Linux Tutorial | Everything you need to know about Linux

By Great Learning Team / Updated on Feb 24, 2022 / 4936

Table of contents

Introduction to Linux

Let's get started with the Linux Tutorial! It is a free, open-source, and community-developed operating system. It was created as a hobby by Linus Torvalds in 1991. Linus, while at university, wanted to create an alternative, free, open-source version of the MINIX operating system that is based on the principles and design of Unix. This OS became the most-used OS on publicly available internet servers and the only OS used on the top 500 fastest supercomputers. The source code of Linux can be modified and distributed commercially or non-commercially to anyone under the GNU General Public License.

Moving ahead in this Linux Tutorial. It was mainly created for personal computers and later on used in other machines like mainframe computers, supercomputers, servers, etc. Linux is also used in embedded systems like automation controls, routers, televisions, digital video recorders, video game consoles, etc. Linux has been designed mainly for the (CLI) command-line interface, but we can also use a desktop environment for graphical experience using ***GNOME*** or ***KDE Desktop*** environment. Linux can run on all major UNIX software tools, applications, and network protocols. It supports 32-bit /64-bit hardware. Linux also has Unix network-centric design ideas and is a multi-user network operating system. The system users have limited rights to use the system, and these rights given by the System administrator who has access to the root account.

A **Linux distro or distribution** is a version of an open-source OS. It is packaged with various other components like installation programs, management tools, and additional software like KVM hypervisor. Red Hat Enterprise Linux from Red Hat is the most popular Linux distribution. RHEL is developed explicitly for the business market. Google's Android OS is based on Linux.

Linux distro or distributions also come in all sizes and shapes. Many, if not all distributions, offer the ability to run directly via DVD/CD in what's known as a 'Live CD' environment or even via USB if the motherboard supports booting from USB.

The three popular Desktop distributions of Linux are:

1. Ubuntu
2. Linux Mint
3. Fedora

The four popular Server versions of Linux are:

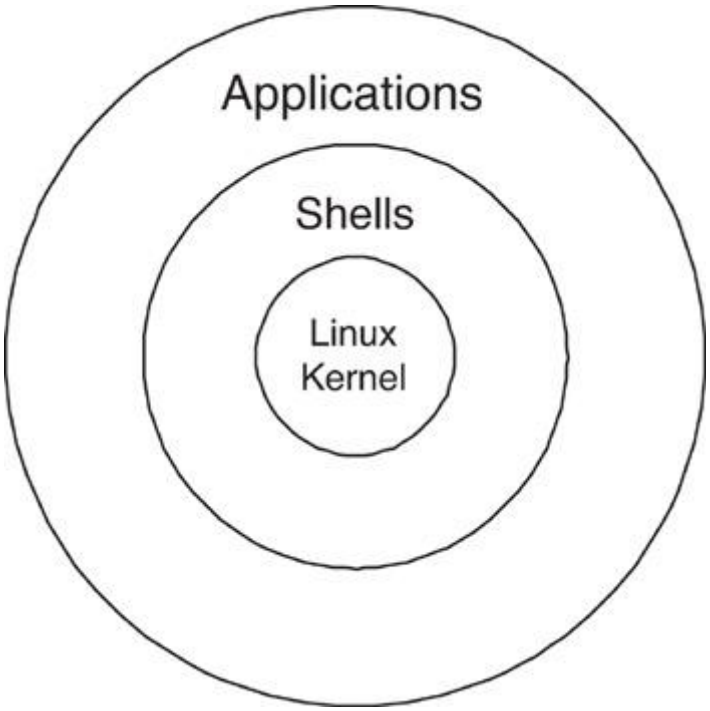
1. Slackware
2. CentOS
3. Debian
4. OpenSUSE

The two most popular Virtual Server versions of Linux are:

1. Citrix XenServer
2. VMWare
3.

Linux Architecture

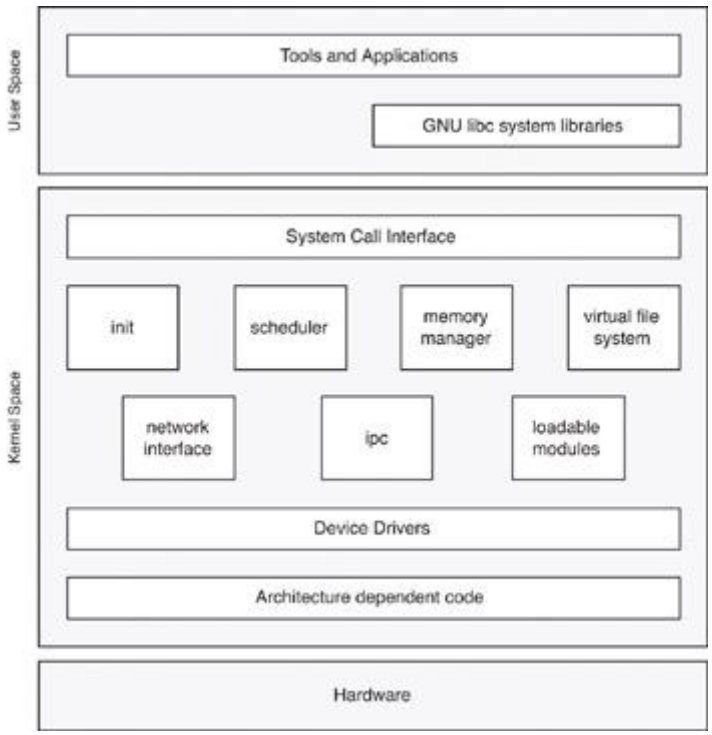
The Linux OS has primarily three components:



- **Kernel**: On the core is the Linux kernel that mediates access to the underlying hardware resources such as the CPU via the scheduler, memory, and peripherals.
- **Shell**: Shell (Ex: Bash, cShell, etc.) provides user access to the kernel, command interpretation and the means to load user applications and execute them.
- **Applications**: Applications make up the bulk of the GNU/Linux operating system. These applications provide useful functions for the OS, such as web browsers, windowing systems, language interpreters, email programs, and programming and development tools.

Kernel Components

It mediates access to the system resources, i.e., interfaces, the CPU, etc. It also enforces the security of the system and protects users from one another. The kernel is made up of the below major components:



- **init**

The ***init*** is performed upon booting the Linux kernel. This provides the entry point for the kernel. The ***Init*** architecture-dependent because different processor architectures have other ***init*** requirements. The ***init*** is also parsed and acts upon any options that are passed to the kernel. The sequence of events that ***init*** performs:

- Kernel Hardware initialization
- Opens the initial console and starts up the ***init*** process
- This process is the parent of all within GNU/Linux

Process Scheduler

The Process Scheduler is to manage the processes running in a system. The schedule is a pre-emptive one. This means that the scheduler permits a process to execute for some duration. If the process is not finished, then the scheduler temporarily halts the process and will start executing another one. The scheduler can be controlled by the scheduling policy (i.e. FIFO or Round-Robin scheduling).

Memory Manager

This is one of the important core parts of the kernel, and it provides physical to virtual memory mapping functions and vice-versa. It also provides paging and swapping to a physical disk. It works with architecture-dependent code to access the machine's physical memory. The kernel maintains its own virtual address space, so each process in the user space has its own virtual address space that is individual and unique.

Virtual File System

The Virtual File System is an abstract layer within the Linux kernel that presents a common view of differing file systems to upper-layer software. The Virtual File System interfaces to the device drivers to mediate how the data is written to the media.

Linux OS supports a large number of file systems (ie.. ext2, Minix, NFS). Rather than presenting each of these as a unique file system, Linux provides a layer into which file systems can plug their common functions, i.e. open, close, read, write, select, and so on. Therefore, if we needed to open a file on an ext2 journaling file system, we could openly use the same common function as we would on any other file systems.

Network Interface

It offers a similar architecture to Virtual File System. The network interface component is made up of 3 layers that work to abstract the details of networking to higher layers while presenting a common interface regardless of the underlying protocol or physical medium.

Loadable Modules

These are an important element of Linux as they provide the means to change the kernel dynamically. The footprint for the kernel can be small, which requires modules dynamically loaded as needed. Outside of new drivers, the kernel module component can also be used to extend the Linux kernel with new functionality.

Device Drivers

This component provides a plethora of device drivers that are available. Half of the Linux kernel source files are dedicated to devising drivers. As it is not surprising, given a large number of hardware devices out there, it does give you a good indication of how much Linux supports.

