# 7 Under the hood

## 7.1 Introduction

MX Linux ultimately inherits its fundamental design from Unix, an operating system that has been around in various forms since 1970, much earlier than MS Windows. From that Linux was developed, from which Debian produces its distribution. The base operating system is the topic of this section. Users coming from MS Windows typically find a lot of unfamiliar concepts, and get frustrated trying to do things the way they are accustomed to doing them.

This section will give you a conceptual overview of some basic aspects of MX Linux, and how they differ from other systems to help ease your transition.

**Links**

* [Wikipedia: Unix](http://en.wikipedia.org/wiki/Unix)
* [Linux Home Page](http://www.linux.org/)
* [Wikipedia Debian](http://en.wikipedia.org/wiki/Debian)

## 7.2 The file system structure

There are two basic uses of the term “file system”.

* The first is the Operating System’s Filesystem. This refers to the files and their organization that the operating system uses to keep track of all the hardware and software resources it has as its disposal while running.
* The other use of the term file system refers to the Disk Filesystem, designed for the storage and retrieval of files on a data storage device, most commonly a disc drive. The Disk Filesystem is set when the disk partition is first formatted, prior to writing any data on the partition.

### The Operating System’s Filesystem

If you If you open Thunar and click on File System in the left pane, you will notice a number of directories with names based on the [Unix Filesystem Hierarchy Standard](https://refspecs.linuxfoundation.org/FHS_3.0/fhs/index.html).

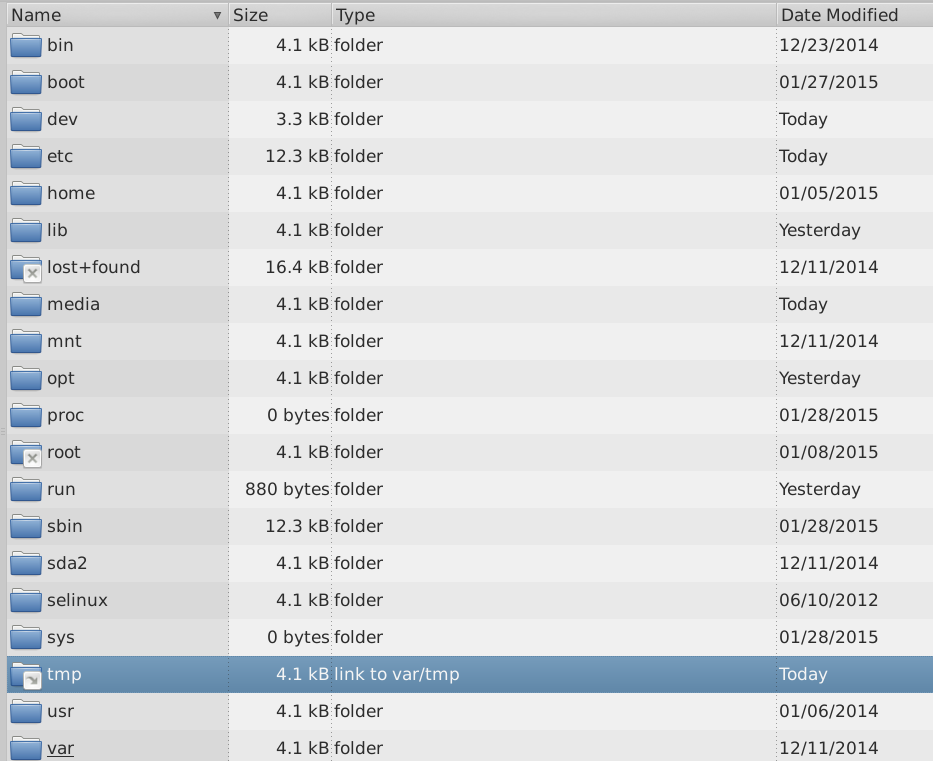


Figure 7-1: The MX filesystem viewed in Thunar

Here is a simple description of the major directories in MX Linux along with an example of when users commonly work with files in those directories:

