

HFS Utilities

Complete Technical Manual

HFS and HFS+ Filesystem Utilities for Unix Systems

hfsutils Project

December 17, 2025

Version 3.2.6

Copyright and License

This manual documents the HFS Utilities (hfsutils) project, a collection of tools for creating, checking, mounting, and manipulating HFS and HFS+ filesystems on Unix-like systems including Linux, BSD, and macOS.

License: GNU General Public License v2 or later

Project: <https://github.com/JotaRandom/hfsutils>

Maintainer: hfsutils Project Contributors

This manual is provided as-is without any warranty. The authors are not responsible for any data loss or system damage resulting from the use of these utilities.

Contents

Chapter 1

Introduction

1.1 Overview

The HFS Utilities (`hfsutils`) project provides a comprehensive set of tools for working with Apple's HFS (Hierarchical File System) and HFS+ (HFS Plus) filesystems on Unix-like operating systems including Linux, BSD, and macOS.

1.1.1 Purpose and Goals

HFS and HFS+ filesystems are commonly used on:

- Classic Mac OS systems (HFS)
- Mac OS X and macOS systems prior to APFS (HFS+)
- iPod devices
- External drives formatted for Mac compatibility
- Disk images and backups from Apple systems

This toolset enables Unix systems to:

- **Create** HFS and HFS+ filesystems with `mkfs.hfs` and `mkfs.hfs+`
- **Check and repair** filesystem integrity with `fsck.hfs` and `fsck.hfs+`
- **Mount** HFS and HFS+ volumes with `mount.hfs` and `mount.hfs+` (requires kernel support)
- **Manipulate files** on HFS volumes without mounting using `hfsutil` commands (useful on systems without HFS kernel drivers)

1.1.2 Supported Systems

The utilities work on any POSIX-compliant system with appropriate kernel support:

Note: On systems without kernel HFS support, the `hfsutil` commands can still be used to access HFS filesystem contents.

System	HFS Support	HFS+ Support
Linux	Kernel module <code>hfs</code>	Kernel module <code>hfsplus</code>
FreeBSD	Native support	Native support
macOS	Native support	Native support
OpenBSD	Via FUSE	Via FUSE
NetBSD	Via FUSE	Via FUSE

Table 1.1: Platform Support Matrix

1.1.3 Key Features

Full Specification Compliance

All utilities strictly adhere to:

- Inside Macintosh: Files (HFS specification)
- Apple Technical Note TN1150 (HFS+ specification)
- Unix/POSIX filesystem utility standards

Zero-Tolerance Validation

The test suite implements a "zero-tolerance" policy:

- ANY deviation from specification = test failure
- ALL filesystems must be 100% correct
- Complete validation before and after `fsck` operations

Journaling Support

HFS+ journaling is supported with appropriate warnings:

- Journal creation with `mkfs.hfs+ -j`
- Journal validation and replay in `fsck.hfs+`
- Linux kernel compatibility warnings (journaling not supported in Linux HFS+ driver)

Date Limit Awareness

The utilities handle filesystem date limits correctly:

- HFS: Maximum date February 6, 2028 (Y2K28 problem)
- HFS+: Maximum date February 6, 2040 (Y2K40 problem)
- Automatic date correction to safe values

1.2 Installation

1.2.1 Building from Source

Prerequisites

```
1 # Debian/Ubuntu
2 sudo apt-get install build-essential
3
4 # Fedora/RHEL
5 sudo dnf install gcc make
6
7 # macOS
8 xcode-select --install
9
10 # BSD
11 # Compiler included by default
```

Compilation

```
1 # Clone repository
2 git clone https://github.com/JotaRandom/hfsutils.git
3 cd hfsutils
4
5 # Build all utilities
6 make
7
8 # Build specific sets
9 make set-hfs          # mkfs.hfs, fsck.hfs, mount.hfs
10 make set-hfsplus      # mkfs.hfs+, fsck.hfs+, mount.hfs+
11
12 # Build with hfsutil
13 make all
```

Installation Options

Linux systems (with kernel HFS/HFS+ drivers):

```
1 sudo make install-linux PREFIX=/usr
```

Complete installation (filesystem utilities + hfsutil):

```
1 sudo make install-complete PREFIX=/usr/local
```

Individual utilities:

```
1 sudo make install-mkfs.hfs+ PREFIX=/usr
2 sudo make install-fsck.hfs PREFIX=/usr
3 sudo make install-mount.hfs+ PREFIX=/usr
```

