NAME

openat − open a file relative to a directory file descriptor

SYNOPSIS

#define _ATFILE_SOURCE
#include <fcntl.h>

int openat(int *dirfd***, const char ***pathname***, int** *flags***);**
int openat(int *dirfd***, const char ***pathname***, int** *flags***, mode_t** *mode***);**

DESCRIPTION

The **openat**() system call operates in exactly the same way as **open**(2), except for the differences described in this manual page.

If the pathname given in *pathname* is relative, then it is interpreted relative to the directory referred to by the file descriptor *dirfd* (rather than relative to the current working directory of the calling process, as is done by **open**(2) for a relative pathname).

If *pathname* is relative and *dirfd* is the special value **AT_FDCWD**, then *pathname* is interpreted relative to the current working directory of the calling process (like **open**(2)).

If *pathname* is absolute, then *dirfd* is ignored.

RETURN VALUE

On success, **openat**() returns a new file descriptor. On error, −1 is returned and *errno* is set to indicate the error.

ERRORS

The same errors that occur for **open**(2) can also occur for **openat**(). The following additional errors can occur for **openat**():

EBADF
dirfd is not a valid file descriptor.

ENOTDIR
pathname is relative and *dirfd* is a file descriptor referring to a file other than a directory.

VERSIONS

openat() was added to Linux in kernel 2.6.16.

CONFORMING TO

POSIX.1-2008. A similar system call exists on Solaris.

NOTES

openat() and other similar system calls suffixed "at" are supported for two reasons.

First, **openat**() allows an application to avoid race conditions that could occur when using **open**(2) to open files in directories other than the current working directory. These race conditions result from the fact that some component of the directory prefix given to **open**(2) could be changed in parallel with the call to **open**(2). Such races can be avoided by opening a file descriptor for the target directory, and then specifying that file descriptor as the *dirfd* argument of **openat**().

Second, **openat**() allows the implementation of a per-thread "current working directory", via file descriptor(s) maintained by the application. (This functionality can also be obtained by tricks based on the use of */proc/self/fd/* dirfd, but less efficiently.)

SEE ALSO

faccessat(2), **fchmodat**(2), **fchownat**(2), **fstatat**(2), **futimesat**(2), **linkat**(2), **mkdirat**(2), **mknodat**(2), **open**(2), **readlinkat**(2), **renameat**(2), **symlinkat**(2), **unlinkat**(2), **utimensat**(2), **mkfifoat**(3), **path_resolution**(7)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.