* /bin
  + This directory contains binary program files which are used by the system during startup, but which also may be required by user actions once the system is fully up and running.
  + Example: Many basic command-line programs, such as the Bash shell, and utilities like /dd/, /grep/, /ls/, and /mount/ are located here, in addition to programs only used by the OS.
* /boot
  + As you might guess, files that Linux needs to boot are located here. The Linux kernel, the core of the Linux operating system, is kept here, as are bootloaders such as GRUB.
  + Example: no file here is commonly accessed by users.
* /dev
  + In this directory are special files that link to the various input/output devices on the system.
  + Example: no file here is commonly accessed directly by users, except in CLI mounting commands.
* /etc
  + This directory contains configuration files for the system as well as application configuration files.
  + Example: The file /etc/fstab specifies mount points for additional filesystems on devices, partitions, etc. that can be configured for your optimal use.
  + Example: display problems sometimes involve editing the file /etc/X11/xorg.conf.
* /home
  + Here the user’s personal directories (data and settings) reside. If there is more than one user, a separate subdirectory is set up for each. No user (except root) can read another user’s home directory. The user’s directory contains both hidden (where the filename is preceded by a dot) and visible files; hidden files can be revealed by clicking View >Show Hidden Files (or Ctrl-H) in Thunar.
  + Example: users typically organize their own files at first by using default directories such as Documents, Music, etc.
  + Example: a Firefox profile is located in the hidden directory .mozilla/firefox/
* /lib
  + This directory contains shared object libraries (analogous to Windows DLL’s) that are required at boot time. In particular, kernel modules will be found here, under /lib/modules.
  + Example: no file here is commonly accessed by users.
* /media
  + Files for removable media such as CDroms, floppy drives, and USB memory sticks are installed here when the media are automounted.
  + Example: After dynamically mounting a peripheral device like a flash drive, you may access it here.
* /mnt
  + Physical storage devices must be mounted here before they can be accessed. After drives or partitions are defined in the file /etc/fstab, then their file system is mounted here.
  + Example: Users can access drives and partitions mounted here.
* /opt
  + This is the intended location of major third-party application subsystems installed by the user.
  + Example: if you install Google Earth, this is where it will be installed. Some distros also place user-installed programs in /opt subdirectories.
* /proc
  + The location for process and system information
  + Example: no file here is commonly accessed by users
* /root
  + This is the home directory for the root user (administrator). Note that this is not the same as “/” the file system root.
  + Example: no file here is commonly accessed by users, but files saved while logged in as the root user may be saved here.
* /sbin
  + Programs are installed here if they are required by the system startup scripts but will not normally be run by users, other than root—in other words, system administration utilities.
  + Example: no file here is commonly accessed by users, but this is where files like modprobe and ifconfig are located.
* /tmp
  + This is the location of temporary files produced by programs—such as compilers—as they run. In general, these are short-term temporary files, of use to a program only while it is running.
  + Example: no file here is commonly accessed by users.
* /usr
  + This directory contains many files for user applications, and is analogous in some ways to the Windows directory “Program Files”.
  + Example: many executables are located in (/usr/bin)
  + Example: documentation (/usr/docs) and configuration files, graphics and icons are in (/usr/share).
* /var
  + This directory contains files that are constantly changing while Linux is running, e.g. logs, system mail and queued processes.
  + Example: you can look in /var/log/ when trying to determine what happened during a process such as installing a package.

### The Disk Filesystem

The disk file system is something about which the average user does not need to be much concerned. The default disk file system used by MX Linux is called ext4, a version of the ext2 file system that is journaled —i.e., it writes changes to a log before enacting them, rendering it more robust. The file system ext4 is set during installation when your hard drive is formatted.

By and large, ext4 has more years on its track record than any of its rivals, and combines stability and speed; for these reasons, we do not recommend installing MX Linux onto a different disk file system unless you are well-educated in the differences. However, MX Linux can read and write to many other formatted disk filesystems, and may even be installed on some of them, if for some reason one of them is preferred over ext4.

**Links**

* [Wikipedia Filesystem](http://en.wikipedia.org/wiki/File_system)
* [Wikipedia. Comparison of filesystems](http://en.wikipedia.org/wiki/File_system)
* [Wikipedia Ext4](http://en.wikipedia.org/wiki/Ext4)

## 7.3 Permissions

MX Linux is an account-based operating system. This means that no program can run without a user account to run under, and any running program is thereby limited by the permissions granted to the user who started it.

NOTE: Much of the security and stability that Linux is known for hinges on the proper use of limited user accounts, and the protection provided by default file and directory permissions. For this reason, you should operate as root only for a procedure that requires it. Never log into MX Linux as root to run the computer for normal activities–running a web browser as root user, for instance, is one of the few ways you could get a virus on a Linux system!

### Basic information

The default file permissions structure in Linux is fairly simple, but more than adequate for most situations. For each file or folder, there are three permissions that can be granted, and three entities (owner/creator, group, others/world) to which they are granted. The permissions are:

* Read permission means that data can be read from the file; it also means the file can be copied. If you don’t have read permission for a directory you can’t even see the names of files listed in it.
* Write permission means that the file or folder can be changed, appended, or deleted. For directories, it specifies whether a user can write to files in the directory.
* Execute permission means whether or not the user can run the file as a script or program. For directories, it determines whether or not the user can enter and make it the current working directory. Every file and folder acquires a single user designated as its owner when it is created on the system. (Note that if you move a file from another partition where it has a different owner, it will keep the original owner; but if you copy and paste it, it will be assigned to you.) It also has a single group designated as its group, by default the group to which the owner belongs. The permissions you grant to others affect everyone who isn’t the owner or in the owning group.

