# Introduction

## Introduction

Ubuntu is a Linux operating system based on Debian, and is available for Desktops and Servers.

The default file manager for Ubuntu is GNOME Files, also called Nautilus, and the default desktop has been GNOME since version 17.10. Of course, it is possible to use other desktops as well.

Since Ubuntu is based on Debian, it uses APT as package manager (among others). APT is used to automatically install, update/upgrade, or remove programs from the system, as well as manage their dependencies by installing packages when required, and uninstalling them when no program are using them anymore.

A new version of Ubuntu is released every 6 months, with a Long-Term-Support (LTS) version released every 2 years in April. For example, the latest LTS version is 20.04 codename “Focal Fossa”, released on April 2020. LTS versions are supported for 5 years, while others for 9 months.

While Ubuntu is based on Debian, many other operating systems are based in Ubuntu themselves, including Kubuntu (which uses KDE instead of GNOME), Lubuntu (which is lighter), and Ubuntu Studio (which provides audio and video tools).

## Download

ISOs for Ubuntu Desktop and Server can be downloaded for free from the [Ubuntu Downloads Page](https://ubuntu.com/" \l "download).

This document will be using Ubuntu 20.04 Server for reference, but most of the following applies for other versions as well.

# Command Line

## Opening a Terminal

On a Server installation, or using SSH, there is no user interface by default, the terminal is always opened. To install the Desktop, enter the following commands:

|  |  |
| --- | --- |
| $  $  $ | sudo apt update && sudo apt upgrade  sudo apt install ubuntu-desktop  sudo reboot |

You will be prompted for the password the first time. It will appear invisible.

On a Desktop installation, or after installing a desktop as above, a terminal can be opened by right-clicking and choosing “Open in Terminal”, or by using Ctrl+Alt+T.

The rest of the document will be working on the terminal.

## Superuser

Some commands need to be run as a superuser (equivalent to administrator in Windows). For examples, some files and directories cannot be manipulated by anyone except a superuser, as is the case for manipulating network services, firewall rules, etc.

To switch to superuser mode in the terminal only, use the su command. To go back, use the logout command. Superuser mode is dangerous to use since it can affect important files with no going back. It is recommended to use this mode only when necessary and the user knows what they are doing.

For more safety, superuser mode can be run temporarily on a single command before automatically switching back by typing sudo before the rest of the command. Still, it is recommended to only use sudo if the user knows what following the command does.

## Package Management

To install new software or manage existing packages, Ubuntu provides the apt (Advanced Package Tool) command, which needs superuser permissions, as well as the more secure snap.

The following commands are available:

|  |  |
| --- | --- |
| apt update | Updates the database of available packages.  Recommended to execute before installing and upgrading. |
| apt install *<package>* | Installs a package, or upgrades it if it is already installed. |
| apt upgrade *[<package>]* | Upgrades (updates) the packages if a new version is available. |
| apt remove *<package>* | Removes (uninstall) a package. |
| apt purge *<package>* | Removes a package and its configuration files. |
| apt autoremove | Removes packages that are no longer used. |
| apt show *<package>* | Shows information about a package. |
| apt search *<term>* | Lists all packages containing the search term. |
| apt list | Lists all installed packages.  Use with --upgradable to see packages with new versions available. |
| snap install *<package>* | Installs (or downloads and install) a Snap. |
| snap remove *<package>* | Removes (uninstall) a Snap. |
| snap info *<package>* | Shows information about a Snap. |
| snap find *<term>* | Lists all Snaps containing the search term. |
| snap list | Lists all installed Snaps. |

APT gets the package repository information used in the update command from the etc/apt/sources.list file and files and the files in the /etc/apt/sources.list.d directory. These contain lines starting with deb or deb-src like so:

|  |
| --- |
| deb http://lb.archive.ubuntu.com/ubuntu focal main restricted  deb-src http://lb.archive.ubuntu.com/ubuntu focal-updates universe |

Lines starting with deb mean the following repository distributes compiled packages (in .deb file format), while those starting with deb-src mean the following repository distributes the source code of packages (in .deb file format).

