File systems

(Redirected from [Mount](https://wiki.archlinux.org/index.php?title=Mount&redirect=no))

From [**Wikipedia**](https://en.wikipedia.org/wiki/File_system):

In computing, a file system (or filesystem) is used to control how data is stored and retrieved. Without a file system, information placed in a storage medium would be one large body of data with no way to tell where one piece of information stops and the next begins. By separating the data into pieces and giving each piece a name, the information is easily isolated and identified.

Taking its name from the way paper-based information systems are named, each group of data is called a "file". The structure and logic rules used to manage the groups of information and their names is called a "file system".

Individual drive partitions can be setup using one of the many different available filesystems. Each has its own advantages, disadvantages, and unique idiosyncrasies. A brief overview of supported filesystems follows; the links are to Wikipedia pages that provide much more information.

**Contents**

 [hide]

* [1Types of file systems](https://wiki.archlinux.org/index.php/File_systems#Types_of_file_systems)
  + [1.1Journaling](https://wiki.archlinux.org/index.php/File_systems#Journaling)
  + [1.2FUSE-based file systems](https://wiki.archlinux.org/index.php/File_systems#FUSE-based_file_systems)
  + [1.3Stackable file systems](https://wiki.archlinux.org/index.php/File_systems#Stackable_file_systems)
  + [1.4Read-only file systems](https://wiki.archlinux.org/index.php/File_systems#Read-only_file_systems)
  + [1.5Clustered file systems](https://wiki.archlinux.org/index.php/File_systems#Clustered_file_systems)
* [2Identify existing file systems](https://wiki.archlinux.org/index.php/File_systems#Identify_existing_file_systems)
* [3Create a file system](https://wiki.archlinux.org/index.php/File_systems#Create_a_file_system)
* [4Mount a file system](https://wiki.archlinux.org/index.php/File_systems#Mount_a_file_system)
  + [4.1List mounted file systems](https://wiki.archlinux.org/index.php/File_systems#List_mounted_file_systems)
  + [4.2Umount a file system](https://wiki.archlinux.org/index.php/File_systems#Umount_a_file_system)
* [5See also](https://wiki.archlinux.org/index.php/File_systems#See_also)

Types of file systems

See **[filesystems(5)](https://jlk.fjfi.cvut.cz/arch/manpages/man/filesystems.5)** for a general overview, and **[Wikipedia:Comparison of file systems](https://en.wikipedia.org/wiki/Comparison_of_file_systems" \o "wikipedia:Comparison of file systems)** for a detailed feature comparison. File systems supported by the kernel are listed in /proc/filesystems.

