Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'readdir.3' command

$ man readdir.3

READDIR(3) Linux Programmer's Manual READDIR(3)

NAME

readdir - read a directory

SYNOPSIS

```
#include <dirent.h>

struct dirent *readdir(DIR *dirp);
```

DESCRIPTION

The readdir() function returns a pointer to a dirent structure repre? senting the next directory entry in the directory stream pointed to by dirp. It returns NULL on reaching the end of the directory stream or if an error occurred.

In the glibc implementation, the dirent structure is defined as fol? lows:

```
struct dirent {
    ino_t          d_ino;       /* Inode number */
    off_t          d_off;       /* Not an offset; see below */
    unsigned short d_reclen;    /* Length of this record */
    unsigned char  d_type;      /* Type of file; not supported
                                   by all filesystem types */
    char           d_name[256]; /* Null-terminated filename */
};
```

The only fields in the dirent structure that are mandated by POSIX.1 are d_name and d_ino. The other fields are unstandardized, and not

present on all systems; see NOTES below for some further details.

The fields of the dirent structure are as follows:

d_ino This is the inode number of the file.

d_off The value returned in d_off is the same as would be returned by calling telldir(3) at the current position in the directory stream. Be aware that despite its type and name, the d_off field is seldom any kind of directory offset on modern filesys? tems. Applications should treat this field as an opaque value, making no assumptions about its contents; see also telldir(3).

d_reclen

This is the size (in bytes) of the returned record. This may not match the size of the structure definition shown above; see NOTES.

d_type This field contains a value indicating the file type, making it possible to avoid the expense of calling lstat(2) if further ac? tions depend on the type of the file.

When a suitable feature test macro is defined (_DEFAULT_SOURCE on glibc versions since 2.19, or _BSD_SOURCE on glibc versions 2.19 and earlier), glibc defines the following macro constants for the value returned in d_type:

DT_BLK This is a block device.

DT_CHR This is a character device.

DT_DIR This is a directory.

DT_FIFO This is a named pipe (FIFO).

DT_LNK This is a symbolic link.

DT_REG This is a regular file.

DT_SOCK This is a UNIX domain socket.

DT_UNKNOWN The file type could not be determined.

Currently, only some filesystems (among them: Btrfs, ext2, ext3, and ext4) have full support for returning the file type in d_type. All applications must properly handle a return of DT_UNKNOWN.

d_name This field contains the null terminated filename. See NOTES.

The data returned by readdir() may be overwritten by subsequent calls to readdir() for the same directory stream.

RETURN VALUE

On success, readdir() returns a pointer to a dirent structure. (This structure may be statically allocated; do not attempt to free(3) it.)

If the end of the directory stream is reached, NULL is returned and er? rno is not changed. If an error occurs, NULL is returned and errno is set appropriately. To distinguish end of stream from an error, set er? rno to zero before calling readdir() and then check the value of errno if NULL is returned.

ERRORS

EBADF Invalid directory stream descriptor dirp.

ATTRIBUTES

For an explanation of the terms used in this section, see at? tributes(7).

Interface	Attribute	Value
readdir()	Thread safety	MT-Unsafe race:dirstream

In the current POSIX.1 specification (POSIX.1-2008), readdir() is not required to be thread-safe. However, in modern implementations (in? cluding the glibc implementation), concurrent calls to readdir() that specify different directory streams are thread-safe. In cases where multiple threads must read from the same directory stream, using read? dir() with external synchronization is still preferable to the use of the deprecated readdir_r(3) function. It is expected that a future version of POSIX.1 will require that readdir() be thread-safe when con? currently employed on different directory streams.

CONFORMING TO

POSIX.1-2001, POSIX.1-2008, SVr4, 4.3BSD.

NOTES

A directory stream is opened using opendir(3).

The order in which filenames are read by successive calls to readdir() depends on the filesystem implementation; it is unlikely that the names will be sorted in any fashion.

Only the fields d_name and (as an XSI extension) d_ino are specified in POSIX.1. Other than Linux, the d_type field is available mainly only on BSD systems. The remaining fields are available on many, but not all systems. Under glibc, programs can check for the availability of the fields not defined in POSIX.1 by testing whether the macros _DI? RENT_HAVE_D_NAMLEN, _DIRENT_HAVE_D_RECLEN, _DIRENT_HAVE_D_OFF, or _DI? RENT_HAVE_D_TYPE are defined.

The d_name field

The dirent structure definition shown above is taken from the glibc headers, and shows the d_name field with a fixed size.

Warning: applications should avoid any dependence on the size of the d_name field. POSIX defines it as char d_name[], a character array of unspecified size, with at most NAME_MAX characters preceding the termi? nating null byte ('\0').

POSIX.1 explicitly notes that this field should not be used as an lvalue. The standard also notes that the use of sizeof(d_name) is in? correct; use strlen(d_name) instead. (On some systems, this field is defined as char d_name[1]!) By implication, the use sizeof(struct di? rent) to capture the size of the record including the size of d_name is also incorrect.

Note that while the call

```
fpathconf(fd, _PC_NAME_MAX)
```

returns the value 255 for most filesystems, on some filesystems (e.g., CIFS, Windows SMB servers), the null-terminated filename that is (cor? rectly) returned in d_name can actually exceed this size. In such cases, the d_reclen field will contain a value that exceeds the size of the glibc dirent structure shown above.

SEE ALSO

getdents(2), read(2), closedir(3), dirfd(3), ftw(3), offsetof(3), opendir(3), readdir_r(3), rewinddir(3), scandir(3), seekdir(3),

telldir(3)

COLOPHON

This page is part of release 5.10 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 https://www.kernel.org/doc/man-pages/.