



## ***Linux Ubuntu 26.04 Manual Pages on command 'lseek.2'***

***\$ man lseek.2***

lseek(2)                      System Calls Manual                      lseek(2)

### NAME

lseek - reposition read/write file offset

### LIBRARY

Standard C library (libc, -lc)

### SYNOPSIS

```
#include <unistd.h>
```

```
off_t lseek(int fd, off_t offset, int whence);
```

### DESCRIPTION

lseek() repositions the file offset of the open file description associated with the file descriptor fd to the argument offset according to the directive whence as follows:

#### SEEK\_SET

The file offset is set to offset bytes.

#### SEEK\_CUR

The file offset is set to its current location plus offset bytes.

#### SEEK\_END

The file offset is set to the size of the file plus offset bytes.

lseek() allows the file offset to be set beyond the end of the file (but this does not change the size of the file). If data is later written at this point, subsequent reads of the data in the gap (a "hole") return null bytes ('\0') until data is actually written into the gap.

### Seeking file data and holes

Since Linux 3.1, Linux supports the following additional values for whence:

## SEEK\_DATA

Adjust the file offset to the next location in the file greater than or equal to off?

set containing data. If offset points to data, then the file offset is set to off?

set.

## SEEK\_HOLE

Adjust the file offset to the next hole in the file greater than or equal to offset.

If offset points into the middle of a hole, then the file offset is set to offset.

If there is no hole past offset, then the file offset is adjusted to the end of the file (i.e., there is an implicit hole at the end of any file).

In both of the above cases, lseek() fails if offset points past the end of the file.

These operations allow applications to map holes in a sparsely allocated file. This can be useful for applications such as file backup tools, which can save space when creating backups and preserve holes, if they have a mechanism for discovering holes.

For the purposes of these operations, a hole is a sequence of zeros that (normally) has not been allocated in the underlying file storage. However, a filesystem is not obliged to report holes, so these operations are not a guaranteed mechanism for mapping the storage space actually allocated to a file. (Furthermore, a sequence of zeros that actually has been written to the underlying storage may not be reported as a hole.) In the simplest implementation, a filesystem can support the operations by making SEEK\_HOLE always return the offset of the end of the file, and making SEEK\_DATA always return offset (i.e., even if the location referred to by offset is a hole, it can be considered to consist of data that is a sequence of zeros).

The \_GNU\_SOURCE feature test macro must be defined in order to obtain the definitions of SEEK\_DATA and SEEK\_HOLE from <unistd.h>.

The SEEK\_HOLE and SEEK\_DATA operations are supported for the following filesystems:

- ? Btrfs (since Linux 3.1)
- ? OCFS (since Linux 3.2)
- ? XFS (since Linux 3.5)
- ? ext4 (since Linux 3.8)
- ? tmpfs(5) (since Linux 3.8)
- ? NFS (since Linux 3.18)
- ? FUSE (since Linux 4.5)
- ? GFS2 (since Linux 4.15)

## RETURN VALUE

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

## ERRORS

`EBADF` `fd` is not an open file descriptor.

`EINVAL` `whence` is not valid. Or: the resulting file offset would be negative, or beyond the end of a seekable device.

`ENXIO` `whence` is `SEEK_DATA` or `SEEK_HOLE`, and offset is beyond the end of the file, or `whence` is `SEEK_DATA` and offset is within a hole at the end of the file.

`EOVERFLOW`

The resulting file offset cannot be represented in an `off_t`.

`ESPIPE` `fd` is associated with a pipe, socket, or FIFO.

## VERSIONS

On Linux, using `lseek()` on a terminal device fails with the error `ESPIPE`.

## STANDARDS

POSIX.1-2024.

## HISTORY

4.3BSD, SVr4, POSIX.1-1988.

`SEEK_DATA`

`SEEK_HOLE`

Solaris 10, FreeBSD 7.0, DragonFly BSD 2.3.1, POSIX.1-2024.

## NOTES

See `open(2)` for a discussion of the relationship between file descriptors, open file descriptions, and files.

If the `O_APPEND` file status flag is set on the open file description, then a `write(2)` always moves the file offset to the end of the file, regardless of the use of `lseek()`.

Some devices are incapable of seeking and POSIX does not specify which devices must support `lseek()`.

## SEE ALSO

`dup(2)`, `fallocate(2)`, `fork(2)`, `open(2)`, `fseek(3)`, `lseek64(3)`, `posix_fallocate(3)`