NOTE: For advanced users, there are additional special attributes beyond read/write/execute that can be set: sticky bit, SUID, and SGID. For more information, see Links section below.

#### Viewing, setting and changing permissions

There are many tools available in MX Linux to view and manage permissions.

* GUI
  + Thunar. To view or change a file’s permissions, right-click the file and select Properties. Click the Permissions tab. Here you can set the permissions granted to the owner, group, and others using the pull-down menus. For some files (like scripts , for instance), you need to check the box to make them executable, and for folders you can check a box to limit the deletion of files inside it to the owners.
  + NOTE: you must be operating as root to change the permission of a file or directory whose owner is root. On bigger folders you MUST refresh your Thunar window or else the permissions will show incorrectly, even though the permissions have actually changed. Just hit F5 to refresh the window or else you will see the original permissions.
  + MX User Manager is an easy way to change permissions by associating a user with specific groups.
* CLI
  + Internal partitions. By default, the root/superuser password is required to mount internal partitions. To change this behavior, click on MX Tweak, Other tab.
  + New external partitions. Formatting a new partition with ext4 requires root permissions, which can lead to unexpected or undesired result of the regular user not being able to write any files to the partition. To change this behavior, consult [the MX/antiX Wiki](https://mxlinux.org/wiki/system/format-ext4-filesystem-be-owned-regular-user).
  + Manual operations. Although MX User Manager covers most daily situations, sometimes it can be preferable to deal with the command line. Basic permissions are represented by r (read), w (write) and x (execute); a dash indicates no permissions. To view permissions of a file on the command line, type this: *ls -l NameofFile*. You may need to use the full location of the file (e.g., /usr/bin/gimp). The -l switch will cause the file to be listed in long format, displaying its permissions among other information.

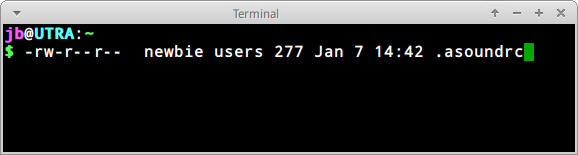


Figure 7-2: Viewing file permissions

The characters right after the opening dash (indicating it is a regular file) contain the three permissions (read/write/execute) for owner, group and others: 9 characters total. Here it shows that the owner has read and write but not execute (rw-), but the group and others can only read. The owner in this case is specified to be “newbie” who belongs to the group “users”.

If for some reason it was necessary to change the ownership of this file to root using the command line, you would use the chown command like in this example:

*chown root /home/newbie/.asoundrc*

For details on using chown, as well as the more detailed chmod, see Links section.

**Links**

* [MX/antiX Wiki: Permissions](https://mxlinux.org/wiki/system/permissions)
* [File Permissions](http://www.linux.com/learn/tutorials/309527-understanding-linux-file-permissions)

## 7.4 Configuration files

### 7.4.1 User config files

Files that hold individual user settings (such as high scores for your games, or the layout of your desktop) are stored within a Users home directory, typically as a hidden file or directory, and can only be edited by that user or by root. These personal configuration files are actually less often edited directly than system files because most of the user configuration is done graphically through the applications themselves. When you open an application and click Edit > Preferences, for example, your selections are written to a (usually hidden) configuration file in your user directory. Likewise in Firefox, when you type about:config in the address bar, you are editing the hidden configuration files. The Xfce configuration files are stored in ~/.config/.

### 7.4.2 System config files

Files that hold system-wide configurations or defaults (such as the file that determines which services automatically launch during boot up) are largely stored in the /etc/ directory and are only editable by root. Most of these files are never touched directly by regular users, such as these for instance:

* */etc/rc.d/rc5.d* — Contains files to control runlevel 5 into which MX Linux boots after login.
* */etc/sysconfig/keyboard* — Used to configure the keyboard.
* */etc/network/interfaces* — Defines internet interfaces on the system.

Some configuration files can contain just a few lines, or even be empty, while others may be quite long. The important point is that if you are looking for a configuration file for an application or process, head for the /etc directory and look around. Caution: because these files affect the whole system, 1) back up any file you intend to edit (easiest in Thunar: copy and paste back in, adding BAK at the end of the file name), and 2) be very careful!

