

NAME

getinode, **putinode** - fetch and store inodes on a UFS file system

LIBRARY

UFS File System Access Library (libufs, -lufs)

SYNOPSIS

```
#include <ufs/ufs/dinode.h>
```

```
#include <ufs/ufs/fs.h>
```

```
#include <libufs.h>
```

```
int
```

```
getinode(struct uufsd *disk, union dinodep *dp, ino_t inumber);
```

```
int
```

```
putinode(struct uufsd *disk);
```

DESCRIPTION

The **getinode()** and **putinode()** functions provide an inode fetch and store API for libufs(3) consumers. They operate on a userland UFS disk structure. The **getinode()** function fetches the specified inode from the filesystem. The **putinode()** function stores the most recently fetched inode to the filesystem.

The *dinodep* union is defined as:

```
union dinodep {
    struct ufs1_dinode *dp1;
    struct ufs2_dinode *dp2;
};
```

Sample code to clear write permissions for inode number *inumber* stored on the filesystem described by *diskp*.

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

```
#include <err.h>
```

```
#include <ufs/ufs/dinode.h>
```

```
#include <ufs/ufs/fs.h>
```

```
#include <libufs.h>
```

```
void
```

```

clearwrite(struct uufsd *diskp, ino_t inumber)
{
    union dinodep dp;

    if (getinode(diskp, &dp, inumber) == -1)
        err(1, "getinode: %s", diskp->d_error);
    switch (diskp->d_ufs) {
    case 1: /* UFS 1 filesystem */
        dp.dp1->di_mode &= ~(S_IWUSR | S_IWGRP | S_IWOTH);
        break;
    case 2: /* UFS 2 filesystem */
        dp.dp2->di_mode &= ~(S_IWUSR | S_IWGRP | S_IWOTH);
        break;
    default:
        errx(1, "unknown filesystem type");
    }
    if (putinode(diskp) == -1)
        err(1, "putinode: %s", diskp->d_error);
}

```

RETURN VALUES

The **getinode()** and **putinode()** functions return 0 on success, or -1 in case of any error. A string describing the error is stored in *diskp->d_error*. The global *errno* often provides additional information.

ERRORS

The function **getinode()** may fail and set *errno* for any of the errors specified for the library function **pread(2)**. It can also fail if the inode number is out of the range of inodes in the filesystem.

The function **putinode()** may fail and set *errno* for any of the errors specified for the library functions **ufs_disk_write(3)** or **pwrite(2)**.

Additionally both functions may follow the **libufs(3)** error methodologies in case of a device error.

SEE ALSO

pread(2), **pwrite(2)**, **libufs(3)**, **ufs_disk_write(3)**

HISTORY

These functions first appeared as part of **libufs(3)** in FreeBSD 13.0.

AUTHORS

Marshall Kirk McKusick <*mckusick@FreeBSD.org*>