Advantages of Linux

1. Open source

Linux is an open-source operating system. We can easily get the source code for Linux and edit it to develop our personal operating system. Linux is widely used for both home and office uses. It is the main OS used for high-performance business and in web servers. Linux has had a strong influence on this world.

2. Low cost

Linux software comes with the GNU General Public License, and there is no need to spend time and huge amounts of money to obtain licenses.

3. Stability

Linux has high stability, and there is no need to reboot the Linux system to maintain performance levels as it rarely freezes or slows down.

4. Performance

Linux provides high performance over various networks, and it has the ability to handle a large number of users simultaneously.

5. Flexibility

It is very flexible and can be used for high-performance server applications, desktop applications, and embedded systems. We can install only the needed components for a particular use and also restrict the use of specific computers.

6. Compatibility

It runs most Unix software packages and also processes all common file formats.

7. Online Help

Linux help is always available on the Internet as there is probably someone out there who is in a Linux newsgroup or a mailing list who is ready to help you to solve your problem. As the source code is available, we can paste the code itself! Linux culture is one that thrives for people helping people.

8. Older Computers

If we install an older computer (Pentium III or later) lying around, we can install Linux and essentially have a new computer. In most cases, Linux runs faster, and we perform all the basics, i.e. email, play games, edit documents and web browsing, create and edit spreadsheets and PowerPoint presentations. Of course, Linux works best on new computers.

9. Debug Time

Linux applications are being developed and tested by many people. As many developers are working on it, there is a chance of quicker solutions if any problem is found. To find a solution, the debug time is only a few hours in some cases.

10. Maintenance

Maintaining the Linux operating system is easy, as the user can update the OS, and all software is installed very easily. The variants of Linux OS have their central software repository, which can update the system and keep it safe. It also offers regular updates, and the system can be updated without rebooting it. The updating can be done periodically, with just a few clicks, or users will automate the updating process. Updating in any other operating system is not so easy compared to a Linux system.

11. No anti-virus software needed

Linux does not need antivirus software to be installed on our computer. Linux has a lesser chance of being affected by viruses. The main reason for strong virus protection in Linux is due to the large number of open source developers who keep an eye on virus-related activities. The source code is updated in no time when it is available online.

12. Text editors

Linux has a wide range of text editors. Programmers can pick any free software packages like visual studio code, Vim, Atom etc. Many text editors are freely available, and we can use it without any problems.

13. Powerful command prompt

Command prompt in Linux is advanced, and a developer can perform most of the work using the command-line interface. We can install various repositories and packages through the command-line interface (CLI).

14. No reboot needed

In windows, we need to reboot the system when we install/uninstall any software. The system often becomes slow while rebooting it. But this is not the case in Linux. We do not need to reboot our system in such cases.

15. Low system specifications

We can still run Linux on old computers that have low specifications. Linux has different distributions that are available for large scale computers, servers, PC etc.

16. Good at multitasking

If we want to do batch work like printing a large file or downloading a big file, we can concurrently perform other tasks like typing/coding any program. Linux is good at such multitasking, and the system will not slow down.

17. Less disk space needed

We can still run Linux if we have limited disk space.

18. File formats

Linux supports a large number of file formats, and we can install different software packages for specific file formats.

19. Programmer-Friendly

Linux is an excellent platform for developers, and at its core, Linux is also designed to be tinker with. Linux modularity, i.e. it is possible to recreate a local testing environment for programs. Programmers/Developers can learn Linux through practical experience. Linux is flexible enough to support if the developers want to peek into the source code or build a program from scratch.

Linux Software is organized in repositories, i.e., places where official and controlled versions of almost every piece of the software reside. These repositories are maintained by the Linux community and guarantees that the applications we download from them are updated and safe.

And this type of organization saves a lot of time for Linux users to find/download an application. We need to simply search for its name in the repository and verify whether it is the latest version or not or scan it for viruses. We can update all the software on our PC with a single click.

20. Linux is secure and versatile

UNIX's idea of security is the basis for the security model used in Linux, which is known to be robust and of proven quality. Linux is not only fit for use as a fort against enemy attacks

from the Internet, but also it will adapt equally to other situations, utilizing the same high standards for security, and our development machine or control station will be as secure as your firewall.

Linux Features

1. Portability

Linux software works in various types of hardware without any deviation. We can also install the **Linux OS** on any hardware without any fear of it being incompatible, and it works in the same way whether it is high- or low-end hardware.

2. Linux OS is Open-source

The source code of Linux is available to the public for use and modified freely as many developers work together in groups to make Linux stronger and better. Moreover, developers keep on working and upgrading the systems continually.

3. Accepts many users

One of the **best features of the Linux Operating System** is that it allows multi-users to use the operating system, random access memory and applications at the same time without any issues.

4. Many programs run simultaneously

We can run many program applications on the Linux OS at the same time. This is one of the best features of the Linux Operating System.

5. Linux Operating System has ordered file system

We have ordered a filing system in the Linux OS, and all the user files and system files are arranged in order.

6. Application Support

Linux has a software repository that allows users to download and install commands in shell or Linux Terminal, and it can also run Windows applications too.

7. Multi-user

In Linux, multiple users can access the resources like ram, memory, and applications at the same time.

8. Adaptable

Linux can work on any of the hardware in a simple and efficient way. The software is portable to any of the devices and needs Linux Kernel and other applications to be installed properly to make it work.

9. Hierarchical

Linux has a standard hierarchy structure, and it has system and user files that are arranged in a structured form.

10. Shell

Linux provides an interpreter program used to execute command operations of the OS, and it is also used to do various types of operations like call operation and so on. Linux Shell works like a programming language, and it provides commands and keywords.

11. Graphical User Interface

Linux has a lot of packages, and it can be installed to make OS graphics as windows.

12. Application Support

Linux software repository allows users to download and install just by a command in shell or Linux Terminal. It can also run Windows applications if in case the user wants to.

13. User-Friendly

Linux is a user-friendly software and it is fast, free and easy to use.

14. Security

Linux platforms' provides high-security levels when compared with others, and it provides a virus-Free environment. It uses the regular virus prevention time needed when working with other operating systems for other more important tasks. It also offers user security systems with authentication features such as encryption of data or password protection or controlled access to particular files.

LINUX has two levels of securities:

- **System-level Security:** Controlled/Maintained by the system administrator.
- **File-level Security:** Controlled/Maintained by the owner of the file.

15. Support's most national or customized keyboards

Linux is used globally, and it is available in multiple languages and also supports most of the custom national keyboards.

16. Live CD/USB

Linux distributions have Live CD or USB features through which users can run or try the operating system even without installing it.

17. Other Features

- Linux has improved Management code
- Linux has improved TCP/Performance
  - Linux has internal kernel threads for handling dependencies between loadable modules and automatic loading of modules on demand.
- Linux has Standardized configuration interface
- Linux Support for a range of SCSI controllers for high performance disk access
- Linux Support for UNIX standard TCP or IP protocols
- Linux Device driver also support for running IP over an Ethernet
- Linux provides much functionality from limited resources and runs a machine having 4MB RAM.
- Linux has live patching capabilities and enables a running system to be patched without the need for a full system reboot. With Linux 5.1, a new capability is being added to live to patch, i.e. Atomic Replace.
- Linux has enabled regular storage devices, including hard drives, to be used for limited forms of memory usage, including swap space. With the use of Linux 5.1, administrators will now be able to more fully use storage, and particularly a class of storage now often referred to as "persistent memory" as regular system memory.
- AMD Free Sync Supporter.
- Implementing Official support for the Raspberry PI Touchscreen.
- Data Encryption.
- Authentication support for ARM Pointer
- Linux Kernel Lockdown: This model aims to enhance Linux kernel security by separating the userland and kernel code, and it also limits the root account access to kernel features that enable arbitrary code modification and execution and in this way. The Linux kernel remains safe even when the root account is compromised.

- ExFAT File System Support: exFAT file system supports large file size exFAT–formatted drives for data storage and transfer. This improvement comes in the wake of Microsoft's recent (August 2019) move to open source and publish its proprietary file system.

- Extended audio hardware support: Linux provides extensive audio support for a No. of hardware components via its open-source sound improvement drivers. It supports several new audio hardware, including the Freescale i.MX7ULP, Cirrus Logic CS47L15/CS47L92 etc.