After this (in green) is the URL of the repository, where the packages and databases are stored. Then (in red), there is the first part of the distro’s codename (“focal” for 20.04, “bionic” for 18.04, and so on). Finally (in purple), there are the sections of the repository (main, restricted, universe, and multiverse).

The “main” section contains officially supported free and open-source software by Canonical. The “restricted” section contains proprietary supported software that is not available under a free license. The “universe” section contains unofficial free and open-source software by the community. The “multiverse” section contains paid software restricted by copyright.

## Navigation and File Manipulation

To check the current directory in which the terminal is opened, you can look at the prompt. For example, the following means that the terminal is opened in the ~/Desktop/Programs folder shown in blue, by the user called “user” in the computer called “ubuntu” shown in green:

|  |  |
| --- | --- |
| **user@ubuntu:~/Desktop/Programs$** | █ |

The following commands are used for navigation and folder-related activity:

|  |  |
| --- | --- |
| pwd | Prints the current working directory (path). |
| ls | List files and folders in the current working directory |
| cd "*<path>*" | Changes the current directory to a relative path. cd Programs in ~/Desktop will send to ~/Desktop/Programs. |
| mkdir "*<path>*" | Creates a relative directory to the current working directory.  mkdir Programs/A in ~/Desktop will create the folder ~/Desktop/Programs/A. |
| mkdir -p "*<path>*" | Creates a relative path to the current working directory.  mkdir Programs/B/BA will create the folders Programs/B and Programs/BA. |
| mv "*<dir1>*" "*<dir2>*" | If dir2 exists, moves dir1 into dir2. Else, renames dir1 to dir2. |
| cp *[*-r*]* "*<dir1>*" "*<dir2>*" | If dir2 exists, copies dir1 and names it dir2. Else, copies dir1 inside dir2.  Use -r option for folders. |
| rm "*<path>*" | Deletes a file. |
| rmdir -r "*<path>*" | Deletes a folder and its contents. |

The following are some “special” paths:

|  |  |
| --- | --- |
| cd / | Absolute path: Root directory. |
| cd /home | Absolute path: Home directory. |
| cd ~ | Absolute path: User directory.  Same as cd /home/<username>. |
| cd . | Relative path: Current directory.  Does nothing on its own. |
| cd .. | Relative path: Parent directory.  Goes one folder back. |

## Reading and Writing Files

|  |  |
| --- | --- |
| *<command>* > "*<filename>*" | Saves the output of the command in a file. |
| cat "*<filename>*" | Outputs the content of a file. |
| more "*<filename>*" | Outputs the content of a file with down pagination.  Scroll down by pressing the “Enter” key. |
| less "*<filename>*" | Outputs the content of a file with pagination.  Navigate using arrow keys, quit with q. |

Unlike in Windows, File and directory names are case-sensitive: file.txt, File.txt, and file.TXT are all different files.

### Nano

Nano is a terminal text editor. It usually comes pre-installed, but can be installed with sudo apt install nano.

To open a file using Nano, use nano "*<filename>*". This will open the editor, with available commands shown on the bottom, prefixed by a caret (^) representing the Ctrl key, or by the letter M representing the Alt key.

These are some of the available commands:

|  |  |
| --- | --- |
| Go to line | Ctrl+\_ - line number - Enter |
| Search | Ctrl+W - search term - Enter |
| Next match | Alt+W |
| Search & Replace | Ctrl+W - search term - replace with - Enter |
| Confirm replace | Y | N |
| Replace all | A |
| Select text | Move to beginning of text - Alt+A - Move to end of text |
| Cancel selection | Ctrl+6 |
| Copy | Alt+6 |
| Cut | Ctrl+K |
| Paste | Move to location - Ctrl+U |
| Save | Ctrl+S |
| Save as | Ctrl+O |
| Exit | Ctrl+X |
| Get help | Ctrl+G |

Other commands are available in the [Nano Documentation](https://www.nano-editor.org/dist/latest/cheatsheet.html).

### Vim

Vim is another terminal text editor. It usually comes pre-installed, but can be installed with sudo apt install vim.

To open a file using Vim, use vim "*<filename>*". This will open the editor in the normal mode. To switch to insert mode, type i. To switch to visual mode, type v. To switch to command-line mode, type :. To switch to normal mode, press Esc.

