



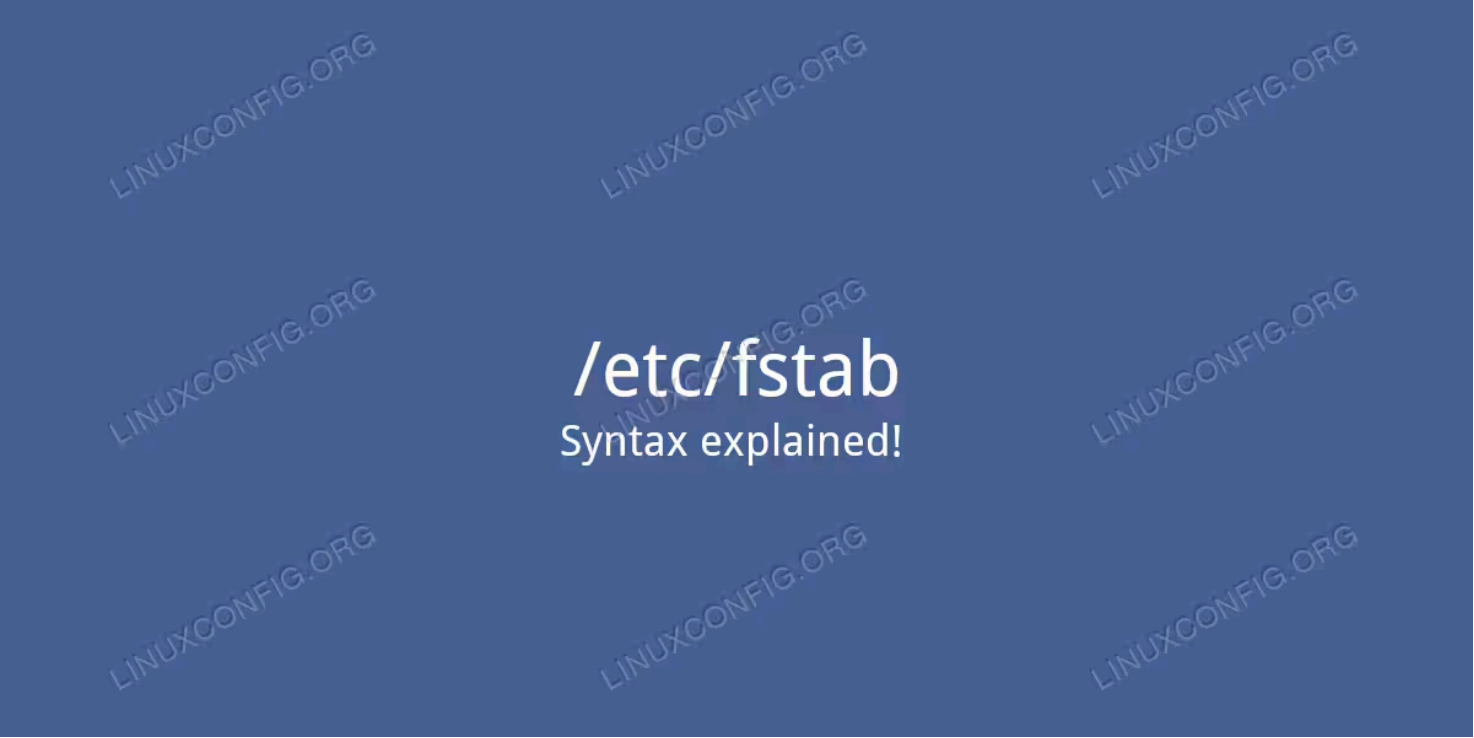
How fstab works - introduction to the /etc/fstab file on Linux

29 May 2020 by Egidio Docile

The `/etc/fstab` file is one of the most important files in a Linux-based system, since it stores static information about filesystems, their mountpoints and mount options. In this tutorial we will learn to know its structure in details, and the syntax we can use to specify each entry in the file.

In this tutorial you will learn:

- How to use the fstab file to provide static filesystem information
- How the fstab file is structured
- What is the purpose of each entry field in the file



/etc/fstab Syntax explained!