Difference between Unix & Linux

Linux	Unix
Linux is a Free & Open Source software development Operating System.	Unix is an operating system that can be utilized by its copywriters.
Cost: Linux can be freely distributed and downloaded. The prices for different versions for Linux are different, but they are normally cheaper than Windows.	Different versions of Unix have different prices.
Users: It can be used by everyone—i.e., home users to developers and computer enthusiasts, etc.	Users: Unix OS was developed mainly for mainframes, servers, workstations etc. The Unix environment/ client-server program model were essential elements in the development of the Internet.
The Linux kernel is developed for sharing and collaboration of code by the community of developers. Linus Torvalds is the father of Linux.	Unix has three distributions: Solaris, AIX & HP-UX. IBM, Apple also uses UNIX for OS X operating systems.
Usage: Linux can be used on a wide variety of computer hardware such as mobile phones/tablets, computers, mainframes, supercomputers etc.	Usage: The UNIX OS is used in servers, workstations & PCs. Support for the majority of financial infrastructure and for many 24/365 high availability solutions.
Processors: Linux has dozens of processes of various kinds.	Processors: Unix has x86or x64, Sparc, Power and many other processors.
Architectures: Linux is originally created for Intel's x86 hardware, and ports available for over two dozen CPU types, including ARM.	Processors: It is available on PA-RISC and Itanium machines. Solaris is also available for x86/x64 based systems.
Features: One of the important killer	Features: ZFS / Next-generation file system, i.e.,

features is that the kernel is updated without reboot.	Dtrace-dynamic kernel tracing.
GUI: Linux provides two types of GUIs, i.e. KDE and Gnome. GUI Linux is optional.	Unix Initially was a command based Operating System and after that Graphic User Interface was created called Common Desktop Environment, and now many of the distributions come with Gnome.
File System Support: Ext2/ Ext3/ Ext4, Jfs, FAT/ FAT3/ NTFS formats.	File System Support: gpfs/ hfs+/ zfs etc formats.
Text-mode interface: BASH (Bourne Again SHell) is Linux's default shell. It can support multiple command interpreters.	Text-mode interface: Originally known as the Bourne Shell. Now, it is compatible with others' including BASH, Korn & C.
Security: Linux virus has been reported between 60-100 till date approx.	Security: UNIX viruses have been reported between 85-120 viruses approx.
Development and Distribution: Linux is an Open Source developed through sharing and collaboration of code and features by forums etc. It is distributed by different vendors like Debian, Red Hat, SUSE, Ubuntu, GentuX etc.	Development and Distribution: Unix systems are categorized into different categories, mostly developed by AT&T and or various commercial vendors and non-profit organizations.
Threat detection and solution: As far as Linux is concerned, the detection of threat and solution is very fast. Linux is especially community-driven, and when any Linux user posts any kind of threat, several developers start working on it from all corners of the world.	Threat detection and solution: Due to the proprietary nature of the real Unix, users need to wait for a while to reveal the proper bug fixing patch. But these aren't as common.
Inception: Inspired by MINIX and after adding many features of GUI/ Drivers etc., the framework of the OS developed by Linus Torvalds became LINUX. LINUX kernel was released in 1991.	Inception: Developed by a team of AT&T employees at Bell Labs, it was written in C language and was designed to be a transportable & multi-tasking / multi-user system within time-sharing configuration.
Various Versions of Linux are Redhat/ Ubuntu/ OpenSuse, etc.	Various Versions of Unix are HP-UX/ AIS,/BSD, etc.
Source Code of Linux is available to the	Source code isn't available to the general public.

general public.	
Linux is considered a Unix clone, and it behaves like Unix but doesn't contain its code.	Unix contains different coding, which is developed by AT&T Labs.
As Linux is open-source, whenever a user posts any issue of threat, developers from all over the world start working on it. And hence, it provides a faster solution.	In Unix, users got to keep waiting a while for some time for the issue to be resolved.
Linux is written in C / other programming languages.	Unix is mostly written in C and assembly languages.
Linux is Multilingual.	Unix is written in English only.
Kernel Type: Monolithic kernel	Kernel type varies. It is often monolithic, MicroKernel and Hybrid.
Linux/ Unix is considered best protected against malware. This is often characterised by a lack of root access and quick updates and also a comparatively low market share. There has been no widespread Linux virus.	Unix is additionally considered to be very safe. It's even harder to infect because the source is code not available, and there's no actively spreading the virus for UNIX nowadays.
Linux has a big market share with 25 million machines installed.	Unix installed on 5.5 million approx.
Limitations of Linux: 1. There's no standard edition of Linux. 2. Linux has patchier support for drivers that can cause system-wide errors. 3. Linux, at least for new users, is not as easy to use as Windows. 4. Many of the programs we use for Windows will only run on Linux using a complicated emulator. For example, Microsoft Office. 5. Linux is best suitable for a corporate user. It's much harder to introduce in a home setting.	**Limitations of Unix:** 1. The unfriendly, terse, inconsistent, and non-mnemonic user interface. 2. Unix Operating system is designed for a slow computer system, so you can't expect fast performance. 3. The Shell interface can be treacherous because typos can destroy files. 4. Versions on different machines are slightly different, so it lacks consistency. 5. Unix does not provide a secure response time for hardware failure, so it does not support real-time response systems.

Linux Vs Windows

	Windows	Linux
Cost	Microsoft Windows typically costs between $99.00 and $199.00 USD for every single licensed copy. Windows 10 was offered as a free upgrade for existing Windows owners. However, the deadline for that offer has long since elapsed.Windows Server 2016 data centre has a price starting at $6155.	Linux license remains completely free. However, organizations that need Linux support can choose paid subscriptions for platforms like RedHat and SUSE. It's better to go with these subscriptions. Otherwise, competent in-house Linux expertise can be expensive. Talking about the infrastructure cost, other things remaining equal (being on-premise or on the cloud), Linux being lightweight, we can expect 20% more on Linux as compared to Windows.
Licensing model	Proprietary software.	The Linux kernel and most of the distributions are open-source and (often) available under a GNU General Public License (GPU).
User-friendliness	From the outset, Windows was designed to be as simple to use as possible, even for users with no IT knowledge.	While it's true that Ubuntu, in particular, is relatively easy for Windows users to grasp, with most of the other Linux distributions, there are significant barriers to entry for newcomers.
GUI(Graphical User Interface)	Microsoft's standards are according to its Windows GUI.	Users of Linux distributions have many freedoms in designing the GUI and can even do away with it altogether.
Support	Windows provides a large range of support services, i.e. within the system and online and also detailed specialist literature available targeted at users with various levels of knowledge.	Support of Linux comes from the extensive user community, and we can find an answer to any problem in the online forums/Wikis.
Software	Windows directs the greatest number of desktop users, and so the biggest selection of commercial software from third-party developers, many of which are not Linux	There are far fewer applications designed specifically for Linux, although some Windows programs can run on Linux.

	compatible. It also leads in video games by a broad margin.	
Installing programs	Programs can be installed either by downloading them from websites or from a hard disk.	The majority of programs, drivers and packages in Linux, provided via fixed repositories.
Uninstalling programs	Components remain on the system when some programs are uninstalled.	Programs in Linux are always completely removed when some programs are uninstalled.
Hardware	Windows drivers are easily available for any type of hardware.	Hardware support is more limited. Some drivers are not available immediately.
Reliability	Although the stability of Windows has improved in recent years, most Linux distributions are still far superior in this respect.	Linux and its distributions are known for being very stable to run.
Security	Windows systems are regularly threatened by viruses and other malware.	Linux systems are only attacked very rarely.
Speed	Windows is known for slowing down the longer it is used.	Linux generally runs very quickly.
Updates	Users are sometimes forced to install updates. Most updates are stable.	Each user decides which updates they want to install. These are sometimes experimental and therefore not recommended for all users.
Trial version	You can download and install a trial version of Windows without paying for a license.	Since most of the distributions are free of charge, you can easily try them out. Many of them can even run off a USB flash drive, so you don’t even need to install them.
Types of user	Windows is aimed at various types of user, i.e. from occasional to expert. Due to the availability of a wide range of Windows-compatible software, the OS	Using Linux requires a minimum level of specialist knowledge. Computer games are only available in limited versions and Linux distributions tend to be suited to most professional IT users.