1.2.2 Verifying Installation

After installation, verify the utilities are available:

```
1 mkfs.hfs --version
2 mkfs.hfs+ --version
3 fsck.hfs --version
```

```
4 fsck.hfs+ --version
5 mount.hfs --help
6 mount.hfs+ --help
7 hfsutil --version    # If installed
```

Check manpages:

```
1 man mkfs.hfs
2 man mkfs.hfs+
3 man fsck.hfs+
4 man mount.hfs+
```

1.3 Quick Start

1.3.1 Creating a Filesystem

Create an HFS filesystem:

```
1 # Create 10MB image file
2 dd if=/dev/zero of=test.img bs=1M count=10
3
4 # Format as HFS
5 mkfs.hfs -L "MyDisk" test.img
```

Create an HFS+ filesystem:

```
1 # Create 50MB image file
2 dd if=/dev/zero of=test.img bs=1M count=50
3
4 # Format as HFS+
5 mkfs.hfs+ -L "MyDisk" test.img
6
7 # Format with journaling (Linux warning will appear)
8 mkfs.hfs+ -j -L "MyDisk" test.img
```

1.3.2 Checking a Filesystem

Verify filesystem integrity:

```
1 # Check HFS filesystem (read-only)
2 fsck.hfs -n test.img
3
4 # Check and repair HFS+ filesystem
5 fsck.hfs+ -y test.img
6
7 # Verbose output
8 fsck.hfs+ -v -n test.img
```

1.3.3 Mounting a Filesystem

On systems with kernel HFS/HFS+ support:

```
1 # Create mount point
2 sudo mkdir /mnt/hfs
3
4 # Mount read-write
5 sudo mount.hfs+ test.img /mnt/hfs
```

```
6 |
7 | # Mount read-only
8 | sudo mount.hfs+ -r test.img /mnt/hfs
9 |
10 | # Unmount
11 | sudo umount /mnt/hfs
```

1.3.4 Using hfsutil Commands

On systems without kernel support or for direct access:

```
1 | # Format volume
2 | hfsutil hformat -L "MyDisk" test.img
3 |
4 | # Mount in hfsutil
5 | hfsutil hmount test.img
6 |
7 | # List contents
8 | hfsutil hls
9 |
10 | # Copy file into volume
11 | hfsutil hcopy myfile.txt :myfile.txt
12 |
13 | # Copy file out of volume
14 | hfsutil hcopy :myfile.txt retrieved.txt
15 |
16 | # Unmount
17 | hfsutil humount
```

1.4 Documentation Structure

This manual is organized as follows:

Chapter 2: HFS Specification Details of the classic HFS filesystem format

Chapter 3: HFS+ Specification Details of the HFS+ filesystem format with journaling

Chapter 4: mkfs Utilities Filesystem creation tools

Chapter 5: fsck Utilities Filesystem checking and repair tools

Chapter 6: mount Utilities Filesystem mounting tools

Chapter 7: hfsutil Commands File manipulation without mounting

Chapter 8: Implementation Details Internal architecture and algorithms

Chapter 9: Testing and Validation Test suite and quality assurance

Chapter 10: Appendix Structure definitions, error codes, glossary

Chapter 2

HFS Specification

2.1 HFS Filesystem Overview

The Hierarchical File System (HFS) is the filesystem used by Apple Computer for Mac OS systems from 1985 until Mac OS X. Also known as "Mac OS Standard" or "HFS Classic", it provides a hierarchical directory structure with support for file and folder metadata.

2.1.1 Key Characteristics

- **Maximum volume size:** 2 TB
- **Maximum file size:** 2 GB
- **Filename length:** 31 characters (Macintosh Roman encoding)
- **Date range:** January 1, 1904 to February 6, 2028 (Y2K28 limit)
- **Allocation block size:** 512 bytes to 64 KB
- **Case sensitivity:** Case-insensitive, case-preserving

2.1.2 Volume Structure

An HFS volume is divided into logical blocks (512 bytes each) and allocation blocks (multiples of logical blocks).

Block	Name	Description
0-1	Boot Blocks	System startup information (1024 bytes)
2	Master Directory Block	Volume metadata and B-tree locations
3+	Allocation Bitmap	Block allocation map
...	Extents B-tree	File extent records
...	Catalog B-tree	Directory and file records
...	Data Area	File contents
Last-1	Alternate MDB	Backup of Master Directory Block

Table 2.1: HFS Volume Layout

2.2 Master Directory Block

The Master Directory Block (MDB) is located at logical block 2 (offset 1024 bytes) and contains critical volume information.