### 7.4.3 Example

Sound problems can be solved with a number of graphical and command-line tools, but once in a while a user needs to edit directly the system-wide configuration file. For many systems, this will be */etc/modprobe.d/snd-hda-intel.conf*. It is a simple file whose top paragraph looks like this:

*# some chips require that the model be set manually  
 # for example asus g71 series may need model=g71v*

*options snd-hda-intel model=auto*

To try to get sound, you might decide to substitute the exact information about the sound model in place of the word “auto”. To find out your sound model, you could open a terminal and type:

*lspci | grep Audio*

The output will depend on the system, but it will take the following form:

*00:05.0 Audio device: nVidia Corporation MCP61 High Definition Audio (rev a2)*

Now you can plug that information back into the configuration file:

# some chips require that the model be set manually  
 # for example asus g71 series may need model=g71v  
 options snd-hda-intel model=nvidia

You would save the file, reboot the machine, and hopefully your sound should be working. You could also try more precision by using *model=nvidia mcp61* instead, if the first did not work.

Links

* [Understanding Linux Configuration Files](https://mxlinux.org/wiki/system/permissions)
* [File Permissions](http://www.linux.com/learn/tutorials/309527-understanding-linux-file-permissions)

## 7.5 Runlevels

MX Linux boots up by default using sysVinit. (A different method, systemd, is also present as an option; it does not use runlevels in the same manner.) After completing the boot process, init executes all startup scripts in a directory specified by the default runlevel (this runlevel is given by the entry for id in /etc/inittab). Like most other Linux versions, MX Linux has 7 runlevels:

Table 10: Runlevels in MX Linux

|  |  |
| --- | --- |
| **Runlevel** | **Comment** |
| 0 | Halt the system |
| 1 | Single-user mode: provides a root console without logon.  Useful if you lose your root password |
| 2 | Multiuser with no network |
| 3 | Console logon, no X (i.e. no GUI) |
| 4 | Not used/custom |
| 5 | Default GUI logon |
| 6 | Reboot the system |

MX Linux defaults to runlevel 5, therefore any init scripts set up in the level 5 config file will run at boot.

#### Use

Understanding runlevels can be handy. When users have a problem with X Window Manager, for instance, they can not correct it on the default runlevel 5, because X is running on that level. But they can get to runlevel 3 to work on the problem in one of two ways.

* From the Desktop: press Ctrl-Alt-F1 to get out of X. To actually drop to runlevel 3, become root and type *telinit 3*; this will stop all the other services still operating on runlevel 5.
* From the GRUB menu: press **e** (for edit) when you see the GRUB screen. On the subsequent screen, add a space and the number 3 at the end of the line (by default where the word „quiet” is) that starts with “linux” located one above the lowest line (the actual boot command). Press F-10 to boot.

Once the cursor is at a prompt, login with your normal username and password. If necessary, you can also login as “root” and provide the administrative password. Useful commands when you are looking at the prompt on runlevel 3 include:

Table 11: Common runlevel 3 commands

|  |  |
| --- | --- |
| **Command** | **Comment** |
| runlevel | Returns the number of the runlevel you are on. |
| halt | Run as root. Shuts the machine down. If that does not work on your system, try poweroff. |
| reboot | Run as root. Reboots the machine. |
| <application> | Runs the application, as long as it is not graphical. For instance, you can use the command nano to edit text files, but not leafpad. |
| Ctrl-Alt-F7 | If you used Ctrl-Alt-F1 to drop out from a running desktop but did not continue down to runlevel 3, this command brings you back to your desktop. |
| telinit 5 | Run as root. If you are on runlevel 3, enter this command to get to the login manager lightdm. |

NOTE: these commands may change in the future if MX Linux switches to a new system manager.

**Links**

* [Wikipedia: Runlevel](http://en.wikipedia.org/wiki/Runlevel)
* [The Linux Information Project: Runlevel Definition](http://www.linfo.org/runlevel_def.html)

## 7.6 The kernel

### 7.6.1 **Introduction**

This Section covers common user-centered interactions with the kernel. Consult the Links for other, more technical aspects

### 7.6.2 Upgrading/Downgrading

#### Basic

Unlike other software on your system, the kernel is not upgraded automatically except below the minor revision level (indicated by the third number in the kernel name). Before you change your current kernel, you would do well to ask yourself some questions:

* Why do I want to upgrade the kernel? Is there a driver I need for new hardware, for instance?
* Should I downgrade the kernel? For example, Core2 Duo processors tend to have odd issues with the default MX-Linux kernel that are solved by switching to the default Debian 4.9 kernel (using MX Package Installer).
* Am I aware that unnecessary changes might bring problems of one kind or another?

MX Linux provides an easy method of upgrading/downgrading the default kernel: open MX Package Installer and click on the “Kernel” category. There you will see a number of kernels that are available to the user. Select the one you want to use (ask on the Forum if unsure) and install it.

Once you check and install the new kernel, reboot and make sure the new kernel is highlighted; if not, click on the options line and select what you want.