	is suitable for professional as well as for entertainment use.	
Access	Every user does not have access to the source code. Only the selected members of the group have access to the source code.	Users have access to the source code of the kernel and can modify it accordingly. This gives the benefit that bugs in OS will be fixed faster. However, the drawback is that the developers may take undue advantage of the loophole.
Privacy	Windows collects all the user data.	Linux Distros do not collect users data.
Customization	Windows has very limited customization options available.	Linux has a variety of distributions that are highly customizable based on the user's requirements.
Data Drives	Windows uses different data drives like C: D: E to stored files and folders.	Linux uses a tree-like hierarchical file system.
Drives	Windows has different drives like C: D: E.	There are no drives in Linux.
Hard Drives	Hard drives, CD-ROMs, printers are considered as devices.	Peripherals like hard drives, CD-ROMs , printers are also considered files in Linux.
User Accounts	There are four types of user account types 1) Administrator, 2) Standard, 3) Child, 4) Guest	There are three types of user account types 1) Regular, 2) Root and 3) Service Account
Admin Privileges	Administrator users have all administrative privileges of computers.	The root user is the superuser and has all administrative privileges.
Naming convention	In Windows, you cannot have two files with the same name	Linux file naming convention is case sensitive. Thus, sample and SAMPLE are two different files

	in the same folder.	in the Linux/Unix operating system.
Home Directory	My document is the default home directory for windows.	/home/username directory is created in linux is the home directory.
Supported platforms	PowerPC: versions 1.0 – NT 4.0; DEC Alpha: versions 1.0 – NT 4.0;MIPS R4000: versions 1.0 – NT 4.0;IA-32: versions 1.0 – 10; IA-64: version XP; x86-64: versions XP – 10; ARM: version RT;	All
Updated Method	Windows Update, Microsoft Update, Microsoft Update Catalog	Many
Development & Distribution	Developed & distributed by Microsoft Corporation.	Linux is developed by Open Source development, i.e. through sharing and collaboration of code, and it is distributed by various vendors.
Preceded by	MS-DOS	Basic Terminal (CLI)
License	Proprietary	GNU General Public License
Run Level	In windows, if we encounter any problem in order to fix it, we need to reboot at run level three as an administrator/root to find and fix the problem.	Linux has an inbuilt ability to stop at different run levels. With this, we can work using the command line and GUI if anyone has an issue.

Linux Distribution

A Linux distribution or a Linux Distro is an operating system made from a software collection based upon the Linux Kernel or a package management system. Linux users get their operating system by downloading one of the Linux distributions available for the various systems ranging from embedded devices and personal computers (Ex: Linux Mint) to powerful supercomputers.

Linux Distribution consists of Linux Kernel, Libraries, GNU Tools, and additional software in a desktop environment. Most of the software is free and open-source.

Due to huge availability of software, distributions have taken in a wide variety of forms i.e. for use on desktops, servers, laptops, netbooks, mobile phones etc as well as minimal environments for use in embedded systems. As there are commercially-backed distributions, like Fedora , openSUSE and Ubuntu and entirely community-driven distributions, such as Debian, Slackware, Gentoo and Arch Linux. Major distributions are pre-compiled and ready to use for a specific instruction set and some distributions (such as Gentoo) are distributed mostly in source code form and compiled locally during installation.

List of Linux Distributions:

1. Ubuntu

Ubuntu is an open-source operating system sponsored by **Canonical company** and a trademark registered by the same company. Ubuntu is based on Debian Linux distribution and is composed of several open-source software and is available for free for businesses.

Ubuntu is also famous for its ease of use and its inclusion of a migration assistant for Windows users, and support for the latest technologies. Ubuntu is available in various flavours targeted at specific niches, Kubuntu, Xubuntu and Lubuntu etc.

2. Fedora

Fedora is upstream of commercial RHEL distribution. It is special because it uses the latest technology and packages from the open-source world rather than RHEL. It also uses a Yum package manager like RHEL.

3. Linux Mint

Linux Mint is based on Ubuntu or Debian is 32- and 64-bit Linux distribution for desktop computers. It started with simply being Ubuntu with pre-installed and full out-of-the-box multimedia support and proprietary software such as Adobe Flash. Newer versions of the Ubuntu-based Linux Mint have been released every six months.

4. OpenSUSE

OpenSUSE started out a German translation of Slackware and is a community supported by SUSE distribution and is the successor to SUSE Linux Professional and serves as the basis for SUSE Linux Enterprise products.

Newer versions are released every eight months and support many languages, and each release is provided with security updates for a period of 18 months.

It is a completely open-source system, and it contains therefore not proprietary drivers or codecs to support most closed multimedia formats. However, it is possible to install the packages to take advantage of these materials, as well as drivers for ATI or Nvidia.

5. PCLinuxOS

PCLinuxOS is a Linux distribution often shortened to PCLOS, with KDE, MATE etc., as its default user interfaces. It is a free software operating system for personal computers, and it is considered a rolling release. And this distribution makes it easy to install drivers, edit photos, get Office software, and start using multimedia. PCLinuxOS uses apt and synaptic to handle RPM packages. PC Linux OS is a rolling distro. It also supports Flatpak. The PCLinuxOS teams offer three different versions **KDE**, **MATE**, and **XFCE**.

6. Debian

Debian is a free OS for computers. It is the most important available distros at present. It serves as the base for Ubuntu, and many users consider it as the best one suited. With Debian, we have the possibility to use all open-source components. Debian has a slow release cycle, having a period of 1 to 3 years.

7. Sabayon/Gentoo

Sabayon Linux is a Linux distribution based on Gentoo, and Sabayon installs the base system from precompiled packages.

Sabayon Linux is a ready distribution that includes the basic proprietary drivers of different graphic and wireless cards, audio codecs – video, flash, java, etc. so that the system can work without being installed after installation. It is not a free tool completely.

8. Arch Linux

Arch Linux distribution was created by Judd Vinet. Arch Linux was designed for advanced users to be the perfect operating system. It adheres to 5 principles: User centrality,

pragmatism, modernity, simplicity, versatility that means that project attempts to have minimal distribution-specific changes, minimum breakage with updates etc. Arch Linux uses a rolling release model meaning no major releases of new versions of the system, and regular updates are needed to obtain the latest arch software.

9. Puppy Linux

Puppy Linux is a Linux distribution made available in Live CD created by Barry Kauler with a very small size, i.e., 100 MB. 170 MB for the latest version. This distribution is designed for lightweight, reliable and easy to use while retaining maximum functionality.

The entirety of OS and all programs can be loaded into RAM. This allows you to remove the boot media after initialization. It provides a package manager which facilitates the installation of new software such as big Linux distributions. This interface has installed / installable packages and automates installation / uninstallation, avoiding the often confusing process of manual installation.

10. CentOS

CentOS is one of the Linux distribution which is available freely (no license) and has strong and big community supported computing platform. Centos 5, 6, 7 and 8 versions will be maintained up to 10 years based on RHEL. There is no need to upgrade CENTOS to the next major and is not limited by upstream restrictions. There are 3 primary CENTOS repositories containing software packages that make CentOS distribution.

1. base: This contain packages that form centos point releases and these are updated when actual release is available in form of ISO images
2. updates: This contains packages and serves as security or enhanced updates etc for point release.
3. addons: Contain packages required for building packages that make the main CentOS distribution but not provided by upstream.

Choosing a Linux distro

distribution name	reason(s) for using
Red Hat Enterprise (RHEL)	Used to want a good support contract.
CentOS	Used without the support contract from Red Hat.

Fedora	Want to have Red Hat on your laptop/desktop.
Linux Mint	I want to have a personal graphical desktop to play music & games movies.
Debian	Used for laptops, servers or any other device.
Ubuntu	Ubuntu based on Debian
Kali	Used to have a pointy-clicky or hacking interface.
others	Advanced users will prefer Arch, Gentoo, OpenSUSE, Scientific, ...

Linux Bash

Bash (Bourne Again SHell) is a Unix shell and a command language interpreter(CLI). A shell is an interface and command line interpreter that executes commands. Bash shell packaged by default for most Linux distributions and is a successor for the Korn shell (KSH) and the C shell (csh).

We can use bash on most Linux and OS X operating systems by using a terminal. Let's see a simple hello students example. Open up your terminal, and write the following line (everything after the $ sign):

abcxyz:~$ echo "Hello students!"

Hello students!

We can see that we used the echo command to print the string "Hello students!" to the terminal.

Writing a bash script

We can put bash commands into a .sh file, and run them from the command line. Consider bash script with the following contents:

#!/bin/bash

echo "Hello students!"

