NAME

readlink, **readlinkat** - read value of a symbolic link

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <unistd.h>

ssize_t
readlink(*const char *restrict path, char *restrict buf, size_t bufsiz*);

ssize_t
readlinkat(*int fd, const char *restrict path, char *restrict buf, size_t bufsize*);

DESCRIPTION

The **readlink**() system call places the contents of the symbolic link *path* in the buffer *buf*, which has size *bufsiz*. The **readlink**() system call does not append a NUL character to *buf*.

The **readlinkat**() system call is equivalent to **readlink**() except in the case where *path* specifies a relative path. In this case the symbolic link whose content is read relative to the directory associated with the file descriptor *fd* instead of the current working directory. If **readlinkat**() is passed the special value AT_FDCWD in the *fd* parameter, the current working directory is used and the behavior is identical to a call to **readlink**().

RETURN VALUES

The call returns the count of characters placed in the buffer if it succeeds, or a -1 if an error occurs, placing the error code in the global variable *errno*.

ERRORS

The **readlink**() system call will fail if:

[ENOTDIR] A component of the path prefix is not a directory.

[ENAMETOOLONG]
A component of a pathname exceeded 255 characters, or an entire path name exceeded 1023 characters.

[ENOENT] The named file does not exist.

[EACCES] Search permission is denied for a component of the path prefix.

[ELOOP] Too many symbolic links were encountered in translating the pathname.

[EINVAL] The named file is not a symbolic link.

[EIO] An I/O error occurred while reading from the file system.

[EINTEGRITY] Corrupted data was detected while reading from the file system.

[EFAULT] The *buf* argument extends outside the process's allocated address space.

In addition to the errors returned by the **readlink**(), the **readlinkat**() may fail if:

[EBADF] The *path* argument does not specify an absolute path and the *fd* argument is neither AT_FDCWD nor a valid file descriptor open for searching.

[ENOTDIR] The *path* argument is not an absolute path and *fd* is neither AT_FDCWD nor a file descriptor associated with a directory.

SEE ALSO

lstat(2), stat(2), symlink(2), symlink(7)

STANDARDS

The **readlinkat**() system call follows The Open Group Extended API Set 2 specification.

HISTORY

The **readlink**() system call appeared in 4.2BSD. The **readlinkat**() system call appeared in FreeBSD 8.0.