ufatfs

A Unix overlay file system for VFAT and exFAT volumes

Mike Swanson

This manual documents ufatfs version 0.1.0 [fcd3475e1b] 2026-11-20 02:14:29.

Copyright © 2026 Mike Swanson

Redistribution and use in source (Texinfo) and "compiled" forms (HTML, PDF and so forth) with or without modification, are permitted provided the following conditions are met:

1. Redistributions of source code (Texinfo) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.
2. Redistributions in compiled form (converted to HTML, PDF and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

This documentation is provided by the UMSFUSE project "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the UMSFUSE project be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this documentation, even if advised of the possibility of such damage.

Table of Contents

1 Introduction 1
1.1 What is ufatfs 1
1.2 VFAT and exFAT's capabilities 1
1.3 Prior art 2

2 ufatfs programs 3
2.1 `ufatfs` 3
2.2 mkfs.ufatfs 4
2.3 fsck.ufatfs 5
2.4 debugfs.ufatfs 6

Appendix A Glossary 7

1 Introduction

1.1 What is ufatfs

ufatfs is a Unix overlay file system intended to be stored on VFAT and exFAT file systems. It extends the capabilities of these file systems with support for case sensitivity, file modes, ownership, special files (devices, sockets, and FIFOs), and symbolic and hard links. All the while it maintains compatibility with the VFAT and exFAT file systems by never modifying nor changing the low-level file system structure, allowing for systems as old as Windows NT 3.5 and Windows 95 to interact with file systems managed by ufatfs. It accomplishes this goal by using a SQLite database at the root of the file system, named `/.ufatfs.db`, accompanied by the directory `/.ufatfs` to manage hard link data.

ufatfs lifts all file naming restrictions through a process known as "mangling." Where VFAT and exFAT prohibit control characters, some special wildcard characters like `*` and `?`, or the double-quote punctuation mark, ufatfs permits the storage of all file names. Characters unable to be stored directly on VFAT or exFAT are replaced by underscores. Naming conflicts, like what might happen with the storage of two files whose names differ only by letter case, are resolved by altering the name before the extension, suffixing a tilde and number, similar to how VFAT generates short filenames.

1.2 VFAT and exFAT's capabilities

VFAT and exFAT both store filenames of up two 255 16-bit code units, which effectively translates into UTF-16.[1] Most of the printable ASCII range is supported, and all non-ACSII code points are supported as well.

The forbidden characters stem from the early PC DOS 1.0 days, and while a few characters DOS prohibited became legal with Windows NT 3.5 and Windows 95's VFAT, many remain illegal:

- All control code characters from U+0000–U+001F are not permitted in filenames.
- The following characters related to command line processing and paths: `" * ? : < > / \ |`.
- Device names `CON`, `PRN`, `AUX`, `NUL`, `COM0`–`COM9`, `LPT0`–`LPT9`, `CONIN$`, `CONOUT$` making up the whole of a basename are not permitted.

ufatfs permits all of these, with the exceptions of the NUL (U+0000) and `/` (U+002F) characters. VFAT and exFAT are also strictly case-insensitive, using a large Unicode case mapping to enforce this across the entire Basic Multilingual Plane. By default, ufatfs permits the storage and use of case-sensitive names.

[1] No UTF-16 compliance is actually enforced on Microsoft Windows, and it permits the possibility of storing almost any sequence of 16-bit characters. Unfortunately Linux's vfat and exfat drivers, as of November 2026, expect that all names are UTF-16 compliant and lossily converts unrepresentable sequences with `?` characters.

1.3 Prior art

ufatfs is inspired by UMSDOS, a Linux overlay file system first released in 1994 and survived for approximately the following decade. It had similar goals to ufatfs, but with some extra constraints on it. UMSDOS fit itself into the original design of FAT, where host names were limited to eight base name characters and three extension characters, with an even more restrictive character set than VFAT supports. Every directory on UMSDOS has an accompanying `--linux-.---` file that describes the member records. This system is incapable of giving the root directory any special modes or ownerships, and record fields are limited in ways that made sense in the 1990s, but no longer: 32-bit timestamps, 16-bit UIDs/GIDs, 8-bit device majors/minors, and file names limited to 220 bytes. The author of ufatfs reimplemented UMSDOS in the UMSFUSE (`https://fossil.chungy.name/umsfuse/`) project, allowing the use of UMSDOS on modern Linux systems. As it is most useful for backwards compatibility, extended capabilities to UMSDOS have been deemed out of scope.