The first line starts with script #! and its special directive which Unix treats differently.

use of #!/bin/bash at beginning of the script file:

The first line indicates to Unix that the file is to be executed by /bin/bash. It is the standard location of the Bourne shell on just about every Unix system. Adding #!/bin/bash as the first line of the script tells the operating system to invoke the particular shell to execute the commands that follow in the script. #! and referred to as hash-bang, or she-bang or sha-bang, and it is executed only if we run the script as an executable.

We can run the script to make the file executable and we call this command under sudo chmod +x 'filename'.

abcxyz:~$ ./myBashScript.sh

Hello students!

The script only has two lines. The first indicates what interpreter to use to run the file (ex: bash). The second line is the command we want to use, echo, followed by what we want to print, which is "Hello students".

Due to the permissions set on the file, sometimes we get an error in executing the script. So, it is recommended to use the following command:

abcxyz:~$ chmod u+x myBashScript.sh

After that, execute the script.

Linux Command Line: Bash Cat

Cat is used to read a file and print it to the standard output. It is derived from its function to concatenate files.

Usage

cat [options] [file_names]

Example

display in terminal the content of file.txt:

cat file.txt

Concatenate two files and display the result in terminal:

cat file1.txt file2.txt

Linux Command Line: Bash cd

Change Directory to the path specified, for Ex: cd projects.

There are a few helpful arguments for this:

- . refers to the current directory, such as ./projects
- .. to be used to move one folder up, use cd and can be combined to move up multiple levels i.e. ../../my-folder
- / is the root of system to get core folders, such as system users.
- ~ is the home directory, and the path /users/username. Move back to folders reference to relative to this path by including it at the start of path, for Ex: ~/projects.

Linux Command Line: Bash head

Head is used to print the first ten lines (by default) or any other amount specified to a file or files. The head command allows you to view the first N lines of a file.

If more than one file is called, then the first ten lines of each file are displayed unless a specific number of lines are specified. Choosing to display the file header is optional using the option below.

Usage

head [options] [file_name(s)]

Most used options:

Example

head file.txt

Prints in terminal the first ten lines of file.txt (default)

head -n 7 file.txt

display in terminal the first 7 lines of file.txt

Linux Command Line: Bash ls

ls is a command on Unix-like operating systems to list contents of a directory, for example folder and file names.

Linux Command Line: Bash man

Man, i.e. **man**ual, is a bash command used to display on-line reference manuals of the given command.

Usage

man [options] [command]

Example

Display the man page of ls:

man ls

Linux Command Line: Bash mv

Moves files and folders.

mv source target

The first argument is the file you want to move and the second is the location to move it to.

Arithmetic Operators in Bash:

Mostly used Arithmetic Operators in Bash

Operator	Operation
+, -, *, /	addition, subtraction, multiply, divide
var++	Increase the variable var by 1
var--	Decrease the variable var by 1
%	Modulus (Return the remainder after division)

Ex:

x=8 ,y=2

echo "x=8, y=2"

echo "Addition of x & y"

echo $(($x + $y))

output : 10

Bash String

Bash String is a data type such as an integer or floating-point unit. It is used to represent text rather than numbers. It is a combination of a set of characters that may also contain numbers.

For example, the word "GreatLearning" and the phrase "Welcome to Great Learning" are the strings. Even "01234" could be considered as a string, if specified correctly.

Bash Comments

For single line comments in bash we should use hash symbol at the beginning (#). HashBung (#!) in the first line of the script file is the exception. Example: Bash Script with single line comments in between commands.

Ex:

#!/bin/bash

 # single line comment in bash

Bash Array

Bash Array Declaration

To declare a variable as a Bash Array, use the keyword declare. The syntax is:

```
declare -a ARRAY_NAME
```

Bash Array Initialization

To initialize a Bash Array, use assignment operator =, and enclose all the elements inside braces (). The syntax to initialize a bash array is:

```
ARRAY_NAME=( ELEMENT_1 ELEMENT_2 ELEMENT_N )
```

Access elements of Bash Array

We can access the elements of Bash Array using the index.

```
echo ${ARRAY_NAME[2]}
```

Print Bash Array with all the information

In order to print all the elements of a bash array along with all the index and details, use declare with option p. They syntax to print the bash array is:

declare -p ARRAY_NAME

Bash For Loop

It is used to perform repetitive tasks and helps us to iterate a particular set of statements over a series of words in a string/elements in an array.

Syntax of For Loop:

```
for variable in list
do
commands
done
```

Bash If

In BASHIF, if a particular condition is true, then only execute a given set of actions. If it is not true, then it will not execute those actions.

Syntax:

```
if [ expression ];
then
statements
fi
#!/bin/bash

read-p " Enter a No. of your choice : " value

if [ $value -gt 200 ]
then
echo "entered No. is greater than 200"
fi
```

Output:

Enter No. of your choice : 300

The entered No. is greater than 200

Bash Else If

Bash else-if statement is used for multiple conditions. In Bash elif, there are several elif blocks with a boolean expression for each one of them. In the case of the first 'if statement', if a condition goes false, then the second 'if condition' is checked.

Syntax of Bash Else If (elif)

```
if [ condition ];
then
<commands>
elif [ condition ];
then
<commands>
else
<commands>
fi
```

Bash Case

The Bash case statement is the simplest form of IF-THEN-ELSE with many ELIF elements. Using the case statement makes our bash script more readable and easier to maintain. These are generally applied to simplify the complex conditions having multiple different choices.

Case Statement Syntax

Syntax of the bash case statement is given below:

```
case expression in
  pattern_1)
        statements
        ;;
    pattern_2)
        statements
        ;;
    pattern_3|pattern_4|pattern_5)
        statements
        ;;
    pattern-n)
        statements
        ;;
    *)
        statements
        ;;
esac
```

Bash While Loop

The **bash while loop** can be defined as a control flow statement which allows executing the given set of commands repeatedly, as long as the applied condition evaluates to true. For example, we can either run echo command many times or just read a text file line by line and process the result by using while loop in Bash.

Syntax of While Loop:

```
while [ expression ];
do
commands;
multiple commands;
done
```

The above syntax is applicable only if the expression contains a single condition.

Bash Until Loop

Bash Until Loop in a bash scripting is used to execute a set of commands repeatedly based on the boolean result of an expression. The set of commands are executed only until the expression evaluates to true.

Syntax:

```
until [ expression ];
do
command1
command2
. . .
. . . .
commandN
done
```

Set Environment Variable

Environment variables are a set of key-value pairs stored on Linux systems used by processes to perform specific operations.

Environment variables can also be used in shell programs or subshells to perform various operations (for example, knowing if the current user is the root or not).

Environment variables in Linux have global/local scope. A global scope environment variable can be accessed in a terminal from anywhere where a particular environment exists in the terminal, i.e. it can be used in all kinds of scripts /programs /process running in the env bound by that terminal, and the local scoped environment is defined in a terminal which cannot be accessed by any program /process running in the terminal. It can be accessed only by the terminal.

Syntax:$NAME

Both Global/local accessed in the same way.

To display : $echo $name

To display all environments : $printenv or $set or $env

Set Environment Variables on Linux using export

On Linux, there are many different ways of setting environment variables.

The best way to set environment variables is to use the export command.

To set global environment :

$ export VAR=”value” or set VAR=”Value”

To Set local environment :

$ NAME =Value

Using export, the environment variable will set for the current shell session.

If you open another shell or restart system, the environment variable won't be accessible anymore.

To get the value of your environment variable, use the "printenv" command with the variable.

$ printenv <variable>

Ex:

$ printenv VAR

value

Alternatively, we can use Linux pipes to get the value of environment variables.

We can also use the echo command but need a dollar sign before the variable name.

$ echo $VAR

Value

To set user wide ENVs

These variable are set and configured in ~/.bashrc, ~/.bash_profile, ~/.bash_login, ~/.profile
files according to the requirement.These variables can be accessed by a particular user and persist through power offs.

Following steps can be followed to do so:
Step 1: Open the terminal.
Step 2:

$ sudo vi ~/.bashrc

Step 3:Enter password.
Step 4:Add variable in the file opened.

export NAME=Value

Step 5: Save and close the file.
Step 6:

$ source ~/.bashrc

Set Persistent Environment Variables on Linux

environment variables were not persistent over shell restarts.

In order to make changes persistent use system files that are read and executed on specific conditions.

