file.

The character 'b' shall have no effect, but is allowed for ISO C stan? dard conformance. Opening a file with read mode (r as the first char? acter in the mode argument) shall fail if the file does not exist or cannot be read.

Opening a file with append mode (a as the first character in the mode argument) shall cause all subsequent writes to the file to be forced to the then current end-of-file, regardless of intervening calls to fseek().

When a file is opened with update mode ('+' as the second or third character in the mode argument), both input and output may be performed on the associated stream. However, the application shall ensure that output is not directly followed by input without an intervening call to fflush() or to a file positioning function (fseek(), fsetpos(), or rewind()), and input is not directly followed by output without an in? tervening call to a file positioning function, unless the input opera? tion encounters end-of-file.

When opened, a stream is fully buffered if and only if it can be deter? mined not to refer to an interactive device. The error and end-of-file indicators for the stream shall be cleared.

If mode is w, wb, a, ab, w+, wb+, w+b, a+, ab+, or a+b, and the file did not previously exist, upon successful completion, fopen() shall mark for update the last data access, last data modification, and last file status change timestamps of the file and the last file status change and last data modification timestamps of the parent directory.

If mode is w, wb, a, ab, w+, wb+, w+b, a+, ab+, or a+b, and the file

did not previously exist, the fopen() function shall create a file as if it called the creat() function with a value appropriate for the path argument interpreted from pathname and a value of S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP | S_IROTH | S_IWOTH for the mode argument.

If mode is w, wb, w+, wb+, or w+b, and the file did previously exist, upon successful completion, fopen() shall mark for update the last data modification and last file status change timestamps of the file.

After a successful call to the fopen() function, the orientation of the stream shall be cleared, the encoding rule shall be cleared, and the associated mbstate_t object shall be set to describe an initial conver? sion state.

The file descriptor associated with the opened stream shall be allo? cated and opened as if by a call to open() with the following flags:

fopen() Mode	open() Flags
r or rb	O_RDONLY
w or wb	O_WRONLY|O_CREAT|O_TRUNC
a or ab	O_WRONLY|O_CREAT|O_APPEND
r+ or rb+ or r+b	O_RDWR
w+ or wb+ or w+b	O_RDWR|O_CREAT|O_TRUNC
a+ or ab+ or a+b	O_RDWR|O_CREAT|O_APPEND

RETURN VALUE

Upon successful completion, fopen() shall return a pointer to the ob? ject controlling the stream. Otherwise, a null pointer shall be re? turned, and errno shall be set to indicate the error.

ERRORS

The fopen() function shall fail if:

EACCES Search permission is denied on a component of the path prefix, or the file exists and the permissions specified by mode are de? nied, or the file does not exist and write permission is denied for the parent directory of the file to be created.

EINTR A signal was caught during fopen().

EISDIR The named file is a directory and mode requires write access.

ELOOP A loop exists in symbolic links encountered during resolution of the path argument.

EMFILE All file descriptors available to the process are currently open.

EMFILE {STREAM_MAX} streams are currently open in the calling process.

ENAMETOOLONG

The length of a pathname exceeds {PATH_MAX}, or pathname resolu? tion of a symbolic link produced an intermediate result with a length that exceeds {PATH_MAX}.

ENFILE The maximum allowable number of files is currently open in the system.

ENOENT The mode string begins with 'r' and a component of pathname does not name an existing file, or mode begins with 'w' or 'a' and a component of the path prefix of pathname does not name an exist? ing file, or pathname is an empty string.

ENOENT or ENOTDIR

The pathname argument contains at least one non-<slash> charac? ter and ends with one or more trailing <slash> characters. If pathname without the trailing <slash> characters would name an existing file, an [ENOENT] error shall not occur.

ENOSPC The directory or file system that would contain the new file cannot be expanded, the file does not exist, and the file was to be created.

ENOTDIR

A component of the path prefix names an existing file that is neither a directory nor a symbolic link to a directory, or the pathname argument contains at least one non-<slash> character and ends with one or more trailing <slash> characters and the last pathname component names an existing file that is neither a directory nor a symbolic link to a directory.

ENXIO The named file is a character special or block special file, and

the device associated with this special file does not exist.

EOVERFLOW

The named file is a regular file and the size of the file cannot be represented correctly in an object of type off_t.

EROFS The named file resides on a read-only file system and mode requires write access.

The fopen() function may fail if:

EINVAL The value of the mode argument is not valid.

ELOOP More than {SYMLOOP_MAX} symbolic links were encountered during resolution of the path argument.

EMFILE {FOPEN_MAX} streams are currently open in the calling process.

ENAMETOOLONG

The length of a component of a pathname is longer than {NAME_MAX}.

ENOMEM Insufficient storage space is available.

ETXTBSY

The file is a pure procedure (shared text) file that is being executed and mode requires write access.

The following sections are informative.

EXAMPLES

Opening a File

The following example tries to open the file named file for reading. The fopen() function returns a file pointer that is used in subsequent fgets() and fclose() calls. If the program cannot open the file, it just ignores it.

```
#include <stdio.h>
...
FILE *fp;
...
void rgrep(const char *file)
{
...
  if ((fp = fopen(file, "r")) == NULL)
```

```
        return;
    ...
    }
```

APPLICATION USAGE

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.5, Standard I/O Streams, creat(), fclose(), fdopen(), fmemo?
pen(), freopen(), open_memstream()

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