

mkdir() — Make a Directory

Standards

Standards / Extensions	C or C++	Dependencies
POSIX.1 XPG4 XPG4.2 Single UNIX Specification, Version 3	both	

Format

```
#define _POSIX_SOURCE
#include <sys/stat.h>

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

General Description

Creates a new, empty directory, *pathname*. The file permission bits in *mode* are modified by the file creation mask of the process, and then used to set the file permission bits of the directory being created.

The *mode* argument is created with one of the following symbols defined in the `sys/stat.h` header file.

Any mode flags that are not defined will be turned off, and the function will be allowed to proceed.

S_IRGRP
Read permission for the file's group.

S_IROTH
Read permission for users other than the file owner.

S_IRUSR
Read permission for the file owner.

S_IRWXG
Read, write, and search or execute permission for the file's group. **S_IRWXG** is the bitwise inclusive-OR of **S_IRGRP**, **S_IWGRP**, and **S_IXGRP**.

S_IRWXO
Read, write, and search or execute permission for users other than the file owner. **S_IRWXO** is the bitwise inclusive-OR of **S_IROTH**, **S_IWOTH**, and **S_IXOTH**.

S_IRWXU
Read, write, and search, or execute, for the file owner; **S_IRWXG** is the bitwise inclusive-OR of **S_IRUSR**, **S_IWUSR**, and **S_IXUSR**.

S_ISGID
Privilege to set group ID (GID) for execution. When this file is run through an `exec` function, the effective group ID of the process is set to the group ID of the file. The process then has the same authority as the file owner, rather than the authority of the actual invoker.

S_ISUID
Privilege to set the user ID (UID) for execution. When this file is run through an `exec` function, the effective user ID of the process is set to the owner of the file. The process then has the same authority as the file owner, rather than the authority of the actual invoker.

S_ISVTX
Indicates shared text. Keep loaded as an executable file in storage.

S_IWGRP
Write permission for the file's group.

S_IWOTH
Write permission for users other than the file owner.

S_IWUSR
Write permission for the file owner.

S_IXGRP
Search permission (for a directory) or execute permission (for a file) for the file's group.

S_IXOTH
Search permission for a directory, or execute permission for a file, for users other than the file owner.

S_IXUSR
Search permission (for a directory) or execute permission (for a file) for the file owner.

The owner ID of the new directory is set to the effective user ID of the process. The group ID of the new directory is set to the group ID of the owning directory.

mkdir() sets the access, change, and modification times for the new directory. It also sets the change and modification times for the directory that contains the new directory.

If *pathname* names a symbolic link, mkdir() fails.

Returned Value

If successful, mkdir() returns 0.

If unsuccessful, mkdir() does not create a directory, returns -1, and sets errno to one of the following values:

Error Code

	Description
EACCESS	The process did not have search permission on some component of <i>pathname</i> , or did not have write permission on the parent directory of the directory to be created.
EEXIST	Either the named file refers to a symbolic link, or there is already a file or directory with the given <i>pathname</i> .
ELOOP	A loop exists in symbolic links. This error is issued if more than POSIX_SYMLoop (defined in the limits.h header file) symbolic links are detected in the resolution of <i>pathname</i> .
EMLINK	The link count of the parent directory has already reached LINK_MAX (defined in the limits.h header file).
ENAMETOOLONG	<i>pathname</i> is longer than PATH_MAX characters or some component of <i>pathname</i> is longer than NAME_MAX characters while _POSIX_NO_TRUNC is in effect. For symbolic links, the length of the <i>pathname</i> string substituted for a symbolic link exceeds PATH_MAX . The PATH_MAX and NAME_MAX values can be determined using pathconf().
ENOENT	Some component of <i>pathname</i> does not exist, or <i>pathname</i> is an empty string.
ENOSPC	The file system does not have enough space to contain a new directory, or the parent directory cannot be extended.
ENOTDIR	A component of the <i>pathname</i> prefix is not a directory.
EROFS	The parent directory of the directory to be created is on a read-only file system.