Using the .bashrc file

To set environment variables persistently it is better to add them to the ".bashrc" file and bashrc file script executed as and when we initialize an interactive shell session and launch a new Terminal via the GNOME interface or use a screen session.

For Ex: add the following entries to your .bashrc file.

$ export TZ="India/Hyderabad"

training@debain:~cat .bashrc | tain –n 10

sources /etc/bash.bashrc).

If ! shopt –oq posix; then

if [-f/usr/share/bash –complete /bash_complete]; then

elif [-f /etc/bash_complete]; then

. /etc/ bash_complete

fi

fi

export TZ="India/Hyderabad"

training@debain:~$

Save file and use the source command to reload the bashrc file for current shell session.

$ source ~/.bashrc

We can print your new environment variable with "printenv" andcheck how date was set on Linux by modifying TZ.

$ printenv TZ

India/Hyderabad

$ date

Sat 19 Oct 2019 10:03:00 AM IST

changes are now updated/persistent over shell or system reloads.

Using .bash_profile

If we plan on connecting to sessions via "login shells", we can also add environment variables directly into the .bash_profile file.

$ export TZ="India/Hyderabad"

$ source ~/.bash_profile

Using etc/environment

If we want to enforce specific environment variables for everyone then define system-wide environment variables.

To set system wide environment variables on Linux, we need to export our variables in the /etc/environment file.

Ex: , To change the editor used globally, we can modify the EDITOR variable in the environment file.

$ export EDITOR=”vi”

training@debain:~$ cat/etc/environment

export EDITOR=”vi”

training@debain:~$

Now login as different users on our system, and will see that the EDITOR variable is set for everybody on the server.

manu@debian:~$ printenv EDITOR

vi

successfully set your environment variables persistently on Linux.

Set Environment Variables in one line

shortcuts in order to set them easily.

One line for .bashrc

$echo”export VAR=” value “”>> ~/.bashrc && source ~/.bashrc

One line for bash_profile

$echo ‘exprt VAR=”value ‘ >> ~/.bash_profile && source ~/.bash_profile

One line for /etc/environment

$ echo “export VAR=”value””>>/etc/environment && source /etc/environment

Unset Environment Variables on Linux

In Linux, there are 2 ways of unsetting environment variables : i.e. by using 1) unset command 2) by deleting variable entries our system files.

unset command :

Syntax to unset an environment variable, $ unset <variable>

$ unset USERNAME

$ printenv USERNAME

<nothing>

set -n command :

unset environment variables using the set command with the “-n” flag.

$ set -n USERNAME

$ printenv USERNAME

<nothing>

Common Set of Environment Variables on Linux

USER: the current username using the system;

EDITOR: To perform file edits on your host;

HOME: home directory of the current user;

PATH: list of directories which are colon separated where the system looks for commands;

PS1: the primary prompt string

PWD: the current/present working directory;

Most recent command executed on the system

MAIL: used to send emails from command line

SHELL: the shell used in order to interpret commands on the system, i.e. bash, sh, zsh and others

LANG: language encoding used on the system;

DESKTOP_SESSION: the present desktop used on host (GNOME/ KDE)

HISTFILESIZE: No. of lines of command history stored file;

HISTSIZE: No.of lines of history allowed in memory;

UID: the current UID for the user

Set PATH environment variable on Linux

To display current PATH environment variable, printenv command is used

$ printenv PATH

/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games

In order to set the PATH environment variable we should add an export line to our .bashrc file and source it.

$echo"export PATH="<path>:$PATH"" >> ~/.bashrc && source ~/.bashrc

$ printenv PATH

Linux Set command

Linux set command is employed to set/ unset certain flags/settings within the shell environment. These flags and settings determine the behavior of an outlined script and help in executing the tasks without facing any issue. The values of shell attributes and parameters are often changed or displayed by using the set command.

Syntax

Bourne shell (sh):

set [–aefhkntuvx[*argument*]]…

C shell (csh):

set [*var*[=*value*]]

set *var* [*n*] = *word*

Korn shell (ksh):

set [+-aefhkmnopstuvx] [+-o *option*]... [+-A *name*] [*arg*...]

Options: Bourne Shell (sh)

In **sh**, the **set** built-in command has the below options:

—	A double-dash ("**—**") option specifies the end of an option list. This option is mainly useful when values listed after the choices will start with a dash themselves.
-a	Mark variables that are edited or created for "export"; environment variables set during this way are going to be passed on to the environments of any subsequent commands.
-e	This option "Exit" immediately if a command exits with a non-zero exit status.
-f	Disable file name generation (globbing).
-h	search and remember function commands as the functions are defined
-k	All arguments of KEYWORD are placed in the environment and not just those that precede the command name.
-n	It will not execute the command but read them.
-t	executes one command and exit after reading
-u	unset variables are treated as an error when substituting.
-v	display shell input lines on reading.
-x	display commands and their arguments as soon as they are executed.

Using **+** instead of **–** causes these flags to be turned off. These flags also can be used to embellish the shell itself. The current set of flags could also be found within in the **$-**. variable The remaining arguments are place parameters and are given in respectively **$1**, **$2**, etc. If arguments are not given , values of all names are displayed.

For every name, the **unset** command deletes the corresponding variable / function() value. The unique variables **PATH**, **PS1**, **PS2**, **MAILCHECK**, and **IF** cannot be unset.

With the built-in **export** command, the given names are marked for automatic export to the environment of next executed commands. If no arguments are presented, variable names that are marked for export for the duration of the current shell's execution are listed. Function names aren't exported.

Options: C Shell (csh)

In **csh**, If no arguments are mentioned, **set** prints the values of all shell variables. Multi-word values are printed as a parenthesized list. With the *var* argument alone, **set** assigns an null value to the variable *var*. **set** assigns value to *var* for the arguments of the type *var* **=** *value*.

Word	A single word /quoted string).
(*wordlist***)**	list of words with space-separated enclosed in parentheses.

Values are command/file name expanded before they are assigned. The form **set** *var***[***n***]=***word* , the *nth* word in a multi word replaces *value* with *word*.

unset deletes variables whose names match pattern. All variables are deleted by "**unset ***"; this is a very bad practice if we don't know what we are doing.

Options: K Shell (ksh)

In **ksh**, the **set** command takes the below options:

	-A	Array assignment. Unset the variable name and assign values sequentially from the list arg. If **+A** is employed, the variable The name isn't unset first.
	-a	subsequent parameters that are defined are automatically exported.

	-e	If set and exists, execute the **ERR** trap if commands has non-zero Status and exit. This mode is disabled while reading profiles.
	-f	Disables file name generation/ substitution
	-h	Designate command becomes a tracked alias when first encountered.
	-k	Places parameter assignment arguments in environment to a command, not just whose arguments precede command name

	-m	Background jobs would run in a different process group and a line and will print lines on completion. The exit status of these background jobs are reported on completion messages. systems with job control these flags are turned on automatically for interactive shells.
	-n	This will read commands & check them for syntactical errors, but don'texecute them and also the flag is Ignored for interactive shells.
	-o *option*	Print current *option* argument and if no option specify an arguments Gives error messages.

Allexport	Similar to **–a** flag
Errexit	Similar to **–e** flag
Bgnice	most background jobs are run at a lower priority, which is the default mode.
Emacs	enters in an **emacs** style in-line editor for command entry.
Emacs	enters in a **gmacs**-style in-line editor for command entry.
ignoreeof	The shell will not exit on end-of-file. Its command exit must be used.
keyword	Same as **-k**.flag
markdirs	Append slash(/) for all director names that result of filename substitution.
monitor	Same as **–m** flag
noclobber	Prevents redirection through truncating existing files. Use redirection symbol(**>|) right care pipe)** to truncate a file when turned on.

Noexec	Same as **-n**.
Noglob	Same as **-f**.
Nolog	Don't save in history file the function definitions
nounset	Same as **-u**.
privileged	Same as **-p**.
Verbose	Same as **-v**.
Trackall	Same as **-h**.
Vi	Puts you in insert mode of a vi-style in-line editor till we hit escape. This option argument puts you in control mode. A return sends the line.
Viraw	processed every character is as it is typed in **vi** mode.
Xtrace	Same as **-x**.

If no option name is supplied then the present option settings are printed.

Using **+** instead of **–** causes above flags to be turned off. These flags can also be used upon invocation of the shell. The present set of flags may be found in **$-**. until **-A** is specified, the remaining arguments are positional parameters and are assigned, in order, to "**$1 $2** …".