2.2.1 MDB Structure

Field	Offset	Description
drSigWord	+0	Signature (0x4244 = 'BD')
drCrDate	+2	Volume creation date
drLsMod	+6	Last modification date
drAtrb	+10	Volume attributes (bit 8 = unmounted cleanly)
drNmFls	+12	Number of files in root directory
drVBMSt	+14	First allocation bitmap block
drAllocPtr	+16	Next unused allocation block
drNmAlBlks	+18	Number of allocation blocks
drAlBlkSiz	+20	Allocation block size (bytes)
drClpSiz	+24	Default clump size
drAlBlkSt	+26	First allocation block
drNxtCNID	+28	Next unused Catalog Node ID
drFreeBks	+32	Number of free allocation blocks
drVN	+36	Volume name (1-27 characters, Pascal string)
drVolBkUp	+64	Last backup date
drVSeqNum	+68	Volume backup sequence number
drWrCnt	+70	Volume write count
drXTClpSiz	+72	Extents B-tree clump size
drCTClpSiz	+76	Catalog B-tree clump size
drNmRtDirs	+80	Number of directories in root
drFilCnt	+82	Number of files on volume
drDirCnt	+86	Number of directories on volume
drFndrInfo	+90	Finder information (32 bytes)
drVCSiz	+122	Volume cache size
drVBMCSiz	+124	Volume bitmap cache size
drCtlCSiz	+126	Common volume cache size
drXTFlSiz	+128	Extents B-tree file size
drXTExtRec	+132	Extents B-tree extents (3 extents)
drCTFlSiz	+144	Catalog B-tree file size
drCTExtRec	+148	Catalog B-tree extents (3 extents)

Table 2.2: Master Directory Block Fields

2.2.2 Critical MDB Fields

drSigWord (Signature)

- Value: 0x4244 ('BD' in ASCII)
- Identifies the volume as HFS
- **Validation:** Must be exactly 0x4244

drAtrb (Attributes)

Bit flags indicating volume state:

- Bit 8 (0x0100): Volume unmounted properly

- Bit 9 (0x0200): Spare blocks exist
- Bit 10 (0x0400): Volume needs consistency check
- Bit 15 (0x8000): Software lock

Important: mkfs.hfs sets bit 8 to indicate clean unmount.

drNxtCNID (Next Catalog Node ID)

- Minimum value: 16
- IDs 1-15 are reserved:
 - 1 = Parent of root directory
 - 2 = Root directory
 - 3 = Extents B-tree file
 - 4 = Catalog B-tree file
 - 5 = Bad block file
- **Validation:** mkfs.hfs initializes to 16

2.2.3 Alternate MDB

The alternate MDB is a complete copy of the MDB located at:

$$\textit{AlternateMDBoffset} = \textit{volume_size} - 1024\textit{bytes} \quad (2.1)$$

Purpose: Provides recovery if the primary MDB is corrupted.

Validation: fsck.hfs verifies both MDBs match.

2.3 B-Trees

HFS uses B-trees for efficient file and directory lookups.

2.3.1 Extents B-Tree

Stores file extent records (physical block ranges):

- Maps file portions to disk blocks
- Handles fragmented files
- Key: File CNID + fork type + start block

2.3.2 Catalog B-Tree

Stores directory and file records:

- Directory entries
- File metadata (type, creator, dates, etc.)
- Key: Parent directory CNID + filename

2.4 Date Representation

HFS dates are 32-bit unsigned integers representing seconds since:

January 1, 1904 00:00:00 GMT

2.4.1 Y2K28 Problem

Maximum representable date:

$$1904 + 2^{32} / (365.25 \times 24 \times 3600) \approx 2040 \quad (2.2)$$

However, **HFS specification limits dates to February 6, 2028.**

Implementation: `mkfs.hfs` and `fsck.hfs` use `hfs_get_safe_time()` to ensure dates are within valid range.

2.5 Allocation Strategy

2.5.1 Allocation Blocks

Files are allocated in multiples of allocation blocks:

- Allocation block size determined by volume size
- Typical sizes: 512 B, 1 KB, 2 KB, 4 KB
- Larger blocks = less overhead, more waste for small files

2.5.2 Clump Size

Default allocation size when extending files:

- Reduces fragmentation
- Calculated as multiple of allocation blocks
- Stored in `drClpSiz` (MDB)

2.6 Compatibility Notes

2.6.1 Character Encoding

HFS uses Macintosh Roman encoding:

- 8-bit encoding
- Incompatible with UTF-8
- Characters >127 map differently than ISO-8859-1

Limitation: Filenames with non-ASCII characters may not display correctly on non-Mac systems.