| **File system** | **Creation command** | **Userspace utilities** | [**Archiso**](https://wiki.archlinux.org/index.php/Archiso)[**[1]**](https://git.archlinux.org/archiso.git/tree/configs/releng/packages.both) | **Kernel documentation**[**[2]**](https://www.kernel.org/doc/Documentation/filesystems/) | **Notes** |
| --- | --- | --- | --- | --- | --- |
| [**Btrfs**](https://wiki.archlinux.org/index.php/Btrfs) | [**mkfs.btrfs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.btrfs.8) | [**btrfs-progs**](https://www.archlinux.org/packages/?name=btrfs-progs) | Yes | [**btrfs.txt**](https://www.kernel.org/doc/Documentation/filesystems/btrfs.txt) | [**Stability status**](https://btrfs.wiki.kernel.org/index.php/Status) |
| [**VFAT**](https://wiki.archlinux.org/index.php/VFAT) | [**mkfs.vfat(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.vfat.8) | [**dosfstools**](https://www.archlinux.org/packages/?name=dosfstools) | Yes | [**vfat.txt**](https://www.kernel.org/doc/Documentation/filesystems/vfat.txt) |  |
| [**exFAT**](https://en.wikipedia.org/wiki/exFAT) | [**mkexfatfs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkexfatfs.8) | [**exfat-utils**](https://www.archlinux.org/packages/?name=exfat-utils) | Yes | N/A (FUSE-based) |  |
| [**F2FS**](https://wiki.archlinux.org/index.php/F2FS) | [**mkfs.f2fs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.f2fs.8) | [**f2fs-tools**](https://www.archlinux.org/packages/?name=f2fs-tools) | Yes | [**f2fs.txt**](https://www.kernel.org/doc/Documentation/filesystems/f2fs.txt) | Flash-based devices |
| [**ext3**](https://wiki.archlinux.org/index.php/Ext3) | [**mke2fs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mke2fs.8) | [**e2fsprogs**](https://www.archlinux.org/packages/?name=e2fsprogs) | Yes ([**base**](https://www.archlinux.org/groups/x86_64/base/)) | [**ext3.txt**](https://www.kernel.org/doc/Documentation/filesystems/ext3.txt) |  |
| [**ext4**](https://wiki.archlinux.org/index.php/Ext4) | [**mke2fs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mke2fs.8) | [**e2fsprogs**](https://www.archlinux.org/packages/?name=e2fsprogs) | Yes ([**base**](https://www.archlinux.org/groups/x86_64/base/)) | [**ext4.txt**](https://www.kernel.org/doc/Documentation/filesystems/ext4.txt) |  |
| [**HFS**](https://en.wikipedia.org/wiki/Hierarchical_File_System) | mkfs.hfsplus(8) | [**hfsprogs**](https://aur.archlinux.org/packages/hfsprogs/)AUR | No | [**hfs.txt**](https://www.kernel.org/doc/Documentation/filesystems/hfs.txt) | [**macOS**](https://en.wikipedia.org/wiki/macOS) file system |
| [**JFS**](https://wiki.archlinux.org/index.php/JFS) | [**mkfs.jfs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.jfs.8) | [**jfsutils**](https://www.archlinux.org/packages/?name=jfsutils) | Yes ([**base**](https://www.archlinux.org/groups/x86_64/base/)) | [**jfs.txt**](https://www.kernel.org/doc/Documentation/filesystems/jfs.txt) |  |
| [**NILFS2**](https://en.wikipedia.org/wiki/NILFS) | [**mkfs.nilfs2(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.nilfs2.8) | [**nilfs-utils**](https://www.archlinux.org/packages/?name=nilfs-utils) | Yes | [**nilfs2.txt**](https://www.kernel.org/doc/Documentation/filesystems/nilfs2.txt) |  |
| [**NTFS**](https://wiki.archlinux.org/index.php/NTFS) | [**mkfs.ntfs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.ntfs.8) | [**ntfs-3g**](https://www.archlinux.org/packages/?name=ntfs-3g) | Yes | N/A (FUSE-based) | [**Windows**](https://en.wikipedia.org/wiki/Microsoft_Windows) file system |
| [**Reiser4**](https://wiki.archlinux.org/index.php/Reiser4) | mkfs.reiser4(8) | [**reiser4progs**](https://aur.archlinux.org/packages/reiser4progs/)AUR | No |  |  |
| [**ReiserFS**](https://en.wikipedia.org/wiki/ReiserFS) | [**mkfs.reiserfs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.reiserfs.8) | [**reiserfsprogs**](https://www.archlinux.org/packages/?name=reiserfsprogs) | Yes ([**base**](https://www.archlinux.org/groups/x86_64/base/)) |  |  |
| [**UDF**](https://en.wikipedia.org/wiki/Universal_Disk_Format) | [**mkfs.udf(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.udf.8) | [**udftools**](https://www.archlinux.org/packages/?name=udftools) | Optional | [**udf.txt**](https://www.kernel.org/doc/Documentation/filesystems/udf.txt) |  |
| [**XFS**](https://wiki.archlinux.org/index.php/XFS) | [**mkfs.xfs(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.xfs.8) | [**xfsprogs**](https://www.archlinux.org/packages/?name=xfsprogs) | Yes ([**base**](https://www.archlinux.org/groups/x86_64/base/)) | [**xfs.txt**](https://www.kernel.org/doc/Documentation/filesystems/xfs.txt) [**xfs-delayed-logging-design.txt**](https://www.kernel.org/doc/Documentation/filesystems/xfs-delayed-logging-design.txt) [**xfs-self-describing-metadata.txt**](https://www.kernel.org/doc/Documentation/filesystems/xfs-self-describing-metadata.txt) |  |
| [**ZFS**](https://wiki.archlinux.org/index.php/ZFS) |  | [**zfs-linux**](https://aur.archlinux.org/packages/zfs-linux/)AUR | No | N/A (**[OpenZFS](https://en.wikipedia.org/wiki/OpenZFS" \o "w:OpenZFS)** port) |  |

**Note:** The kernel has its own NTFS driver (see [**ntfs.txt**](https://www.kernel.org/doc/Documentation/filesystems/ntfs.txt)), but it has limited support for writing files.