Software Requirements and Conventions Used

Software Requirements and Linux Command Line Conventions

Category	Requirements, Conventions or Software Version Used
System	Distribution-independent

Category	Requirements, Conventions or Software Version Used
Software	No specific software is needed to follow this tutorial
Other	Familiarity with basic concepts like ‘mountpoint’, and ‘filesystem’
Conventions	# – requires given linux commands to be executed with root privileges either directly as a root user or by use of <code>sudo</code> command \$ – requires given linux commands to be executed as a regular non-privileged user



The role of fstab

The first thing we must know about the `fstab` file is that it is meant to be only read by programs and never written except by the system administrator. Each line in the file describes a filesystem, and contains fields used to provide information about its mountpoint, the options which should be used when mounting it etc. Each field can be separated by another either by spaces or tabs. Let's analyze each field and its role in an entry.

Fstab fields

Each entry line in the fstab file contains six fields, each one of them describes a specific information about a filesystem.

First field - The block device

The first field in each fstab entry holds information about the local or remote block device which should be mounted. The most typical way to reference a block device is by using its node inside the `/dev` directory, so for example to reference the first partition of the `sda` block device we use `/dev/sda1` as value.

Alternative ways to reference a block device is by using its `LABEL` or `UUID` (Universal Unique Identifier). The latter is the absolutely preferred method, since it guarantees to univocally reference a filesystem, as its name states. On `GPT` partitioned disks it's also possible to reference a filesystem by using `PARTUUID` or `PARTLABEL`.

To obtain information about filesystems we can run the `lsblk` command, eventually with the `-o` option to specify the fields we want to retrieve, or by using the `-fs` one, which is the equivalent of using `-o` and provide `NAME,FSTYPE,LABEL,UUID,MOUNTPOINT` as arguments. By default the program will display information about all existing filesystems. To avoid this behavior, a filesystem reference must be passed as an argument:

```
$ lsblk -d -fs /dev/sdb1
NAME FSTYPE LABEL UUID                                FSAVAIL FSUSE%
MOUNTPOINT
```

```
sdb1 ext4      80b496fa-ce2d-4dcf-9afc-bcaa731a67f1    13.3G    1%  
/mnt/example
```

In the example above we used also the `-d` option for `lsblk`, short for `--nodeps`, to hide filesystems structure trees from the output. Now that we gathered information about a filesystem we can create an entry in fstab for it. In the first field of the entry, to reference the `/dev/sdb1` we will use its `UUID`:

```
UUID=80b496fa-ce2d-4dcf-9afc-bcaa731a67f1
```

Second field - The mountpoint

In each fstab entry, the second field specifies the `mountpoint` for the filesystem: what directory in the system should be used to access its content. This should be always provided except if the block device we are referencing it's used as swap. In that case `"none"` should be used. Suppose we want to mount our filesystem to `"/mnt/example"`; we would write:

```
UUID=80b496fa-ce2d-4dcf-9afc-bcaa731a67f1 /mnt/example
```

Third field - The filesystem type

The third field of an fstab entry specifies the type of filesystem in use on the raw block device or partition. The filesystem must be among the ones supported by the operating system like, for example ext4, xfs etc. In case of a remote filesystem we can use, for

example `cifs` as the value of this field if the filesystem is shared via samba or `nfs` if it is shared via the `Network File System`. In the case of our example, we know the `sdb1` device is formatted with the `ext4` filesystem, therefore our fstab entry becomes:

```
UUID=80b496fa-ce2d-4dcf-9afc-bcaa731a67f1 /mnt/example ext4
```

Fourth field - Mount options

The fourth field of each entry in the fstab file is used to provide a list of options to be used when mounting the filesystem. To use the default set of mount options we specify `default` as a value. Default options are:

- `rw` (read-write);
- `suid` (respect `setuid` and `setgid` bits);
- `dev` (interpret characters and block devices on the filesystem);
- `exec` (allow executing binaries and scripts);
- `auto` (mount the filesystem when the `-a` option of the mount command is used);
- `nouser` (make the filesystem not mountable by a standard user);
- `async` (perform I/O operations on the filesystem asynchronously).

To see the list of the available options we can consult the `mount` manual:

```
$ man mount
```

At this point, our entry becomes:

```
UUID=80b496fa-ce2d-4dcf-9afc-bcaa731a67f1 /mnt/example ext4 defaults
```

Fifth field - Should the filesystem be dumped ?

