**What is the file system in Linux?**

1)

So you have hard drive which can store data in 0 and 1. But you want to store and access file and folders right? For that you need proper structure to utilize your hard drive.

What file system does is it makes a table (well, kind of!) which stores data about files(and folders) like File name. When it was created. When it was modified. Who owns it. Who are allowed to access it. And lastly **Where are the data of file stored on hard drive.**

So when accessing the file. We go to table(which is also on hard drive only) and search for file. When we find it, we now know where it’s the data is located on hard drive. Something similar goes for folders too.

*So filesystems provides this table kind of structure, manage and store data on HDD and provide access when required.*

2)

File types or file *system* types? A folder is a folder, regardless of operating system, and as far as I know, the file type doesn’t matter. A .txt file is the same across all operating systems’ file systems, as is a .doc, .xcf, . jpeg, .gif, .bmp, .scs, .jar or any other number of file types. They can all be downloaded or created and written to a device regardless of the file *system* type.

A file *system* type is different. The file system type defines how a storage device is read from and written to, how much storage is available after the file system’s hierarchy is implemented, what defines a block or sector and other things. That’s why you’ll often see an operating system report that there is only 240 GB available on your 256 GB storage device.

Windows sticks you with NTFS, Apple with HFS. Linux is an operating system that will allow you to tailor it to your needs, and you choose which file system you want. Some of the available options are ext2, ext3, ext4, btrfs, reiserfs, and several more. Additionally, Linux has packages that make the various file explorers and other programs able to read and manipulate files from the other file systems, which is extremely handy when you’re working in a mixed-OS environment.

3 )

Linux, originally being a cone of Minix UNIX adheres to the "UNIX-philosophy", which includes: "Everything is a file". This means that everything, including hardware, is represented as a file in the Linux system. For example, the main drive of a Linux system may be represented by the file "/dev/hda0" and mounted as the directory "/". In the same fashion, your webcamera is also represented as a file which could be called (for example) "/dev/webcam0"

4 )

You are clearly new to linux, so this might come as a surprise to you. But unlike windows linux does’nt have a fixed file system. Windows only support the NTFS file system, while in linux world you have plenty of options:

1. **ext4/ext3**: The most stable and recommended, perfect for daily laptop, desktop usage. It is the default FS for most of the distros including the Ubuntu, go for this if you are a newbie and just want it to be up and running.
2. **XFS:**Focused on concurrency rather than integrity, mostly used in servers.
3. **ZFS:**This is a FS that is generally used in BSD systems and is equipped with many fancy technical features such as automatically detecting and adding space to the logical partition when you attach a new physical disk(takes care of formatting and everything for ya) and many more. This is also an advanced FS and relatively new to linux world, Canonical just added its support in Ubuntu 16.04.
4. **Brtfs:**Pronounced as butter-fs is an ambitious project trying to capture the best of everything and is currently in heavy development. It is said to be the next thing after ext file systems. But since it is in development i would not recommend it. Although keep an eye on it if you are interested.

These are the top of my head but there are many that cater to special needs.

5 )

Sorry. It’s a semi non-sensical question. Not your fault though, for it’s convention.

Linux has a root *mount point*. Not a root *file system*. Linux file system formats can be good choices of many, including EXT4, XFS, BTRFS, NFS, TEMPFS, GFS, etc etc etc.. (okay, maybe some *better* than others - but it’s a matter of *appropriateness*). But all the file systems get mounted onto the file system tree, eventually leading to the root (/) mount point.

Windows, mounts a c: and can mount a d: and an e: …etc etc — one drive letter per installed hard drive. Windows typically uses NTFS format for a file system, and the file system is normally assigned a drive letter for accessibility.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**/bin**

Commands needed during bootup that might be used by normal users (probably after bootup).

**/sbin**

Like /bin, but the commands are not intended for normal users, although they may use them if necessary and allowed. /sbin is not usually in the default path of normal users, but will be in root's default path.

**/etc**

Configuration files specific to the machine.

**/root**

The home directory for user root. This is usually not accessible to other users on the system

**/lib**

Shared libraries needed by the programs on the root filesystem.

**/lib/modules**

Loadable kernel modules, especially those that are needed to boot the system when recovering from disasters (e.g., network and filesystem drivers).

**/dev**

Device files. These are special files that help the user interface with the various devices on the system.

**/tmp**

Temporary files. As the name suggests, programs running often store temporary files in here.

**/boot**

Files used by the bootstrap loader, e.g., LILO or GRUB. Kernel images are often kept here instead of in the root directory. If there are many kernel images, the directory can easily grow rather big, and it might be better to keep it in a separate filesystem. Another reason would be to make sure the kernel images are within the first 1024 cylinders of an IDE disk. This 1024 cylinder limit is no longer true in most cases. With modern BIOSes and later versions of LILO (the LInux LOader) the 1024 cylinder limit can be passed with logical block addressing (LBA). See the **lilo** manual page for more details.

**/mnt**

Mount point for temporary mounts by the system administrator. Programs aren't supposed to mount on /mnt automatically. /mnt might be divided into subdirectories (e.g., /mnt/dosa might be the floppy drive using an MS-DOS filesystem, and /mnt/exta might be the same with an ext2 filesystem).

**/proc, /usr, /var, /home**

Mount points for the other filesystems. Although /proc does not reside on any disk in reality it is still mentioned here. See the section about /proc later in the chapter.