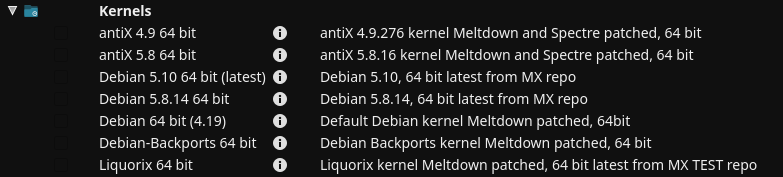


Figure 7-3: Kernel options in MX Package Installer for 64bit architecture

#### Advanced

Here is a basic approach for manually upgrading the Linux kernel on your system.

* First, find out what you currently have installed. Open a terminal and enter *inxi S*. For instance, a user of MX-19 64 bit version might see something like this:

Kernel: 5.8.0-2-amd64 x86\_64 bits

Be sure to write down the name of the kernel from the output of that command.

* Second, select and install a new kernel. Open Synaptic, search on linux-image and look for a higher kernel number that matches the architecture (e.g., 686) and processor (e.g., PAE) that you already have, unless you have a good reason to change. Install the one you want or need in the usual manner.
* Third, install the linux-headers package that match the new kernel you selected. There are two methods of doing this.
  + Look carefully at the Synaptic entries beginning linux-headers and match the kernel.
  + Alternatively, you can install the headers more easily after rebooting into the new kernel by typing the following code in a root terminal:

*apt-get install linux-headers-$(uname -r)*

Headers will also be installed if you use a command such as *m-a prepare*.

* When you reboot, you should automatically boot into the highest available kernel. If it doesn’t work, you have the option to return to what you were using: reboot, and when you see the GRUB screen highlight Advanced Options for whatever partition you want to boot into, then select the kernel and press Enter.

### 7.6.3 Kernel upgrade and drivers

[Dynamic Kernel Module Support (DKMS)](https://en.wikipedia.org/wiki/Dynamic_Kernel_Module_Support) automatically recompiles all DKMS driver modules when a new kernel version is installed. This allows drivers and devices outside of the mainline kernel to continue working after a Linux kernel upgrade. The exception concerns proprietary graphics drivers (Section 3.3.2).

* NVidia drivers
  + If installed with sgfxi, they must be rebuilt with sgfxi, see Section 6.5.3
  + If installed with the MX Nvidia driver installer or via synaptic/apt-get, the kernel modules may need to be rebuilt. Re-running MX Nvidia driver installer from the menu should offer to reinstall and rebuild the modules. If your reboot gets stuck at a console prompt, become root and enter "*ddm-mx -i nvidia*" to reinstall and rebuild the driver modules.
* Intel drivers
  + You may need to upgrade the driver, depending on the kernel you select for upgrade target.

### 7.6.4 More options

Other considerations and choices exist with respect to kernels:

* Other pre-rolled kernels exist such as the Liquorix kernel, which is a version of the Zen kernel and is intended to provide a better desktop use experience in terms of responsiveness, even under heavy loads such as during gaming, plus low latency (important for audio work). MX Linux updates the Liquorix kernels frequently, so it is most easily installed through the MX Package Installer, in the Kernel section.
* Distros (e.g., MX’s sister distro antiX) often roll their own.
* Knowledgeable individuals may compile a specific kernel for particular hardware.

### 7.6.5 Links

* [Wikipedia: Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel)
* [Anatomy of the Linux kernel](http://www.ibm.com/developerworks/library/l-linux-kernel)
* [Linux kernel archives](http://www.kernel.org/)
* [Interactive map of Linux kernel](http://www.makelinux.net/kernel_map)

### 7.6.6 Kernel panic and recovery

A kernel panic is a relatively rare action taken by the MX Linux system when it detects an internal fatal error from which it can not safely recover. It can be caused by a number of different factors that range from hardware problems to a bug in the system itself. When you get a kernel panic, try rebooting with the MX Linux LiveMedium, which will overcome temporarily any software problems and hopefully allow you to see and offload your data. If that doesn’t work, then unplug all unnecessary hardware and try again.

Your first concern is to access and secure your data. Hopefully, you have it backed up somewhere. If not, you can use one of the data recovery programs such **ddrescue** that is supplied with MX Linux. Your last resort is to take your hard drive to a professional recovery business.

There are a number of steps you might have to take to recover a functional MX Linux system once you have your data safe, although ultimately you may have to reinstall using the LiveMedium. Depending on the type of failure, the following steps may be undertaken:

1. Remove packages that broke the system.
2. Reinstall the graphic driver.
3. Reinstall GRUB using MX Boot Repair.
4. Reset the root password.
5. Reinstall MX Linux, selecting the check box on Screen 2 to keep /home so that your personal configurations will not be lost.