The fifth field in each entry can be either 0 or 1. The value is used by the dump backup program (if installed) to know what filesystem should be dumped. Typically our entry becomes:

```
UUID=80b496fa-ce2d-4dcf-9afc-bcaa731a67f1 /mnt/example ext4 defaults
```

Sixth field - Fsck order

The sixth field is used to establish the order by which another utility, `fsck`, should check filesystems on boot. The value of `1` must always be used for the root filesystem; for all the others we can use `2`. If this value is not provided it defaults to 0, and the filesystem will not be checked. With this last field our example entry is finally complete:

```
UUID=80b496fa-ce2d-4dcf-9afc-bcaa731a67f1 /mnt/example ext4 defaults 0
```

Conclusions

In this tutorial we learned how /etc/fstab, one of the most important files in a linux-based operating system, is structured. We learned that it contains static information about filesystems and we saw that each entry in the file is composed by six fields, each one with a specific purpose we examined.

Related Linux Tutorials:

- [Ultimate Web Server Benchmark: Apache, NGINX,...](#)
- [Best Linux Distro: How to Choose Guide for Every User](#)
- [Linux Configuration files: Top 30 most important](#)
- [An Introduction to Linux Automation, Tools and Techniques](#)
- [Mastering Bash Script Loops](#)
- [How to mount a Samba shared directory at boot](#)
- [How to set filesystems mount order on modern Linux...](#)
- [Assigning File Permissions to Specific Users with...](#)
- [Findmnt Command: Querying Filesystems in Linux Made Easy](#)
- [Access and modify virtual machines disk images with...](#)

📁 System Administration

🔑 administration, beginner, commands, filesystem

- < Check Linux Mint Version
 - > Persisting data into a PostgreSQL database with PHP
-
-

Comments and Discussions



5 replies

**Christopher_Stark**

June 2020

Can anybody help me? I'm desperately trying to transfer this mount line, which is successful in command line, into an fstab instruction that works. I've been trying for weeks and no joy...

**Christopher_Stark**

June 2020

```
sudo mount -t cifs //10.0.1.6 /5TBdrive /mnt/5TB -o user=libreelec,pass=libreelec, vers=3.0|
```

[1 reply](#)**sandmann** Moderator[► Christopher_Stark](#) June 2020

Hi Christopher_Stark,

Welcome to our forums.

How about something like this:

```
//10.0.1.6/5TBdrive /mnt/5TB cifs username=libreelec,password=libreelec,vers=3.0 0 0
```

This should do the trick, or at least provide some error message that may prove useful in debugging.

**sawkyrom**

September 2021

Excellent explanation of fstab! Thank you.

What would you suggest if the /etc/fstab file is an fstab.swp and displays the following error message in nano terminal when trying to edit?

```
[ Error reading lock file /etc/.fstab.swp: Not enough data read ]
```

[1 reply](#)



sawkyrom moderator

Hi Sawkyrom,

Welcome to our forums.

The file in question is a hidden temporary replacement of the original text file during editing. If the file persists and no other session is editing the original, it could be a leftover from a broken session. This doesn't mean you should not have the one with the original name in place. Maybe you deleted it by accident?

LinuxConfig



NEWSLETTER

Subscribe to Linux Career Newsletter to receive latest news, jobs, career advice and featured configuration tutorials.

[SUBSCRIBE](#)

TAGS

18.04

[administration](#)[apache](#) [applications](#) [backup](#)[bash](#) [beginner](#) [browser](#)[centos](#) [centos8](#)[commands](#) [database](#)[debian](#) [desktop](#)[development](#) [docker](#)[error](#) [fedora](#) [filesystem](#)[firewall](#) [gaming](#) [gnome](#)

FEATURED TUTORIALS

[How to install the NVIDIA drivers on Ubuntu 22.04](#)[How to mount USB drive in Linux](#)[How to install tar.gz file on Linux](#)[How to enable/disable wayland on Ubuntu Desktop](#)[Linux IP forwarding](#)

LATEST TUTORIALS

[Ultimate Web Server Benchmark: Apache, NGINX, LiteSpeed, OpenLiteSpeed, Caddy & Lighttpd Compared](#)[How to Install Elasticsearch on Ubuntu/Debian Linux](#)[How to Install and Run DeepSeek AI on Ubuntu/Debian \(No GPU\)](#)