NAME

fopen, **fdopen**, **freopen**, **fmemopen** - stream open functions

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <stdio.h>

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

*FILE **
fdopen(*int fildes, const char *mode*);

*FILE **
freopen(*const char *path, const char *mode, FILE *stream*);

*FILE **
fmemopen(*void * restrict buf, size_t size, const char * restrict mode*);

DESCRIPTION

The **fopen**() function opens the file whose name is the string pointed to by *path* and associates a stream with it.

The argument *mode* points to a string beginning with one of the following letters:

"r" Open for reading. The stream is positioned at the beginning of the file. Fail if the file does not exist.

"w" Open for writing. The stream is positioned at the beginning of the file. Truncate the file to zero length if it exists or create the file if it does not exist.

"a" Open for writing. The stream is positioned at the end of the file. Subsequent writes to the file will always end up at the then current end of file, irrespective of any intervening fseek(3) or similar. Create the file if it does not exist.

An optional "+" following "r", "w", or "a" opens the file for both reading and writing. An optional "x" following "w" or "w+" causes the **fopen**() call to fail if the file already exists. An optional "e" following the above causes the **fopen**() call to set the FD_CLOEXEC flag on the underlying file descriptor.

The *mode* string can also include the letter "b" after either the "+" or the first letter. This is strictly for compatibility with ISO/IEC 9899:1990 ("ISO C90") and has effect only for **fmemopen**(); otherwise "b" is ignored.

Any created files will have mode "S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP | S_IROTH | S_IWOTH" (0666), as modified by the process' umask value (see umask(2)).

Reads and writes may be intermixed on read/write streams in any order, and do not require an intermediate seek as in previous versions of *stdio*. This is not portable to other systems, however; ISO/IEC 9899:1990 ("ISO C90") and IEEE Std 1003.1 ("POSIX.1") both require that a file positioning function intervene between output and input, unless an input operation encounters end-of-file.

The **fdopen**() function associates a stream with the existing file descriptor, *fildes*. The mode of the stream must be compatible with the mode of the file descriptor. The "x" mode option is ignored. If the "e" mode option is present, the FD_CLOEXEC flag is set, otherwise it remains unchanged. When the stream is closed via fclose(3), *fildes* is closed also.

The **freopen**() function opens the file whose name is the string pointed to by *path* and associates the stream pointed to by *stream* with it. The original stream (if it exists) is closed. The *mode* argument is used just as in the **fopen**() function.

If the *path* argument is NULL, **freopen**() attempts to re-open the file associated with *stream* with a new mode. The new mode must be compatible with the mode that the stream was originally opened with: Streams open for reading can only be re-opened for reading, streams open for writing can only be re-opened for writing, and streams open for reading and writing can be re-opened in any mode. The "x" mode option is not meaningful in this context.

The primary use of the **freopen**() function is to change the file associated with a standard text stream (stderr, stdin, or stdout).

The **fmemopen**() function associates the buffer given by the *buf* and *size* arguments with a stream. The *buf* argument is either a null pointer or point to a buffer that is at least *size* bytes long. If a null pointer is specified as the *buf* argument, **fmemopen**() allocates *size* bytes of memory. This buffer is automatically freed when the stream is closed. Buffers can be opened in text-mode (default) or binary-mode (if "b" is present in the second or third position of the *mode* argument). Buffers opened in text-mode make sure that writes are terminated with a NULL byte, if the last write hasn't filled up the whole buffer. Buffers opened in binary-mode never append a NULL byte.

RETURN VALUES

Upon successful completion **fopen**(), **fdopen**(), **freopen**() and **fmemopen**() return a FILE pointer.

Otherwise, NULL is returned and the global variable *errno* is set to indicate the error.

ERRORS

[EINVAL] The *mode* argument to **fopen**(), **fdopen**(), **freopen**(), or **fmemopen**() was invalid.

The **fopen**(), **fdopen**(), **freopen**() and **fmemopen**() functions may also fail and set *errno* for any of the errors specified for the routine malloc(3).

The **fopen**() function may also fail and set *errno* for any of the errors specified for the routine open(2).

The **fdopen**() function may also fail and set *errno* for any of the errors specified for the routine fcntl(2).

The **freopen**() function may also fail and set *errno* for any of the errors specified for the routines open(2), fclose(3) and fflush(3).

The **fmemopen**() function may also fail and set *errno* if the *size* argument is 0.

SEE ALSO

open(2), fclose(3), fileno(3), fseek(3), funopen(3)

STANDARDS

The **fopen**() and **freopen**() functions conform to ISO/IEC 9899:1990 ("ISO C90"), with the exception of the "x" mode option which conforms to ISO/IEC 9899:2011 ("ISO C11"). The **fdopen**() function conforms to IEEE Std 1003.1-1988 ("POSIX.1"). The "e" mode option does not conform to any standard but is also supported by glibc. The **fmemopen**() function conforms to IEEE Std 1003.1-2008 ("POSIX.1"). The "b" mode does not conform to any standard but is also supported by glibc.

HISTORY

An **fopen**() function appeared in Version 1 AT&T UNIX.