Be sure to ask on the Forum if you have any questions about these procedures.

**Links**

* [GNU C Library Home Page](http://www.gnu.org/software/libc)
* [Ddrescue](http://www.forensicswiki.org/wiki/Ddrescue)

## 7.7 Our positions

### **7**.7.1 Systemd

MX Linux ships with two [init](https://en.wikipedia.org/wiki/Init) systems: SysVinit (default) and [systemd](https://en.wikipedia.org/wiki/Systemd). Because the use of the ”systemd” as a system and service manager has been controversial, we want to be clear about its function in MX Linux: **systemd is included but not enabled.** You can scan your MX system and discover files bearing *systemd\** names, but those simply provide a compatibility hook/entrypoint when needed.

MX Linux uses *systemd-shim*, which emulates the systemd functions that are required to run the helpers without actually using the service. This means that SysVinit remains the default init yet MX Linux can use Debian packages that have systemd dependencies such as CUPS and Network Manager. This approach also allows the user to retain the ability to choose his/her preferred init at boot by selecting the systemd entry.

### **7**.7.2 Non-free software

MX Linux is fundamentally user-oriented, so includes a certain amount of [non-free software](https://www.gnu.org/philosophy/categories.en.html) to assure that the system works out of the box as much as possible. The user can see a list by opening a [console or terminal](https://fossbytes.com/difference-between-shell-console-terminal/) and typing:

*vrms*

Examples:

* The “wl” driver (broadcom-sta) and non-free firmware with proprietary components.
* A dedicated tool for installing Nvidia graphic drivers.
* Adobe Flash Player (distributed by permission).

Our rationale: it is much easier for advanced users to remove these drivers than it is for regular users to install them. And it's particularly difficult to install a driver for a network card without Internet access!

# 8 Glossary

Linux terms can be confusing and offputting at first, so this Glossary provides a list of the ones used in this Manual to get you started.

