NAME

getgrent, setgrent, endgrent - get group file entry

SYNOPSIS

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

struct group *getgrent(void);

void setgrent(void);

void endgrent(void);
```

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

setgrent():
_SVID_SOURCE || _BSD_SOURCE || _XOPEN_SOURCE >= 500 ||
_XOPEN_SOURCE && _XOPEN_SOURCE_EXTENDED ||
/* Since glibc 2.12: */ _POSIX_C_SOURCE >= 200809L

getgrent(), **endgrent**():
_SVID_SOURCE || _BSD_SOURCE || _XOPEN_SOURCE >= 500 ||
_XOPEN_SOURCE && _XOPEN_SOURCE_EXTENDED

DESCRIPTION

The **getgrent**() function returns a pointer to a structure containing the broken-out fields of a record in the group database (e.g., the local group file */etc/group*, NIS, and LDAP). The first time **getgrent**() is called, it returns the first entry; thereafter, it returns successive entries.

The **setgrent**() function rewinds to the beginning of the group database, to allow repeated scans.

The **endgrent**() function is used to close the group database after all processing has been performed.

The *group* structure is defined in <*grp.h*> as follows:

```
struct group {
char *gr_name; /* group name */
char *gr_passwd; /* group password */
gid_t gr_gid; /* group ID */
char **gr_mem; /* NULL-terminated array of pointers
to names of group members */
};
```

For more information about the fields of this structure, see group(5).

RETURN VALUE

The **getgrent**() function returns a pointer to a *group* structure, or NULL if there are no more entries or an error occurs.

Upon error, *errno* may be set. If one wants to check *errno* after the call, it should be set to zero before the call.

The return value may point to a static area, and may be overwritten by subsequent calls to **getgrent**(), getgrgid(3), or getgrnam(3). (Do not pass the returned pointer to free(3).)

ERRORS

EAGAIN
The service was temporarily unavailable; try again later. For NSS backends in glibc this indicates a temporary error talking to the backend. The error may correct itself, retrying later is suggested.

EINTR
A signal was caught.

EIO I/O error.

EMFILE
The calling process already has too many open files.

ENFILE
Too many open files in the system.

ENOENT
A necessary input file cannot be found. For NSS backends in glibc this indicates the backend is not correctly configured.

ENOMEM
Insufficient memory to allocate *group* structure.

ERANGE
Insufficient buffer space supplied.

FILES

/etc/group
local group database file

ATTRIBUTES

Multithreading (see pthreads(7))

The **getgrent**() function is not thread-safe.

The **setgrent**() and **endgrent**() functions are thread-safe.

CONFORMING TO

SVr4, 4.3BSD, POSIX.1-2001.

SEE ALSO

fgetgrent(3), getgrent_r(3), getgrgid(3), getgrnam(3), getgrouplist(3), putgrent(3), group(5)

COLOPHON

This page is part of release 3.74 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at http://www.kernel.org/doc/man-pages/.