NAME

lseek - reposition read/write file offset

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <unistd.h>

off_t
lseek(*int fildes, off_t offset, int whence*);

DESCRIPTION

The **lseek**() system call repositions the offset of the file descriptor *fildes* to the argument *offset* according to the directive *whence*. The argument *fildes* must be an open file descriptor. The **lseek**() system call repositions the file position pointer associated with the file descriptor *fildes* as follows:

If *whence* is SEEK_SET, the offset is set to *offset* bytes.

If *whence* is SEEK_CUR, the offset is set to its current location plus *offset* bytes.

If *whence* is SEEK_END, the offset is set to the size of the file plus *offset* bytes.

If *whence* is SEEK_HOLE, the offset is set to the start of the next hole greater than or equal to the supplied *offset*. The definition of a hole is provided below.

If *whence* is SEEK_DATA, the offset is set to the start of the next non-hole file region greater than or equal to the supplied *offset*.

The **lseek**() system call allows the file offset to be set beyond the end of the existing end-of-file of the file. If data is later written at this point, subsequent reads of the data in the gap return bytes of zeros (until data is actually written into the gap). However, the **lseek**() system call does not, by itself, extend the size of a file.

A "hole" is defined as a contiguous range of bytes in a file, all having the value of zero, but not all zeros in a file are guaranteed to be represented as holes returned with SEEK_HOLE. File systems are allowed to expose ranges of zeros with SEEK_HOLE, but not required to. Applications can use SEEK_HOLE to optimise their behavior for ranges of zeros, but must not depend on it to find all such ranges in a file. Each file is presented as having a zero-size virtual hole at the very end of the file. The existence of a hole at the end of every data region allows for easy programming and also provides compatibility to the

original implementation in Solaris. It also causes the current file size (i.e., end-of-file offset) to be returned to indicate that there are no more holes past the supplied *offset*. Applications should use **fpathconf**(*_PC_MIN_HOLE_SIZE*) or **pathconf**(*_PC_MIN_HOLE_SIZE*) to determine if a file system supports SEEK_HOLE. See pathconf(2).

For file systems that do not supply information about holes, the file will be represented as one entire data region.

RETURN VALUES

Upon successful completion, **lseek**() returns the resulting offset location as measured in bytes from the beginning of the file. Otherwise, a value of -1 is returned and *errno* is set to indicate the error.

ERRORS

The **lseek**() system call will fail and the file position pointer will remain unchanged if:

[EBADF] The *fildes* argument is not an open file descriptor.

[EINVAL] The *whence* argument is not a proper value or the resulting file offset would be negative for a non-character special file.

[ENXIO] For SEEK_DATA, there are no more data regions past the supplied offset. Due to existence of the hole at the end of the file, for SEEK_HOLE this error is only returned when the *offset* already points to the end-of-file position.

[EOVERFLOW] The resulting file offset would be a value which cannot be represented correctly in an object of type *off_t*.

[ESPIPE] The *fildes* argument is associated with a pipe, socket, or FIFO.

SEE ALSO

dup(2), open(2), pathconf(2)

STANDARDS

The **lseek**() system call is expected to conform to IEEE Std 1003.1-2008 ("POSIX.1").

The SEEK_HOLE and SEEK_DATA directives, along with the ENXIO error, are extensions to that specification.

HISTORY

The **lseek**() function appeared in Version 7 AT&T UNIX.

BUGS

If the **lseek**() system call is operating on a device which is incapable of seeking, it will request the seek operation and return successfully, even though no seek was performed. Because the *offset* argument will be stored unconditionally in the file descriptor of that device, there is no way to confirm if the seek operation succeeded or not (e.g. using the **ftell**() function). Device types which are known to be incapable of seeking include tape drives.

The **lseek**() system call will not detect whether media are present in changeable media devices such as DVD or Blu-ray devices. A requested seek operation will therefore return sucessfully when no medium is present.

This document’s use of *whence* is incorrect English, but is maintained for historical reasons.