posixovl (`https://github.com/jengelh/posixovl`) is another project with a similar goal to ufatfs: a VFAT-compatible overlay providing Unix capabilities. It predates ufatfs by a couple decades, and employs different strategies to the Unix metadata storage and filename restrictions. It stores a single metadata file next to each stored Unix name: `.pxovl.$mangled-unix-name`, costing an entire cluster's worth of storage for a few bytes of additional data. Its name mangling strategy also involves swapping out characters to embed their hexadecimal values inside of parantheses and a percent size, eg `%(3F)` for the `?` character. This reduces the available length of names dramatically. posixovl is also incapable of overcoming the case-insensitive nature of VFAT and does not permit storing file names that differ only in case.

ufatfs has learned from the failings of both prior systems, whether limitations were caused by historical FAT restrictions, or by poor design. All metadata for ufatfs is stored within a single `/.ufatfs.db` file at the root of the file system, this file itself is an SQLite database and space-efficient. Even if a Windows user enables the display of hidden files, there is not much "clutter" added to the file system view. Characters that need replacement for storage are swapped with a single underscore (_), keeping the original file name length limit of 255 characters intact.

2 ufatfs programs

2.1 ufatfs

`ufatfs` [‘`OPTION`’...] ‘`VOLUME`’ [‘`MOUNTPOINT`’]

‘`ufatfs`’ mounts a ufatfs `VOLUME` that has been prepared by ‘`mkfs.ufatfs`.’ The volume can be any directory on the tree, preferably the root of a VFAT or exFAT file system. If `MOUNTPOINT` is omitted, an “overmount” is performed, where the original tree is replaced by the ufatfs view for new processes.

Unmount a ufatfs file system with ‘`fusermount3 -u MOUNTPOINT`.’

`--verbose`
: Log debug messages.

`--version`
`-V`
: Prints the version.

`-o ro`
: Mount as read-only.

`-o attr_timeout=SECONDS`
: Attribute and entry cache time (default 1; entries 0 on a case-insensitive volume).

`-o fakeroot`
: Root-like permissions: you can read and write anything, and `chown`/`chmod` any file to any value. This is the default mode when mounted as a non-root user.

`-o permissions`
: The kernel checks stored owners and modes for access control. This is the default when the root user mounts a ufatfs file system.

`-o multithreaded`
: Use several FUSE threads, even on a volume using a rollback journal.

`-o reserve=BYTES`
: Host space kept free for the database, so files can still be deleted on a full volume. Optionally takes a suffix `k` or `m`. Defaults to 64 pages, or at most 1/20 of the host space. Use “0” to disable the reservation.

The following options come as standard FUSE options:

`--help`
`-h`
: Print a usage summary.

`-o debug`
`-d`
: Enable debug output (imples ‘`-f`’).

`-f`
: Foreground operation.

`-s`
: Disable multithreaded operation (default with a rollback journal).

`-o clone_fd`
: Use separate FUSE device fd for each thread (may improve performance).

`-o max_idle_threads=N`
: The maximum number of idle worker threads allowed (default: -1).

`-o max_threads=N`
: The maximum number of worker threads allowed (default: 10).

`-o allow_other`
: Allow access by all users.

`-o allow_root`
: Allow access by root.

`-o auto_unmount`
: Automatically unmount on process termination.

`-o io_uring`
: Enable io-uring.

`-o io_uring_q_depth=N`
: io-uring queue depth.

2.2 mkfs.ufatfs

`mkfs.ufatfs` [‘`OPTION`’...] ‘`VOLUME`’

Create a ufatfs metadata database (`/.ufatfs.db`) and the `/.ufatfs` store directories at the root of ‘`VOLUME`’. ‘`VOLUME`’ typically should be the root of a VFAT or exFAT volume, but this is not a requirement.

`-f`
: Forcefully replace an existing database and allow a non-empty ‘`VOLUME`’. The metadata database will not retain any old information, and existing host files are not automatically adopted, see Section 2.3 [fsck.ufatfs], page 5.

`-e`
: Give symbolic links, device files, FIFOs (named pipes), and sockets a zero-length placeholder (empty) file on the host.

`-i`
: Enable case-insensitive mode: `FOO` and `foo` are the same entry. Case folding is performed using exFAT’s table, except if ‘`-b`’ is used, in which only ASCII case-insensitivity applies.

`-b`
: File names are opaque series of bytes: any bytes except ‘`NUL`’ and ‘`/`’ are permitted in any sequence. Non-ASCII 0x80–0xff bytes are mapped to U+E080–U+E0FF on the host. The default mode enforces UTF-8 compliant file names, which matches VFAT and exFAT’s expectations that all names are Unicode.