* **applet**: A program designed to be executed from within another application. Unlike an application, applets can not be executed directly from the operating system.
* **backend**: Also /back-end./ The backend includes the various components of a program that process the user input entered through the frontend. See also frontend.
* **backport**: Backports are new packages that have been recompiled to run on a released distribution in order to keep it up-to-date.
* **BASH**: The default shell (command-line interpreter) on most Linux systems as well as on Mac OS X, BASH is an acronym for Bourne-again shell.
* **BitTorrent**: Also/bit torrent/ or /torrent./ A method invented by Bram Cohen to distribute large files without the need for a single individual to provide the hardware, hosting and bandwidth resources required.
* **boot block**: An area of a disk outside the MBR having information for loading the operating system that is needed to start a computer.
* **bootloader**: Program that initially chooses an operating system to load after the BIOS has finished initializing the hardware. Extremely small in size. the bootloader’s only job is to hand control of the computer over to the Operating System’s kernel. Advanced bootloaders offer a menu to choose between several installed operating systems.
* **chainloading**: Also /chain loading./ Instead of directly loading an operating system, a boot manager like GRUB can use chain loading to pass control from itself to a boot sector on a hard disk partition. The target boot sector is loaded in from disk (replacing the boot sector from which the boot manager itself was loaded) and the new boot program is executed. In addition to when it is necessary, as in booting Windows from GRUB, the advantage to chainloading is that each operating system on the hard disk drive —and there could be dozens— can be responsible for having the correct data in it’s own boot sector. So GRUB residing in the MBR need not be rewritten every time there are any changes. GRUB can simply chainload the relevant information from the boot sector of a given partition whether it has changed or remained the same since the last boot time.
* **cheat code**: Codes can be entered when booting a LiveMedium to change the booting behavior. They are used to pass options to the MX Linux operating system to set parameters for particular environments.
* **command line interface (CLI)**: Also known as console, terminal, command prompt, shell, or bash. This is a UNIX-style text interface, which MS-DOS was also designed to resemble. A root console is one where administrative privileges have been acquired after entering the root password.
* **desktop environment**:The software which provides a graphical desktop (windows, icons, desktop, task bar, etc) for an operating system user.
* **disk image**: A file containing the complete contents and structure of a data storage medium or device such as a hard drive or DVD. See also ISO.
* **Distribution**: A Linux Distribution, or **distro**, is a particular packaging of the Linux kernel with various GNU software packages, and different desktops or window managers. Since—unlike the proprietary code used in the Microsoft and Apple OS’s—GNU/Linux is Free, Open-Source Software, literally anyone in the world who has the ability can freely build on what has been done and innovate a new vision of a GNU/Linux operating system. MX Linux is a distro based on the Debian Linux family.
* **file system**: Also file system. This refers to the way that files and folders are logically arranged on a computer’s storage devices so they may be found by the operating system. It can also refer to the type of formatting on a storage device, such as the common Windows formats NTFS and FAT32, or the Linux formats ext3, ext4 or ReiserFS, and in this sense refers to the method actually used to encode binary data on the Hard Disk Drive, floppy, flash drive, etc.
* **firmware**. The small programs and data structures that internally control the electronic components
* **free-as-in-speech**: The English word “free” has two possible meanings: 1) without cost, and 2) without restrictions. In part of the open-source software community, an analogy used to explain the difference is 1) “free” as in beer vs. 2) “free” as in speech. The word /freeware/ is used universally to refer to software that is simply without cost, whereas the phrase /free software/ loosely refers to software that is more properly called open-source software, licensed under some type of open source license.
* **frontend**: Also front-end. The frontend is the part of a software system that interacts directly with the user. See also backend.
* **GPL**: The GNU General Public License. This is a license under which many open-source applications are released. It specifies that you may view, modify, and redistribute the source code of applications released under it, within certain limits; but that you may not distribute the executable code unless you also distribute the source code to anyone who asks for it.
* **GPT:** A partitioning scheme used by native UEFI
* **Graphical User Interface (GUI)**: This refers to a program or operating system interface that uses pictures (icons, windows, etc), as opposed to text (command-line) interfaces.
* **home directory**: One of the 17 top-level directories branching from the root directory in MX Linux, /home contains a subdirectory for every registered user of the system. Within each Users home directory s/he has full read-write privileges. Further, most of the user-specific configuration files for various installed programs are stored in hidden subdirectories within the /home/username/ directory—as is downloaded email. Other downloaded files usually go by default into the home/username/Documents or /home/username/Desktop subdirectories.
* **IMAP**: The Internet Message Access Protocol is a protocol that allows an e-mail client to access a remote mail server. It supports both on-line and off-line modes of operation.
* **interface**: A point of interaction between computer components, often referring to the link between a computer and a network. Examples of interface names in MX Linux include **WLAN** (wireless) and **eth0** (basic wired).
* **IRC**: Internet Relay Chat, an older protocol to render the exchange of text messages easier.
* **ISO**: A disc image following an international standard that contains data files and file system metadata, including boot code, structures, and attributes. This is the normal method for delivering Linux versions such as MX Linux over the Internet. See also **disk image**.
* **kernel**: The layer of software in an operating system that interacts directly with the hardware.
* **LiveCD/DVD**: A bootable compact disc from which one can run an operating system, usually with a complete desktop environment, applications, and essential hardware functionality.
* **LiveMedium**: a general term that includes both LiveCD/DVD and LiveUSB.
* **LiveUSB**: A USB flash drive on which an operating system has been loaded in such a way that it can be booted and run. See LiveDVD.
* **mac address**: a hardware address that uniquely identifies each node (connection point) of a network. It is formed of a string of usually six sets of two-digits or characters, separated by colons.
* **man page**: Short for **manual**, man pages typically contain detailed information about switches, arguments, and sometimes the inner workings of a command. Even GUI programs often have man pages, detailing available command line options. Available in Start menu by typing a # before the name of the man page you want into the Search box, for example: *#pulseaudio*.
* **MBR**: Master Boot Record: the first 512-byte sector of a bootable hard disk drive. Special data written to the MBR enables the computer’s BIOS to pass the boot process off to a partition with an installed operating system.