Using **+** instead of **–** causes the above flags to be turned off. These flags can also be used upon invocation of the shell. The present set of flags may be found in **$-**. until **-A** is specified. The remaining arguments are positional parameters and are assigned, in order, to "**$1 $2** …". If no arguments are specified, then the names and values of all variables are displayed on the standard output.

The variables given by the list of names are automatically unassigned; their values and attributes are deleted. Read-only variables cannot be **unset**. If the **-f**-flag is set, then the names point out to function names.

While using **unset**, the variables given by the array of names, in the same way, are unassigned, and their values & attributes are deleted, and read-only variables can't be **unsent**.

With the **built-in export** command, the said names are marked for automatic export to the environment next to -executed commands. **KSH** commands that are preceded by one /two "*****" characters are treated specifically in the following ways:

- Variable assignment lists preceding the command stay in effect when the command completes.
- Input or Output redirections are processed next to variable assignments.
- Errors can cause a script that contains them to abort.
- Words following a command or preceded by ** that is in the form of a variable assignment, expanded with the same rules as a variable assignment. This assignment that is tilde (~), is performed after the **equal** sign and word splitting and file name generation are not performed.

Examples:

set PATH="/bin:/usr/bin:/usr/sbin:usr/local/bin"

In **csh**, the command sets the environment variable **PATH**, so that shell can search for files in the **/bin**, **/usr/bin**, **/usr/sbin** and **/usr/local/bin** directories, in those order.

Linux Export Command

Introduction

Linux export commands make a variable something that will be included in the child process environment. It will not affect other already existing environments. There is no way to set variables in one terminal and have them automatically appear in another terminal. The environment is established for each process on its own.

The **export** command is bash shell BUILTINS commands, i.e. it is part of the shell. The **export** command is easy to use, and it has direct syntax with only three available command options. The shell gives the export attribute to the variables associated with the required names that cause them to be an environment of subsequently executed commands.

Frequently Used Options

- **-p**
  List all names that are exported in the current shell
- **-n**
  Remove names from export list

- **-f**
  Names are exported as functions

Export basics

For example:

$ ai=greatlearning.in

$ echo $ai

greatlearning.in

$ bash

$ echo $ai

$

Line 1: new variable called “ai” is created with string “greatlearning.in”

- Line 2: echo command to print out a contents of the variable “ai”
- Line 3: created a new child bash shell
- Line 4: variable “ai” have no any values defined

It can be seen from above, any new child process forked from a parent process by default does not inherit parent’s variables. This is where the **export** command is used. Ex:
export command

$ ai=greatlearning.in

$ echo $ai

greatlearning.in

$ export ai

$ bash

$ echo $ai

Greatlearning.in

$

In line three, we used the **export** command to make the variable “ai” to be exported whenever a new child process is created. Thus, the variable “ai” still contains the string “greatlearning.in” even though a new bash shell was created. It is worth mentioning that in order to export the variable “ai” to be available in a new process, the process must be forked from the parent process where the actual variable was exported. The relationship between the child and parent process is detailed below.

Child vs Parent process

Any process is often a parent and child process at an equivalent time. But one exception is that the **init** process, which is usually marked with PID (process ID) 1.
Therefore, **init** may be a parent of all processes running on your Linux system.

$ ps –p 1

PID TTY TIME CMD

1? 00:00:03 init

Any process created will normally have a parent process from which it was created and will be considered as a child of this parent process. example:

$ echo $$

27860

$ bash

$ echo $$

28030

$ ps – -ppid 27860

PID TTY TIME CMD

28030 pts/2 00:00:00 bash

Line 1: print a PID for a current shell – 27860

- Line 2: A new child process created from the process ID 27860
- Line 3: Display a PID for a current shell – 28030
- Line 4: With use of the **ps** command print the child process of PID 27860

While creating a new child process an **export** command ensures that any exported variables in the parent process are also available in the child process.

Using export command

Now that we've learned some basics, we will still need to explore the **export** command comprehensively. When using the **export** command with no option and arguments, it'll simply print all names marked for export to a toddler process. This is the equivalent when using the -p option:

$ export

declare –x COLORFGBG =”15;0”

declare –x DEFAULTS_PATH=”/usr/share/gconf/cinnamon.default.path”

declare -x DESKTOP_SESSION=”cinnamon”

declare -x DISPLAY =”:0”. . . . .

As shown before, to export a variable we just use the variable's name as an argument to an export command.

$ MYVAR =10

$ export | grep MYVAR

$ export MYVAR

$ export | grep MYVAR

declare –x MYVAR=”10”

As you'll see, once the MYVAR variable is exported it will show up within the list of exported variables (line 4). The above example are often shortened by using the **export** command directly with variable assessment.

$ export MYVAR =10

$ export | grep MYVAR

Declare –x MYVAR ="10"

The commonly **export** command is when defining the /PATH shell variable:

export PATH =$PATH:/usr/local/bin

In the example above, we' ve included an additional path /usr/local/bin to the prevailing PATH definition.

Exporting a shell function

With the choice of the **export** command can be implemented to export functions. In the example below, we will create a new bash function called ***printname,*** which can simply use the **echo** command to print the string "greatlearning.in".

$ printname () { echo "greatlearning.in" }

$ printname

greatlearning.in

$ export –f printname

$ bash

$ printname

greatlearning.in

Removing names from export list

In the above example, we now have the MYVAR variable defined in our export list.

$ export | grep MYVAR

Declare –x MYVAR =”10”

To delete this variable from the export list we should use the -n export option.

$ export | grep MYVAR

Declare –x MYVAR=”10”

$ export –n MYVAR

$ export | grep MYVAR

$

Linux commands list

Command	Description
Adduser	Add a new user
Aspell	The ***aspell*** command used to spell check on a text file.
Basename	basename command used to strip off components from filenames that aren’t required.
Bzip2	This command is used to create compressed file archives in bzip2 format.
Cpulimit	It is a tool that limits the CPU usage of a process. It is also useful to control batch jobs if we don’t want to eat too many CPU cycles.
arch	Print machine architecture
Awk	Find and Replace text within file(s)
Bc	An arbitrary precision calculator language
cal	Display a calendar
cat	Concatenate files and print on the standard output
chdir	Change working directory

chgrp	Change the group ownership of files
chkconfig	Tool for maintaining the /etc/rc[0-6].d directory hierarchy
chmod	It will modify the access permissions of files and directories
chown	Change the user and group ownership of files
chroot	Change root directory
cksum	Print CRC checksum and byte counts
clear	Clear terminal screen
cmp	Compare two files
comm	Compare two sorted files line by line
cp	Copy one/ more files from line location to another
Csh	The ***csh*** command is used to move between Linux user shells.
cron	Daemon to execute scheduled commands
crontab	Is used to run list of commands at regular schedule
csplit	Split a file into context-determined pieces
cut	Divide a file into several parts
curl	used to download files from the internet by HTTP/HTTPS.
Dig	used to query DNS servers and for resolving DNS records.
date	Display or change the date & time
dc	Desk Calculator
dd	Data Dump – Convert & copy a file
df	Display free disk space
diff	Display the differences between two files
diff3	Show differences among three files
dir	Briefly list directory contents
dircolors	Colour setup for `ls'

dirname	Convert full pathname to a path
du	Estimate file space usage
echo	Display message on screen
ed	A line-oriented text editor (edlin)
egrep	pattern searching command that match an extended expression
eject	Eject CD-ROM
env	Display, set, or remove environment variables
expand	Convert tabs to spaces
expr	Evaluate expressions
factor	Print prime factors
false	Do nothing, unsuccessfully
fdformat	Low-level format a floppy disk
fdisk	Partition table manipulator for Linux
fgrep	Used for to search fixed character string in file(s)
find	Search/locate for files that based a desired criteria
fmt	Reformat paragraph text
fold	Wrap text to fit a specified width
format	Format disks or tapes
free	Display memory usage
fsck	Filesystem consistency check and repair
gawk	Find and Replace text within file(s)
grep	Used for Searching plaintext data sets that match a given pattern
groups	Print group names a user is in
gzip	Compress or decompress named file(s)
head	Output the first part of file(s)

