I understand what mounting is in Linux, and I understand device files. However I do not understand WHY we need to mount.

For example, as explained in the [accepted answer of this question](http://unix.stackexchange.com/questions/3192/what-is-meant-by-mounting-a-device-in-linux), using this command:

mount /dev/cdrom /media/cdrom

we are mounting the CDROM device to /media/cdrom and eventually are able to access the files of CDROM with the following command

ls /media/cdrom

which will list the content of the CDROM.

Why not skip mounting altogether, and do the following?

ls /dev/cdrom

And have the content of the CDROM Listed. I expect one of the answers to be: "This is how Linux is designed". But if so, then why was it designed that way? Why not access the /dev/cdrom directory directly? What's the real purpose of mounting?

You can read or write /dev/cdrom (eg, using dd or cat) but when you do that you are just reading or writing the raw bytes of the device. That can be useful in various circumstances (like cloning a partition), but generally we want to see the directories and files stored on the device.

When you mount a device you're basically telling the kernel to use a layer of software (the filesystem driver) to translate those raw bytes into an actual filesystem. Thus mounting a device associates the filesystem on that device to the directory hierarchy.

One reason is that block level access is a bit lower level than ls would be able to work with. /dev/cdrom, or dev/sda1 may be your CD ROM drive and partition 1 of your hard drive, respectively, but they aren't implementing ISO 9660 / ext4 - they're just RAW pointers to those devices known as [Device Files](https://en.wikipedia.org/wiki/Device_file).

One of the things mount determines is HOW to use that raw access - what file system logic / driver / kernel modules are going to manage the reads/writes, or translate ls /mnt/cdrom into which blocks need to be read, and how to interpret the content of those blocks into things like file.txt.

Other times, this low level access can be good enough; I've just read from and written to serial ports, usb devices, tty terminals, and other relatively simple devices. I would never try to manually read/write from /dev/sda1 to, say, edit a text file, because I'd basically have to reimplement ext4 logic, which may include, among other things: look up the file inodes, find the storage blocks, read the full block, make my change(s), write the full blocks, then update the inode (perhaps), or instead write this all to the journal - much too difficult.

One way to see this for yourself is just to try it:

[root@ArchHP dev]# cd /dev/sda1

bash: cd: /dev/sda1: Not a directory

/dev is a directory, and you can cd and ls all you like. /dev/sda1 is not a directory; it's a special type of file that is what the kernel offers up as a 'handle' to that device.

See [the wikipedia entry on Device Files](https://en.wikipedia.org/wiki/Device_file) for a more in depth treatment.