**Journaling**

All the above filesystems with the exception of ext2, FAT16/32, Btrfs and ZFS, use [**journaling**](https://en.wikipedia.org/wiki/Journaling_file_system). Journaling provides fault-resilience by logging changes before they are committed to the filesystem. In the event of a system crash or power failure, such file systems are faster to bring back online and less likely to become corrupted. The logging takes place in a dedicated area of the filesystem.

Not all journaling techniques are the same. Ext3 and ext4 offer data-mode journaling, which logs both data and meta-data, as well as possibility to journal only meta-data changes. Data-mode journaling comes with a speed penalty and is not enabled by default. In the same vein, [**Reiser4**](https://wiki.archlinux.org/index.php/Reiser4) offers so-called [**"transaction models"**](https://reiser4.wiki.kernel.org/index.php/Reiser4_transaction_models), which include pure journaling (equivalent to ext4's data-mode journaling), pure Copy-on-Write approach (equivalent to btrfs' default) and a combined approach which heuristically alternates between the two former.

**Note:** Reiser4 does not provide an equivalent to ext4's default journaling behavior (meta-data only).

The other filesystems provide ordered-mode journaling, which only logs meta-data. While all journaling will return a filesystem to a valid state after a crash, data-mode journaling offers the greatest protection against corruption and data loss. There is a compromise in system performance, however, because data-mode journaling does two write operations: first to the journal and then to the disk. The trade-off between system speed and data safety should be considered when choosing the filesystem type.

Filesystems based on copy-on-write, such as Btrfs and ZFS, have no need to use traditional journal to protect metadata, because they are never updated in-place. Although Btrfs still has a journal-like log tree, it is only used to speed-up fdatasync/fsync.

**FUSE-based file systems**

[**Filesystem in Userspace**](https://en.wikipedia.org/wiki/Filesystem_in_Userspace) (FUSE) is a mechanism for Unix-like operating systems that lets non-privileged users create their own file systems without editing kernel code. This is achieved by running file system code in *user space*, while the FUSE kernel module provides only a "bridge" to the actual kernel interfaces.

Some FUSE-based file systems:

* **adbfs-git** — Mount an Android device connected via USB.

[**http://collectskin.com/adbfs/**](http://collectskin.com/adbfs/) || **[adbfs-git](https://aur.archlinux.org/packages/adbfs-git/)**AUR

* [**EncFS**](https://wiki.archlinux.org/index.php/EncFS) — EncFS is a userspace stackable cryptographic file-system.

[**https://vgough.github.io/encfs/**](https://vgough.github.io/encfs/) || **[encfs](https://www.archlinux.org/packages/?name=encfs)**

* **fuseiso** — Mount an ISO as a regular user.

[**http://sourceforge.net/projects/fuseiso/**](http://sourceforge.net/projects/fuseiso/) || **[fuseiso](https://www.archlinux.org/packages/?name=fuseiso)**

* [**gitfs**](https://wiki.archlinux.org/index.php/Gitfs) — gitfs is a FUSE file system that fully integrates with git.

[**https://www.presslabs.com/gitfs/**](https://www.presslabs.com/gitfs/) || **[gitfs](https://aur.archlinux.org/packages/gitfs/)**AUR

* [**gocryptfs**](https://wiki.archlinux.org/index.php/Gocryptfs) — gocryptfs is a userspace stackable cryptographic file-system.

[**https://nuetzlich.net/gocryptfs/**](https://nuetzlich.net/gocryptfs/) || **[gocryptfs](https://aur.archlinux.org/packages/gocryptfs/)**AUR

* [**SSHFS**](https://wiki.archlinux.org/index.php/SSHFS) — SSHFS is a FUSE-based filesystem client for mounting directories over SSH.

[**https://github.com/libfuse/sshfs**](https://github.com/libfuse/sshfs) || **[sshfs](https://www.archlinux.org/packages/?name=sshfs)**

* **vdfuse** — Mounting VirtualBox disk images (VDI/VMDK/VHD).

[**https://github.com/muflone/virtualbox-includes**](https://github.com/muflone/virtualbox-includes) || **[vdfuse](https://aur.archlinux.org/packages/vdfuse/)**AUR

* **xbfuse-git** — Mount an Xbox (360) ISO.