In normal and command-line modes, commands will appear on the bottom. These are some of the available commands:

|  |  |
| --- | --- |
| Open file | :edit "*<filename>*" |
| Go to line | :*<number>* |
| Search | :/"*<search\_term>*" |
| Next match | n |
| Select text | Move to beginning of text - v - Move to end of text |
| Cancel selection | Ctrl+6 |
| Copy | y |
| Cut | Ctrl+K |
| Paste | p |
| Save | :w |
| Exit | :q |
| Get help | :help *[<topic>]* |

Other commands are available in the [Vim Documentation](https://vimhelp.org/).

## Services

Systemctl, ufw

## Downloading Files

curl and wget are two command-line utilities used to download files using HTTP and FTP, with support for HTTP GET and POST requests.

### Wget

Wget can handle large downloads, recursive downloads, and multiple file downloads. While downloading, Wget will print a progress bar as well as the amount of bytes downloaded, the download speed, and the ETA.

To download a single file, simply pass its URL (HTTP or FTP) to the wget command:

|  |  |
| --- | --- |
| $ | wget "*<url>*" |

This will download the file with its default name given by its uploader (the last part of the URL). To rename the file, use the -O option followed by the desired name:

|  |  |
| --- | --- |
| $ | wget "*<url>*" -O "*<name>*" |

To download multiple files, a text file needs to be created first with one URL per line, which Wget will read line-by-line and download when using the -i option followed by the file name:

|  |  |
| --- | --- |
| $ | wget -i "*<filename>*" |

Wget lets the user download a whole website and its necessary files (except those mentioned by --reject) to a directory specified by the -P option :

|  |  |
| --- | --- |
| $ | wget "*<url>*" --mirror --convert-links -P "*<path>*" -p --reject="*<extension>*" |

Wget allows downloading only file with specific types from a webpage using -r -A. followed by the file extension to download:

|  |  |
| --- | --- |
| $ | wget "*<url>*" -r -A."*<extension>*" |

When downloading via FTP, the connection is anonymous, but Wget allows to authenticate using a username and a password:

|  |  |
| --- | --- |
| $ | wget "*<ftp\_url>*" --ftp-username="*<username>*" -ftp-password="*<password>*" |

To POST data to a URL, the --post-data flag should be specified, followed by a list of parameters (e.g. "param1=value1&param2=value2"):

|  |  |
| --- | --- |
| $ | wget "*<url>*" --post-data "*<parameters>*" |

Parameters can also be written in a file, with each key-value pair on a line. The command stays the same, with the file path specified instead of the parameters:

|  |  |
| --- | --- |
| $ | wget "*<url>*" --post-data "*<filename>*" |

Wget also supports resuming downloads that have been interrupted using the -c option, limiting the download speed using --limit-rate=*<rate>*, masking as a browser using --user-agent=*<agent>*,

### Curl

Curl provides more functionalities than Wget, like APIs that can be used inside code. It also supports protocols unsupported by Wget, including TELNET, FILE, SFTP.

While downloading, Curl will print the percentage of completion as well as the size of the download, the amount of bytes downloaded, the download and upload speeds, the ETA, and others.

To download a single URL, simply pass its URL (HTTP or FTP) to the curl command:

|  |  |
| --- | --- |
| $ | curl "*<url>*" |

This will download the file in the terminal itself - Useful when the URL contains small JSON text only or the like used in GET requests. To download the file, use the -O option:

|  |  |
| --- | --- |
| $ | curl "*<url>*" -O |

This will download the file with its default name given by its uploader (the last part of the URL). To rename the file, use the -o option followed by the desired name:

|  |  |
| --- | --- |
| $ | curl "*<url>*" -o "*<name>*" |

To POST data to a URL, -X POST should be specified, as well as headers using -H (multiple can be specified using -H multiple times), and optional parameters (e.g. "param1=value1&param2=value2"):

|  |  |
| --- | --- |
| $ | curl -X POST "*<url>*" *[*-H "*<header>*"*]* *[*-d "*<parameters>*"*]* |

# Software

## Nginx

## Introduction

Nginx (pronounced “engine X”) is a free and open source software used as a web server, load balancer, reverse proxy, mail proxy, cache, and more.