* **md5sum**: A program that calculates and verifies a file’s data integrity. The MD5 hash (or checksum) functions as a compact digital fingerprint of a file. It is extremely unlikely that any two non-identical files will have the same MD5 hash. Because almost any change to a file will cause its MD5 hash to also change, the MD5 hash is commonly used to verify the integrity of files.
* **mirror**: Also mirror site. An exact copy of another Internet site, commonly used to provide multiple sources of the same information to supply reliable access to large downloads.
* **module**: Modules are pieces of code that can be loaded and unloaded into the kernel upon demand. They extend the functionality of the kernel without the need to reboot the system.
* **mountpoint**: The place on the root file system where a fixed or removable device is attached (mounted) and accessible as a subdirectory. All computer hardware needs to have a mountpoint in the file system to be usable. Most standard devices such as keyboard, monitor and your primary hard disk drive are mounted automatically at boot.
* **mtp**: MTP stands for Media Transfer Protocol and operates at the file level so that your device doesn’t expose its entire storage device. Older Android devices used USB mass storage for transferring files back and forth with a computer.
* **NTFS®**: Microsoft’s New Technology File System debuted in 1993 on the Windows NT Operating System, geared to business networks, and with revisions entered the mainstream Windows user’s desktop computers in later versions of Windows 2000. It has been the standard file system since Windows XP was introduced in late 2001. Unix/Linux-oriented folk say it stands for “Nice Try File System”!
* **open-source**: Software whose source code has been made available to the public under a license that allows individuals to modify and redistribute the source code. In some cases, open-source licenses restrict the distribution of binary executable code.
* **package**: A package is a discrete, non-executable bundle of data that includes instructions for your package manager about installation. A package doesn’t always contain a single application; it might contain only part of a large application, several small utilities, font data, graphics, or help files.
* **package manager**: A package manager such as (Synaptic or Gdebi) is a collection of tools to automate the process of installing, upgrading, configuring, and removing software packages.
* **Panel**: The highly configurable panel in Xfce4 appears by default at the left side of the screen and contains navigation icons, open programs and system notifications.
* **Partition Table**: A partition table is a hard disk architecture that expands on the older Master Boot Record (MBR) partitioning scheme using globally unique identifiers (GUID) to enable the existence of more than the original four partitions.
* **persistence**: the ability when running a LiveUSB to retain changes made during a live session.
* **port**: A virtual data connection that can be used by programs to exchange data directly, instead of going through a file or other temporary storage location. Ports have numbers assigned for specific protocols and applications, such as 80 for HTTP, 5190 for AIM, etc.
* **purge**: A command that removes not only the package named, but also any configuration and data files associated with it (though not those in a user’s home directory).
* **repo**: A shortened form of repository.
* **repository**: A software repository is an internet storage location from which software packages may be retrieved and installed via a package manager.
* **root**: Root has two common meanings in a UNIX/Linux OS; they are intimately connected, but the distinction is important to understand.
  + The **root file system** is the basic logical structure of all the files the operating system can access, whether programs, processes, pipes or data. It should follow the Unix Filesystem Hierarchy Standard, which specifies where in the hierarchy to locate all types of files.
  + The **root user** who owns the root file system —and so has all permissions necessary to do anything to any file. While it is sometimes necessary to temporarily assume the powers of the **/root user/** to install or configure programs, it is dangerous and violates the basic security structure of Unix/Linux to log in and operate as /root/ unless absolutely necessary. In a command line interface, a regular user can temporarily become root by issuing the command **su** then entering the root password.
* **runlevel**: A runlevel is a preset operating state on a Unix-like operating system. A system can be booted into any of several runlevels, each of which is represented by a single digit integer. Each runlevel designates a different system configuration and allows access to a different combination of processes (i.e., instances of executing programs). See Section 7.5.
* **script**: An executable text file, containing commands in an interpreted language. Usually refers to BASH scripts which are used extensively “under the hood” of the Linux operating system, but other languages may be used as well.
* **session**: A login session is the period of activity between a user logging in and logging out of a system. In MX Linux, this typically indicates the lifetime of a particular user “process” (the program code and its current activity) that Xfce invokes.
* **SSD**: A solid-state drive (SSD) is a nonvolatile storage device that stores persistent data on solid-state flash memory.
* **source code**: The human-readable code in which software is written prior to being assembled or compiled into machine-language code.
* **switch**: A switch (also /flag/, /option/ or /parameter/) is a modifier appended to a command to change its behavior. A common example is **-R** (recursive), which tells the computer to carry out the command through all subdirectories.
* **symlink**: Also symbolic link and soft link. A special type of file that points to another file or directory and not to data. It allows the same file to have different names and/or locations.
* **tarball**: An archiving format, like zip, popular on the Linux platform. Unlike zip files, though, tarballs may use one of a number of different compression formats, such as gzip or bzip2. They usually end in file extensions like .tgz, .tar.gz, or .tar.bz2.

Many archive formats are supported in MX with a graphical application called Archive Manager. Usually an archive can be extracted simply by right-clicking on it in Thunar.

* **(U)EFI:** Unified Extensible Firmware Interface is a kind of system firmware used on recent machines. It defines a software interface between an operating system and platform firmware, and represents the successor of old BIOS.
* **Unix**: Also UNIX. The operating system which Linux is modeled after, developed in the late 1960’s at Bell Labs and used primarily for servers and mainframes. Like Linux, Unix has many variations.
* **UUID (Universally Unique IDentifier)**. A universally unique identifier (UUID) is a 128-bit number that identifies unique Internet objects or data.
* **window manager**: A component of a desktop environment that provides the basic maximize/minimize/close/move functions for windows in the GUI environment. Sometimes it can be used as an alternative to a full desktop environment. In MX Linux, the default window manager is Xfce4.
* **X**: Also X11, xorg. The X Window System is a networking and display protocol which provides windowing on bitmap displays. It provides the standard toolkit and protocol to build graphical user interfaces (GUIs) on Unix-like operating systems and OpenVMS, and is supported by almost all other modern operating systems.