[**http://multimedia.cx/xbfuse/**](http://multimedia.cx/xbfuse/) || **[xbfuse-git](https://aur.archlinux.org/packages/xbfuse-git/)**AUR

* **xmlfs** — Represent an XML file as a directory structure for easy access.

[**https://github.com/halhen/xmlfs**](https://github.com/halhen/xmlfs) || **[xmlfs](https://aur.archlinux.org/packages/xmlfs/)**AUR

See **[Wikipedia:Filesystem in Userspace#Example uses](https://en.wikipedia.org/wiki/Filesystem_in_Userspace" \l "Example_uses" \o "wikipedia:Filesystem in Userspace)** for more.

**Stackable file systems**

* **aufs** — Advanced Multi-layered Unification Filesystem, a FUSE based union filesystem, a complete rewrite of Unionfs, was rejected from Linux mainline and instead OverlayFS was merged into the Linux Kernel.

[**http://aufs.sourceforge.net**](http://aufs.sourceforge.net/) || **[aufs](https://aur.archlinux.org/packages/aufs/)**AUR

* [**eCryptfs**](https://wiki.archlinux.org/index.php/ECryptfs) — The Enterprise Cryptographic Filesystem is a package of disk encryption software for Linux. It is implemented as a POSIX-compliant filesystem-level encryption layer, aiming to offer functionality similar to that of GnuPG at the operating system level.

[**http://ecryptfs.org**](http://ecryptfs.org/) || **[ecryptfs-utils](https://www.archlinux.org/packages/?name=ecryptfs-utils)**

* **mergerfs** — a FUSE based union filesystem.

[**https://github.com/trapexit/mergerfs**](https://github.com/trapexit/mergerfs) || **[mergerfs](https://aur.archlinux.org/packages/mergerfs/)**AUR

* **mhddfs** — Multi-HDD FUSE filesystem, a FUSE based union filesystem.

[**http://mhddfs.uvw.ru**](http://mhddfs.uvw.ru/) || **[mhddfs](https://aur.archlinux.org/packages/mhddfs/)**AUR

* [**overlayfs**](https://wiki.archlinux.org/index.php/Overlayfs) — OverlayFS is a filesystem service for Linux which implements a union mount for other file systems.

[**https://www.kernel.org/doc/Documentation/filesystems/overlayfs.txt**](https://www.kernel.org/doc/Documentation/filesystems/overlayfs.txt) || [**linux**](https://www.archlinux.org/packages/?name=linux)

* **Unionfs** — Unionfs is a filesystem service for Linux, FreeBSD and NetBSD which implements a union mount for other file systems.

[**http://unionfs.filesystems.org/**](http://unionfs.filesystems.org/) || not packaged? [**search in AUR**](https://aur.archlinux.org/packages/)

* **unionfs-fuse** — A user space Unionfs implementation.

[**https://github.com/rpodgorny/unionfs-fuse**](https://github.com/rpodgorny/unionfs-fuse) || **[unionfs-fuse](https://www.archlinux.org/packages/?name=unionfs-fuse)**

**Read-only file systems**

* [**SquashFS**](https://en.wikipedia.org/wiki/SquashFS) — SquashFS is a compressed read only filesystem. SquashFS compresses files, inodes and directories, and supports block sizes up to 1 MB for greater compression.

[**http://squashfs.sourceforge.net/**](http://squashfs.sourceforge.net/) || **[squashfs-tools](https://www.archlinux.org/packages/?name=squashfs-tools)**

**Clustered file systems**

* [**Ceph**](https://wiki.archlinux.org/index.php/Ceph) — Unified, distributed storage system designed for excellent performance, reliability and scalability.

[**https://ceph.com/**](https://ceph.com/) || **[ceph](https://www.archlinux.org/packages/?name=ceph)**

* [**Glusterfs**](https://wiki.archlinux.org/index.php/Glusterfs) — Cluster file system capable of scaling to several peta-bytes.