hostname	Print or set system name
id	Print user and group ids
info	Help info
install	Copy files and set attributes
join	Join lines on a common field
kill	Stop a process from running
less	Display output one screen at a time
ln	Make links between files
locate	Find files
logname	Print current login name
lpc	Line printer control program
lpr	Off line print
lprm	Remove jobs from the print queue
ls	List information about file(s)
man	Help manual
mkdir	Create new folder(s)
mkfifo	Make FIFOs (named pipes)
mknod	Make block or character special files
more	Display output one screen at a time
mount	Mount a file system
mv	Move or rename files or directories
nice	Used to start process with specified value
nl	Number lines and write files
nohup	Run a command immune to hang-ups
passwd	Modify a user password

paste	Merge lines of files
pathchk	Check file name portability
pr	Convert text files for printing
printcap	Printer capability database
printenv	Print environment variables
printf	Format and print data
ps	Process status
pwd	Print Working Directory
Ping	Used to check connection to server
quota	Display disk usage and limits
quotacheck	Scan a file system for disk usage
quotactl	Set disk quotas
ram	ram disk device
rcp	Copy files between two machines
rm	Remove files
rmdir	Remove folder(s)
rpm	Remote Package Manager
rsync	Remote file copy (Synchronize file trees)
screen	Terminal window manager
sdiff	Merge two files interactively
sed	Stream Editor
select	Accept keyboard input
seq	Print numeric sequences
shutdown	Shutdown or restart linux
sleep	Delay for a specified time

sort	Sort text files
split	Split a file into fixed-size pieces
su	Substitute user identity
sum	Print a checksum for a file
symlink	Make a new name for a file
sync	Synchronize data on disk with memory
tac	Concatenate and write files in reverse
tail	Output the last part of files
tar	Tape Archiver
tee	Redirect output to multiple files
test	Evaluate a conditional expression
time	Measure Program Resource Use
touch	Change file timestamps
top	List processes running on the system
traceroute	Trace Route to Host
tr	Translate, squeeze, and/or delete characters
true	Do nothing, successfully
tsort	Topological sort
tty	Print filename of terminal on stdin
umount	Unmount a device
Umask	Command for user files creation mask
uname	Print system information
unexpand	Convert spaces to tabs
uniq	Uniquify files

units	Convert units from one scale to another
unshar	Unpack shell archive scripts
useradd	Create new user account
usermod	Modify user account
users	List users currently logged in
uuencode	Encode a binary file
uudecode	Decode a file created by uuencode
vdir	Verbosely list directory contents (`ls -l -b')
watch	Execute/display a program periodically
wc	Print byte, word, and line counts
whereis	Report all known instances of a command
which	Locate/Search a program file in the user's path
who	Display all usernames that currently logged in
whoami	Display the current user ID and name
Users	Display user names of currently logged in
Userdel	Command deletes user account
Unalias	Delete an alias
xargs	Execute utility & passing construct argument list(s)
yes	Display a string until interrupted
What is	Shows/search online manual page description
Wget	Used to download files from web
Wall	Used to send or display message to all users
Youtube-dl	Used to download videos
zcmp/zdiff	Used to compare compressed files
Zz	This commands offers quick access to files and directories

Linux commands with examples

1. ls

$ ls

This command shows a list of all the files and directories present in the current working directory.

$ ls ~

This command shows the list of files that are present in the home directory.

$ ls -ltr

This command will display the file having the name mentioned in the command and will give details of that file.

$ ls –ltr

This command will list all files in the order of time in which they were created. The terms "ltr" stand for l: long listing, t : time, r: recursive.

2.df

$ df

"df" is a "disk file system". This command shows a summary list of the total disk space available and the used disk space on the
file.

$ df -h

'-h' parameter is used to display the summary of the total disk space used and available on your Linux file system in MB/GB.

3.mkdir

$ mkdir

Used to create a new directory on Linux file system. For Ex:- mkdir greatlearning will create a new directory named “greatlearning”.

4.rm

$ rm

Used to remove a file from your Linux file system.

For example- “rm data”, this will remove file named data present in the current working directory.

$ rmdir

Used to remove a complete directory from the Linux file system.

For example- “rmdir allstuffs” will delete the allstuffs directory from the Linux file system.

5.pwd

$ pwd

“pwd” stands for “Present Working Directory”. It will display the directory name on which you are currently present or working and also shows the path of that directory.

6. cd

$ cd

“cd” stands for change directory. By this command, a user can easily navigate to the directory of his choice at any moment of time. cd command changes the present working directory, to the directory name specified by the user in which he wants to navigate.

$ cd /

This is used to navigate to the root directory of a user’s Linux file system.

$ cd ~

This command is used to go to the home directory.

$ cd..

This command is used to navigate to one directory level up to the current working directory.

$ cd –

This command is used to navigate to the previous directory or simply go one directory back to the directory which the user visited.

7.clear

$ clear

This command will clear all the data present on the Linux terminal window.

8. mv

$ mv

This command is used to change the name of a file/directory or to move a particular file/directory from one place to another.

9. cp

$ cp

This is used to copy files/directory from one place to another. The user can easily create multiple copies of a files/directories using this command.

10. cat

$ cat

Displays all the contents of a file on the output device.

11. du

$ du

"du" stands for "Disk Utility". This command displays the details about how much space is occupied by a file/directory in the disk.

$ du -sh

("-s"= Summary and "-h"= Human Readable). It shows the details of the space occupied by a file/directory on disk in bytes, megabytes, gigabytes, etc.

12. touch

$ touch

This command creates an empty file with the specified name in the current working directory. An empty file with size 0 bytes gets created and will not be changed until the user makes any changes to it.

13. who

$ who

It displays the number of users who are currently logged on Linux OS.

14. echo

$ echo

displays text written after the word "echo".

Ex.- echo My House name is Sweet Dreams, this command output will be: "My House name is Sweet Dreams".

15. date

$ date

Displays the current date and time of your system. We easily know the current day and time on the Linux terminal using this command.

16 .gzip

$ gzip filename

This command compresses content of files, gives extension of .gz and needs to be uncompressed before use.

17. touch

$ touch filename

It creates new empty files and also used to change date and time of recent access and modification.

18. locate

$ locate filename

Used to search find files by name and it also searches very fast, runs in the background to trace the location of files. It also searches the file and stores them in a database.

19. echo

$ echo String

This command writes arguments to the standard output or display line of text. This command is used in batch and scripts. This also plays an important role in building shell script.

20. grep

$ grep “String” filename

This command searches the text or file for lines containing a match to the given strings/words.

$ grep “String” filename1 filename2

This command searches the mentioned String in multiple files.

21. clear

$ clear

This command clears the screen.

22. logout

$ logout

This command is used to exit a login shell or to get out of a current session.

23. exit

$ exit

This command is to exit a shell, like log out. It also reminds us that some jobs are running in the background.

24. wc

$ wc [options] filename

This command is used to print the number of newlines, word, bytes in a file.

$ wc -l filename

This command is used to print the number of lines in a file.

$ wc -w filename

This command is used to print the number of words in a file.

$ wc -c filename

This command is used to display the count of bytes of a file.

25. sort

$ sort filename

This command sorts the file in alphabetical order and is used for printing lines of input text files and concatenation of all files in sorted order.

$ sort -u filename

-u in sort command removes duplicate records in a file and only the first record is retained.

$ sort -n file

This command sorts a file numerically.

26. kill

$ kill

It stops a process and kills or terminates a process without logging out or restarting the system.

27. ps

$ ps

It will display current running process in the system and also used for view process running on system. It provides information on current processes like CPU usage, user id, command name and memory usage.

$ ps -ax

It shows all current running processes.

28. uptime

$ uptime

This command shows time from which system has been running. It will show current time of system, No.of users that are currently active and system load average.

29. sleep

$ sleep number [suffix]

These command delays or pauses for a specified amount of time. It actually causes the system with a given amount of time and also suspends the system for mentioned duration, then turns on or resumes.

$ sleep (n)

Sleep for (n) seconds.

$ sleep (nm)

Sleep for (n) minutes.

$ sleep (nh)

Sleeps for (n) hours.

$ sleep (nd)

Sleeps for (n) days.

30. seq

$ seq n

Prints numbers starting from 1 to n and also prints the numbers from first to last with an increment. All numbers can be real, not just integers.

$ seq n1 n2

Prints number starting from n1 to n2.

This brings us to the end of the Linux Tutorial. We hope that you found this helpful and were able to learn some valuable information from the same. If you wish to learn more such concepts, you can check out the Free Online Courses available on Great Learning Academy.

Sharing is caring:

Great Learning Team