2.6.2 Resource Forks

HFS supports dual-fork files:

- **Data fork:** Regular file contents
- **Resource fork:** Application-specific data

Note: Most Unix systems only support data forks. Resource forks are preserved but not directly accessible.

2.6.3 Type and Creator Codes

HFS files have 4-character type and creator codes:

- **Type:** File type (e.g., 'TEXT', 'PICT')
- **Creator:** Creating application (e.g., 'MSWD' for Microsoft Word)

Modern usage: Largely replaced by filename extensions.

Chapter 3

HFS+ Specification

3.1 HFS+ Overview

HFS+ (also known as HFS Plus or Mac OS Extended) is the successor to HFS.

This chapter is under development and will include:

- HFS+ volume structure
- Volume Header details
- B-tree enhancements
- Journaling support
- Unicode filenames
- Y2K40 date limits

Note: See `TN1150_HFS_PLUS_VOLUME_FORMAT.md` and `WIKIPEDIA_HFS_PLUS.md` in the `doc/` directory for detailed specifications.

Chapter 4

mkfs Utilities

4.1 mkfs.hfs - HFS Filesystem Creation

This chapter will include complete mkfs.hfs and mkfs.hfs+ documentation from manpages.

4.1.1 Command Synopsis

```
mkfs.hfs [-f] [-L label] device  
mkfs.hfs+ [-f] [-j] [-L label] [-s size] device
```

Note: Full content will be converted from doc/man/mkfs.hfs.8 and mkfs.hfplus.8

Chapter 5

fsck Utilities

5.1 fsck.hfs - HFS Filesystem Check

This chapter will include complete fsck.hfs and fsck.hfs+ documentation from manpages.

5.1.1 Command Synopsis

```
fsck.hfs [-dfnpvy] device  
fsck.hfs+ [-dfnpvy] device
```

Note: Full content will be converted from doc/man/fsck.hfs.8 and fsck.hfs+.8

Chapter 6

mount Utilities

6.1 mount.hfs - HFS Filesystem Mounting

This chapter will include complete mount.hfs and mount.hfs+ documentation from manpages.

6.1.1 Command Synopsis

```
mount.hfs [-o options] [-r] [-w] [-v] device mountpoint  
mount.hfs+ [-o options] [-r] [-w] [-v] device mountpoint
```

Note: Full content will be converted from doc/man/mount.hfs.8 and mount.hfs+.8

Chapter 7

hfsutil Commands

7.1 hfsutil Commands

This chapter will include hfsutil command documentation.

7.1.1 Commands

- hformat - Format HFS volumes
- hmount - Mount HFS volumes
- humount - Unmount HFS volumes
- hls - List directory contents
- hcopy - Copy files
- hmkdir - Create directories
- And more...

Note: Full content will be converted from doc/man/hfsutils.1 and related manpages

Chapter 8

Implementation Details

8.1 Implementation Details

This chapter will include implementation details from `HFS_IMPLEMENTATION_NOTES.md`

8.1.1 Topics to Cover

- Code structure
- B-tree algorithms
- Validation strategies
- Repair mechanisms

Chapter 9

Testing and Validation

9.1 Testing and Validation

This chapter will include test suite documentation.

9.1.1 Zero-Tolerance Policy

All filesystems must be 100% correct. Any deviation from specification results in test failure.

9.1.2 Test Suite

- test/test_mkfs.sh - Filesystem creation tests
- test/test_fsck.sh - Validation tests
- test/test_hfsutils.sh - Command tests

Appendix A

Appendix

A.1 Structure Definitions

This appendix will include complete structure definitions for HFS and HFS+.

A.2 Error Codes

This section will list all error codes used by the utilities.

A.3 Glossary

Allocation Block Unit of disk space allocation

B-tree Balanced tree data structure

Catalog Directory of all files and folders

CNID Catalog Node ID

Extent Contiguous range of allocation blocks

Fork Part of a file (data or resource)

MDB Master Directory Block (HFS)

Volume Header Main metadata structure (HFS+)

A.4 References

- Inside Macintosh: Files
- Apple Technical Note TN1150
- Linux HFS/HFS+ kernel documentation
- BSD filesystem documentation