[**https://www.gluster.org/**](https://www.gluster.org/) || **[glusterfs](https://www.archlinux.org/packages/?name=glusterfs)**

* [**IPFS**](https://wiki.archlinux.org/index.php/IPFS) — A peer-to-peer hypermedia protocol to make the web faster, safer, and more open. IPFS aims replace HTTP and build a better web for all of us. Uses blocks to store parts of a file, each network node stores only content it is interested, provides deduplication, distribution, scalable system limited only by users. (currently in aplha)

[**https://ipfs.io/**](https://ipfs.io/) || [**go-ipfs**](https://www.archlinux.org/packages/?name=go-ipfs)

* [**MooseFS**](https://en.wikipedia.org/wiki/MooseFS) — MooseFS is a fault tolerant, highly available and high performance scale-out network distributed file system.

[**https://moosefs.com**](https://moosefs.com/) || **[moosefs](https://www.archlinux.org/packages/?name=moosefs)**

* [**OpenAFS**](https://wiki.archlinux.org/index.php/OpenAFS) — Open source implementation of the AFS distributed file system

[**http://www.openafs.org**](http://www.openafs.org/) || **[openafs](https://aur.archlinux.org/packages/openafs/)**AUR

* [**OrangeFS**](https://en.wikipedia.org/wiki/OrangeFS) — OrangeFS is a scale-out network file system designed for transparently accessing multi-server-based disk storage, in parallel. Has optimized MPI-IO support for parallel and distributed applications. Simplifies the use of parallel storage not only for Linux clients, but also for Windows, Hadoop, and WebDAV. POSIX-compatible. Part of Linux kernel since version 4.6.

[**http://www.orangefs.org/**](http://www.orangefs.org/) || not packaged? [**search in AUR**](https://aur.archlinux.org/packages/)

* **Sheepdog** — Distributed object storage system for volume and container services and manages the disks and nodes intelligently.

[**https://sheepdog.github.io/sheepdog/**](https://sheepdog.github.io/sheepdog/) || not packaged? [**search in AUR**](https://aur.archlinux.org/packages/)

* [**Tahoe-LAFS**](https://en.wikipedia.org/wiki/Tahoe-LAFS) — Tahoe Least-Authority Filesystem is a free and open, secure, decentralized, fault-tolerant, peer-to-peer distributed data store and distributed file system.

[**https://tahoe-lafs.org/**](https://tahoe-lafs.org/) || **[tahoe-lafs](https://aur.archlinux.org/packages/tahoe-lafs/)**AUR

Identify existing file systems

To identify existing file systems, you can use **[lsblk](https://wiki.archlinux.org/index.php/Lsblk" \o "Lsblk)**:

$ lsblk -f

NAME FSTYPE LABEL UUID MOUNTPOINT

sdb

└─sdb1 vfat Transcend 4A3C-A9E9

An existing file system, if present, will be shown in the FSTYPE column. If [**mounted**](https://wiki.archlinux.org/index.php/Mount), it will appear in the MOUNTPOINT column.

Create a file system

File systems are usually created on a [**partition**](https://wiki.archlinux.org/index.php/Partition), inside logical containers such as [**LVM**](https://wiki.archlinux.org/index.php/LVM), [**RAID**](https://wiki.archlinux.org/index.php/RAID) and **[dm-crypt](https://wiki.archlinux.org/index.php/Dm-crypt" \o "Dm-crypt)**, or on a regular file (see **[Wikipedia:Loop device](https://en.wikipedia.org/wiki/Loop_device" \o "wikipedia:Loop device)**). This section describes the partition case.

**Note:** File systems can be written directly to a disk, known as a *superfloppy* or **[partitionless disk](https://wiki.archlinux.org/index.php/Partitioning" \l "Partitionless_disk" \o "Partitioning)**. Certain limitations are involved with this method, particularly if [**booting**](https://wiki.archlinux.org/index.php/Arch_boot_process) from such a drive. See **[Btrfs#Partitionless Btrfs disk](https://wiki.archlinux.org/index.php/Btrfs" \l "Partitionless_Btrfs_disk" \o "Btrfs)** for an example.

**Warning:**

* After creating a new filesystem, data previously stored on this partition can unlikely be recovered. **Create a backup of any data you want to keep**.
* The purpose of a given partition may restrict the choice of file system. For example, an [**EFI System Partition**](https://wiki.archlinux.org/index.php/EFI_System_Partition) must contain a [**FAT32**](https://wiki.archlinux.org/index.php/FAT32) (mkfs.vfat) file system, and the file system containing the /boot directory must be supported by the [**boot loader**](https://wiki.archlinux.org/index.php/Boot_loader).

