NAME

dir, **dirent** — directory file format

SYNOPSIS

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

DESCRIPTION

Directories provide a convenient hierarchical method of grouping files while obscuring the underlying details of the storage medium. A directory file is differentiated from a plain file by a flag in its `inode`(5) entry. It consists of records (directory entries) each of which contains information about a file and a pointer to the file itself. Directory entries may contain other directories as well as plain files; such nested directories are refered to as subdirectories. A hierarchy of directories and files is formed in this manner and is called a file system (or referred to as a file system tree).

Each directory file contains two special directory entries; one is a pointer to the directory itself called dot '`.`' and the other a pointer to its parent directory called dot-dot '`..`'. Dot and dot-dot are valid pathnames, however, the system root directory '`/`', has no parent and dot-dot points to itself like dot.

File system nodes are ordinary directory files on which has been grafted a file system object, such as a physical disk or a partitioned area of such a disk. (See `mount`(1) and `mount`(8).)

The directory entry format is defined in the file ⟨sys/dirent.h⟩ and further in the file ⟨dirent.h⟩. When the macro `_DARWIN_FEATURE_64_BIT_INODE` is not defined (see `stat`(2) for more information on this macro), the *`dirent`* structure is defined as:

```
/*** Excerpt from <sys/dirent.h> ***/
/*
 * The dirent structure defines the format of directory entries.
 *
 * A directory entry has a struct dirent at the front of it, containing its
 * inode number, the length of the entry, and the length of the name
 * contained in the entry.  These are followed by the name padded to a 4
 * byte boundary with null bytes.  All names are guaranteed null terminated.
 * The maximum length of a name in a directory is 255.
 */

struct dirent { /* when _DARWIN_FEATURE_64_BIT_INODE is NOT defined */
        ino_t      d_ino;               /* file number of entry */
        __uint16_t d_reclen;            /* length of this record */
        __uint8_t  d_type;              /* file type, see below */
        __uint8_t  d_namlen;            /* length of string in d_name */
        char    d_name[255 + 1];   /* name must be no longer than this */
};
```

However, when the macro `_DARWIN_FEATURE_64_BIT_INODE` is defined, the *`dirent`* structure is defined as:

```
/*
 * The dirent structure defines the format of directory entries.
 *
 * A directory entry has a struct dirent at the front of it, containing its
 * inode number, the length of the entry, and the length of the name
 * contained in the entry.  These are followed by the name padded to a 4
 * byte boundary with null bytes.  All names are guaranteed null terminated.
 * The maximum length of a name in a directory is 1023.
 */

struct dirent { /* when _DARWIN_FEATURE_64_BIT_INODE is defined */
        ino_t      d_fileno;     /* file number of entry */
```

```
        __uint64_t d_seekoff;    /* seek offset (optional, used by servers) */
        __uint16_t d_reclen;     /* length of this record */
        __uint16_t d_namlen;     /* length of string in d_name */
        __uint8_t  d_type;       /* file type, see below */
        char    d_name[1024];    /* name must be no longer than this */
};
```

In addition:

```
/*
 * File types
 */
#define DT_UNKNOWN       0
#define DT_FIFO          1
#define DT_CHR           2
#define DT_DIR           4
#define DT_BLK           6
#define DT_REG           8
#define DT_LNK          10
#define DT_SOCK         12
#define DT_WHT          14
```

```
/*** Excerpt from <dirent.h> ***/

#define d_fileno        d_ino        /* backward compatibility */

/* definitions for library routines operating on directories. */
#define DIRBLKSIZ       1024

struct _telldir;                /* see telldir.h */

/* structure describing an open directory. */
typedef struct _dirdesc {
        int     __dd_fd;      /* file descriptor associated with directory */
        long    __dd_loc;     /* offset in current buffer */
        long    __dd_size;    /* amount of data returned by getdirentries */
        char    *__dd_buf;    /* data buffer */
        int     __dd_len;     /* size of data buffer */
        long    __dd_seek;    /* magic cookie returned by getdirentries */
        long    __dd_rewind;  /* magic cookie for rewinding */
        int     __dd_flags;   /* flags for readdir */
        pthread_mutex_t __dd_lock; /* for thread locking */
        struct _telldir *__dd_td; /* telldir position recording */
} DIR;

#define dirfd(dirp)     ((dirp)->dd_fd)

/* flags for opendir2 */
#define DTF_HIDEW       0x0001  /* hide whiteout entries */
#define DTF_NODUP       0x0002  /* don't return duplicate names */
#define DTF_REWIND      0x0004  /* rewind after reading union stack */
```

```
#define __DTF_READALL   0x0008  /* everything has been read */
```

SEE ALSO

`fs`(5), `inode`(5)

HISTORY

A **`dir`** file format appeared in Version 7 AT&T UNIX.