`-s`
: Enable small profile mode, for small volumes such as floppy disks. SQLite database pages are set to 1024 bytes, WAL checkpoints every 16 pages, and the WAL (or journal) is cut to 16 KiB afterward. The default profile offers greater crash resiliance, but costs 4 MiB of storage for the WAL.

`-j MODE`
: Journaling mode: ‘`wal`’ (default) or rollback (SQLite’s ‘`journal_mode TRUNCATE`’). Rollback offers extreme space efficiency with reduced crash resilience.

`-p SIZE`
: Set the database page size, a power of two from 512 to 65536 bytes (default is 4096, or 1024 with ‘`-s`’).

`-u UID` Sets the owner of the root directory to '`UID`'.

`-g GID` Sets the group ownership of the root directory to '`GID`'.

`-m MODE` Sets the mode of the root directory, specified in octal (default: 755).

`-h` Prints usage summary.

`-V` Prints the version.

2.3 fsck.ufatfs

`fsck.ufatfs` ['`OPTION`'...] '`VOLUME`'

Check and repairs a ufatfs volume at '`VOLUME`', which must be unmounted.

`-n` Check only, change nothing (default without a terminal).

`-p`
`-a` Preen: fix what is safe to fix without prompting.

`-y` Assume yes to all prompts and fix everything.

`-f` Force a full check (always done, for compatibility with `fsck`).

`-v` Verbose operation.

`-u UID` Sets the owner of stray files to '`UID`' (default: your UID).

`-g GID` Sets the group owner of stray files to '`GID`' (default: your GID).

`--strays=adopt`
Give stray host files and directories a ufatfs entry. (This is default)

`--strays=delete`
Delete stray host files not present in ufatfs (prompts for confirmation, unless '`-y`' is used).

`--strays=ignore`
Report the presence of stray host files but neither adopt nor delete them.

`--no-hidden`
Do not check the FAT hidden attribute. By default it is checked and dot-files in ufatfs are marked as hidden on the FAT. (Skipped on non-FAT file systems.)

`--no-host-modes`
Do not check that a regular file's host write and execute permissions follow its ufatfs mode (write or execute bit set anywhere: set it on the host). Execute permissions are not supported on FAT.

`--no-host-times`
Do not check that host atime or mtime match the ufatfs database. By default, ufatfs atime and mtime timestamps are matched on the host file system, to the best possible effort (eg, FAT has a date range of 1980–2107).

`--vacuum` Rebuilds the database with SQLite's `VACUUM` command, optimizing database pages and frees empty space back to the host. Requires free space enough for a full database copy. (With '`-n`', only report what it would reclaim.)

`--profile=default|small`
: Change the database layout between `mkfs.ufatfs`'s default or small profiles, and always performs a `VACUUM`.

`--journal=wal|rollback`
: Change the database journaling mode, as with `mkfs.ufatfs`'s '`-j`' option.

`--page-size=N`
: Change the database page size, as with `mkfs.ufatfs`'s '`-p`' option, and always performs a `VACUUM`.

`-h`
: Prints usage summary.

`-V`
: Prints the version.

2.4 debugfs.ufatfs

`debugfs.ufatfs` ['`--no-warning`'] ['`VOLUME`'|'`DBFILE`'] ['`sqlite3 options`'] ['`SQL`']

Run the SQLite shell on a ufatfs metadata database, with the ufatfs SQL functions (`ufatfs_name_key_bytes`, `ufatfs_name_key`, `ufatfs_host_key`) available on the shell.

This permits full read and write capabilities to the database, but extreme caution must be used, especially with any non-`SELECT` queries issued against the database. It is possible to corrupt or get the database into a state ufatfs is not designed to cope with.

`--no-warning`
: Do not print the warning on startup.

`--help`
: Prints usage summary, including available SQLite shell options.

`--version`
`-V`
: Prints the version.

Standard SQLite shell options are available, use '`--help`' for a list of what '`debugfs.ufatfs`' is built with, or visit `https://sqlite.org/cli.html#command_line_options` for a detailed list.

Appendix A Glossary

8+3
8.3 A two-component file name limit that requires a basename of 1–8 characters, followed by an extension of 0–3 characters.

host name The file name as it appears on the low-level file system, such as the intended VFAT and exFAT targets for ufatfs.

Unix name
The file name as it is represented in ufatfs.

mangling An algorithm to transform Unix names into those compatible with VFAT and exFAT systems. Most names are capable of being translated as-is, with exceptions requiring replacement characters or special pre-extension suffixes.