Before continuing, [**identify the device**](https://wiki.archlinux.org/index.php/Lsblk) where the file system will be created and whether or not it is mounted. For example:

$ lsblk -f

NAME FSTYPE LABEL UUID MOUNTPOINT

sda

├─sda1 C4DA-2C4D

├─sda2 ext4 5b1564b2-2e2c-452c-bcfa-d1f572ae99f2 /mnt

└─sda3 56adc99b-a61e-46af-aab7-a6d07e504652

Mounted file systems **must** be **[unmounted](https://wiki.archlinux.org/index.php/File_systems" \l "Umount_a_file_system)** before proceeding. In the above example an existing filesystem is on /dev/sda2 and is mounted at /mnt. It would be unmounted with:

# umount /dev/sda2

To find just mounted file systems, see [**#List mounted file systems**](https://wiki.archlinux.org/index.php/File_systems#List_mounted_file_systems).

To create a new file system, use **[mkfs(8)](https://jlk.fjfi.cvut.cz/arch/manpages/man/mkfs.8)**. See [**#Types of file systems**](https://wiki.archlinux.org/index.php/File_systems#Types_of_file_systems) for the exact type, as well as userspace utilities you may wish to install for a particular file system.

For example, to create a new file system of type [**ext4**](https://wiki.archlinux.org/index.php/Ext4) (common for Linux data partitions) on /dev/sda1, run:

# mkfs.ext4 /dev/sda1

**Tip:**

* Use the -L flag of *mkfs.ext4* to specify a [**file system label**](https://wiki.archlinux.org/index.php/Persistent_block_device_naming#by-label). *e2label* can be used to change the label on an existing file system.
* File systems may be *resized* after creation, with certain limitations. For example, an [**XFS**](https://wiki.archlinux.org/index.php/XFS) filesystem's size can be increased, but it cannot reduced. See [**Resize capabilities**](https://en.wikipedia.org/wiki/Comparison_of_file_systems#Resize_capabilities) and the respective file system documentation for details.

The new file system can now be mounted to a directory of choice.

Mount a file system

To manually mount filesystem located on a device (e.g., a partition) to a directory, use [**mount(8)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mount.8). This example mounts /dev/sda1 to /mnt.

# mount /dev/sda1 /mnt

This attaches the filesystem on /dev/sda1 at the directory /mnt, making the contents of the filesystem visible. Any data that existed at /mnt before this action is made invisible until the device is unmounted.

[**fstab**](https://wiki.archlinux.org/index.php/Fstab) contains information on how devices should be automatically mounted if present. See the **[fstab](https://wiki.archlinux.org/index.php/Fstab" \o "Fstab)** article for more information on how to modify this behavior.

If a device is specified in /etc/fstab and only the device or mount point is given on the command line, that information will be used in mounting. For example, if /etc/fstab contains a line indicating that /dev/sda1 should be mounted to /mnt, then the following will automatically mount the device to that location:

# mount /dev/sda1

Or

# mount /mnt

*mount* contains several options, many of which depend on the file system specified. The options can be changed, either by:

* using flags on the command line with *mount*
* editing **[fstab](https://wiki.archlinux.org/index.php/Fstab" \o "Fstab)**
* creating **[udev](https://wiki.archlinux.org/index.php/Udev" \o "Udev)** rules
* [**compiling the kernel yourself**](https://wiki.archlinux.org/index.php/Arch_Build_System)
* or using filesystem-specific mount scripts (located at /usr/bin/mount.\*).

See these related articles and the article of the filesystem of interest for more information.

**List mounted file systems**

To list all mounted file systems, use **[findmnt(8)](https://jlk.fjfi.cvut.cz/arch/manpages/man/findmnt.8)**:

$ findmnt

*findmnt* takes a variety of arguments which can filter the output and show additional information. For example, it can take a device or mount point as an argument to show only information on what is specified:

$ findmnt /dev/sda1

*findmnt* gathers information from /etc/fstab, /etc/mtab, and /proc/self/mounts.

**Umount a file system**

To unmount a file system use **[umount(8)](https://jlk.fjfi.cvut.cz/arch/manpages/man/umount.8)**. Either the device containing the file system (e.g., /dev/sda1) or the mount point (e.g., /mnt) can be specified:

# umount /dev/sda1

Or

# umount /mnt