Nginx can handle thousands of simultaneous users while using low memory, auto-indexing, static files, IPv6, WebSockets, rewrites and redirects, and more.

Nginx was built with the goal of having a better performance than Apache. It uses a lot less memory and can handle 4 time more requests per second, at the expense of being less flexible.

### Installation

First, create a file /etc/apt/sources.list.d/nginx.list with the following content:

|  |
| --- |
| deb https://nginx.org/packages/ubuntu/ focal nginx  deb-src https://nginx.org/packages/ubuntu/ focal nginx |

If Apache is running, stop it with systemctl stop apache2.Then, to install Nginx, execute the following:

|  |  |
| --- | --- |
| $  $ | sudo apt update  sudo apt install nginx |

If the following warning appears,

|  |  |
| --- | --- |
|  | W: GPG error: https://nginx.org/packages/ubuntu focal InRelease: The following signatures couldn't be verified because the public key is not available: NO\_PUBKEY $key |

execute these commands while replacing $key with the key given in the warning:

|  |  |
| --- | --- |
| $  $  $ | sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys $key  sudo apt update  sudo apt install nginx |

To start Nginx so that it can serve web pages, execute the following:

|  |  |
| --- | --- |
| $  $  $ | sudo fuser -k 80/tcp  sudo fuser 443/tcp  sudo systemctl start nginx |

To make sure Nginx is installed and running, enter the following:

|  |  |
| --- | --- |
| $ | systemctl status nginx |

If you go on localhost in the browser, you should see the Nginx landing page.

## Apache

## Introduction

Apache is a free and open source software used as a web server. It is also known as httpd, short for “HTTP daemon”. It can be also used as load balancer, reverse proxy, and more.

It has many modules including authentication, support for languages such as Python, Perl, and mainly PHP, and handling of auto-indexing, static files, IPv6, WebSockets, rewrites, and more.

Apache is slightly more used than Nginx, but is slower and uses more memory.

## Installation

If Nginx is running, stop it with systemctl stop nginx. Then, to install Apache 2, execute the following:

|  |  |
| --- | --- |
| $  $ | sudo apt update  sudo apt install apache2 |

To allow traffic on port 80 for Apache, allow it on the firewall:

|  |  |
| --- | --- |
| $ | sudo ufw allow 'Apache' |

To start Apache so that it can serve web pages, execute the following:

|  |  |
| --- | --- |
| $ | sudo systemctl start apache2 |

To make sure Apache is installed and running, enter the following:

|  |  |
| --- | --- |
| $ | systemctl status apache2 |

If you go on localhost in the browser, you should see the Apache2 landing page.

## PHP

## Introduction

PHP is an open-source, widely-used, object-oriented, server-side programming language for building interactive web pages and HTTP requests. PHP can be embedded inside HTML, and HTML can be embedded inside PHP.

It can be used to dynamically create HTML elements, query a database, fill and send forms, configure sessions and cookies, basic programming, and more. PHP can also be used outside browsers, on the command-line or in desktop applications with the help of PHP-GTK.

Unlike JavaScript, which is run on the client-side, PHP is run on the server-side, and its output is sent as HTML to the client.

PHP supports most web servers, including Nginx, Apache, and IIS.

## Installation

To install PHP, execute the following:

|  |  |
| --- | --- |
| $  $ | sudo apt update  sudo apt install php |

To install some other modules, run:

|  |  |
| --- | --- |
| $ | sudo apt install php-common php-cli php-json php-xml php-mysql php-curl |

To make sure PHP is installed and to check its version, execute:

|  |  |
| --- | --- |
| $ | php -v |

To access Apache-related configurations for PHP, open /etc/php/*<version>*/apache2/php.ini by replacing “*<version>*” with the major and minor versions only. For example, for PHP 7.4.3:

|  |  |
| --- | --- |
| $ | less /etc/php/7.4/apache2/php.ini |

In this file, line starting with a semicolon are comments and are therefore ignored.

## MySQL

## Introduction

## Installation