NAME

mkdir − create a directory

SYNOPSIS

```
#include <sys/stat.h>
#include <sys/types.h>

int mkdir(const char *pathname, mode_t mode);
```

DESCRIPTION

mkdir() attempts to create a directory named *pathname*.

The argument *mode* specifies the permissions to use. It is modified by the process's *umask* in the usual way: the permissions of the created directory are (*mode* & ˜*umask* & 0777). Other mode bits of the created directory depend on the operating system. For Linux, see below.

The newly created directory will be owned by the effective user ID of the process. If the directory containing the file has the set-group-ID bit set, or if the file system is mounted with BSD group semantics (*mount -o bsdgroups* or, synonymously *mount -o grpid*), the new directory will inherit the group ownership from its parent; otherwise it will be owned by the effective group ID of the process.

If the parent directory has the set-group-ID bit set then so will the newly created directory.

RETURN VALUE

mkdir() returns zero on success, or −1 if an error occurred (in which case, *errno* is set appropriately).

ERRORS

EACCES
The parent directory does not allow write permission to the process, or one of the directories in *pathname* did not allow search permission. (See also **path_resolution**(7).)

EEXIST
pathname already exists (not necessarily as a directory). This includes the case where *pathname* is a symbolic link, dangling or not.

EFAULT
pathname points outside your accessible address space.

ELOOP
Too many symbolic links were encountered in resolving *pathname*.

ENAMETOOLONG
pathname was too long.

ENOENT
A directory component in *pathname* does not exist or is a dangling symbolic link.

ENOMEM
Insufficient kernel memory was available.

ENOSPC
The device containing *pathname* has no room for the new directory.

ENOSPC
The new directory cannot be created because the user's disk quota is exhausted.

ENOTDIR
A component used as a directory in *pathname* is not, in fact, a directory.

EPERM
The file system containing *pathname* does not support the creation of directories.

EROFS
pathname refers to a file on a read-only file system.

CONFORMING TO

SVr4, BSD, POSIX.1-2001.

NOTES

Under Linux apart from the permission bits, only the **S_ISVTX** mode bit is honored. That is, under Linux the created directory actually gets mode (*mode* & ˜*umask* & 01777). See also **stat**(2).

There are many infelicities in the protocol underlying NFS. Some of these affect **mkdir**().

SEE ALSO

mkdir(1), **chmod**(2), **chown**(2), **mkdirat**(2), **mknod**(2), **mount**(2), **rmdir**(2), **stat**(2), **umask**(2), **unlink**(2), **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/.