

linux terminal user manual

Version

Ivan Moreno Villamil

April 4, 2022

Contents

1	Introduction	3
2	Commands in the shell	3
2.0.1	El Command man of Linux	3
2.0.2	pwd (Print Working Directory)	4
2.0.3	cd (Change Directory)	4
2.0.4	ls (List Directories)	6
2.0.5	touch	7
2.0.6	file	7
2.0.7	cat	7
2.0.8	less	8
2.0.9	history	8
2.0.10	cp (Copy)	9
2.0.11	mv (Move)	10
2.0.12	mkdir (Make Directory)	10
2.0.13	rm (Remove)	10
2.0.14	find	11
2.0.15	alias	11
2.0.16	whatis	12
2.0.17	Command tar	12
2.0.18	git commands	13

List of Figures

1	Command man	4
2	Command Manual ls	4
3	Command man	4
4	Command cd	5
5	Ingreso dentro de carpeta1	5
6	Command ls	6
7	Command ls -a	6
8	Command ls -l	7
9	Command touch	7
10	Command file	7
11	Command cat	7
12	Command less	8
13	Command history	8
14	Command cp	9
15	Command cp *	9

16	Command mv	10
17	Command find	11
18	Command alias	11
19	Command whatis	12
20	Command tar -czvf	12
21	Command tar -cvzf	13
22	Command tar -cvjf	13
23	Command tar -jxvf	13
24	Command git init	14
25	Command git add	14
26	Command git commit	14
27	Command git config	14
28	Command git status	15
29	Command git push	15
30	Command git checkout	15
31	Command git branch	16
32	Command git merge	16
33	Command git log	16
34	Command git rm	16

1 Introduction

What is Linux?

Linux is an operating system, like macOS or Windows. It is also the most popular open source operating system, and it gives you a lot of freedom. It feeds the vast majority of the servers that make up the Internet. It is the foundation on which everything is built. But not only that. Android is based on (a modified version of) Linux.

What is Linux shell?

A shell is a command interpreter that exposes an interface for the user to work with the underlying operating system, allowing us to execute operations using text and commands, and provides users with advanced features such as the ability to create scripts.

In this manual you will find basic information on the basic commands used to manage the terminal and gain access to:

- Commands to navigate through the different directories
- Commands to list the contents of a directory, search for files, etc.
- Commands to create, delete, copy and move files and directories
- More information

2 Commands in the shell

2.0.1 El Command man of Linux

This Command allows me to understand all the Commands, by giving me a description of each of them, it works like the manual inside the console, so, Every time I don't know how to use a Command, I write **man <command>** to get the manual:

Figure 1: Command man

```
ivan@ivan-VirtualBox: ~  
ivan@ivan-VirtualBox:~$ man  
¿Qué página de manual quiere?  
Por ejemplo, pruebe 'man man'.  
ivan@ivan-VirtualBox:~$ man ls
```

Figure 2: Command Manual ls

```
LS(1) User Commands LS(1)  
NAME  
ls - list directory contents  
SYNOPSIS  
ls [OPTION]... [FILE]...  
DESCRIPTION  
List information about the FILES (the current directory by default).  
Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-  
fied.  
Mandatory arguments to long options are mandatory for short options  
too.  
-a, --all  
do not ignore entries starting with .  
-A, --almost-all  
do not list implied . and ..  
Manual page ls(1) line 1 (press h for help or q to quit)
```

2.0.2 pwd (Print Working Directory)

When you feel lost in the file system, call the Command **pwd** to know where you are, this Command allows me to know in which directory I am:

Figure 3: Command man

```
ivan@ivan-VirtualBox:~/Descargas$ pwd  
/home/ivan/Descargas
```

2.0.3 cd (Change Directory)

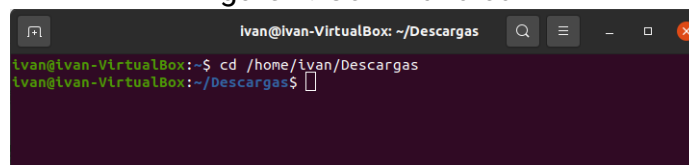
Remember that we will have to navigate our way using paths. There are two different ways to specify a path, with absolute and relative paths.

- Absolute Path: This is the path from the root directory. The root directory is commonly displayed as a forward slash. Whenever your path starts with / it means you are starting from the root directory. For example, /home/ivan/Downloads.
- Relative Path – This is the path from where you are currently in the file system. If I was in the location /home/ivan/Downloads and I wanted to get to a directory inside Downloads called tax, I don't have to specify the full path from the root like /home/ivan/Downloads/tax, I can just go to tax/ in its place.

Knowing the above, we can use the Command cd, which allows us to change directories.

So now I changed my directory location to /home/ivan/Downloads.

Figure 4: Command cd

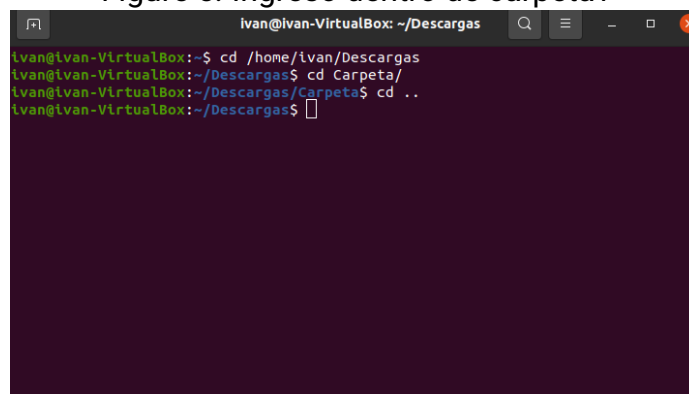


```
ivan@ivan-VirtualBox: ~/Descargas
ivan@ivan-VirtualBox:~$ cd /home/ivan/Descargas
ivan@ivan-VirtualBox:~/Descargas$
```

Now from this directory I have a folder inside called Carpeta1, I can navigate to that folder with:

cd Carpeta1/

Figure 5: Ingreso dentro de carpeta1



```
ivan@ivan-VirtualBox: ~/Descargas
ivan@ivan-VirtualBox:~/Descargas$ cd Carpeta/
ivan@ivan-VirtualBox:~/Descargas/Carpeta$ cd ..
ivan@ivan-VirtualBox:~/Descargas$
```

Notice how I just used the name of folder1? It's because it was already in /home/ivan/Descargas.

It can be quite tiring to navigate with absolute and relative paths all the time, luckily there are some shortcuts to help you out.

- `cd .` (current directory). This is the directory you are currently in.
- `cd ..` (parent directory). Takes you to the directory above your current one.
- `cd -` (previous directory). This will take you back to the previous directory you were in.

2.0.4 ls (List Directories)

The Command `ls` will display a list of directories and files in the current directory by default, however you can specify which path you want to list the directories from.

`ls` is quite a useful tool, it also shows you detailed information about the files and directories you are viewing.

Figure 6: Command `ls`

```
ivan@ivan-VirtualBox:~/Descargas$ pwd
/home/ivan/Descargas
```

Also note that not all files in a directory will be visible. File names starting with `.` are hidden, you can see them however with Command `ls` and pass the `-a` flag (a for all).

Figure 7: Command `ls -a`

```
ivan@ivan-VirtualBox:~$ ls -a
.      .bashrc  .config  file.py  Ivan     Música  Videos
..     books    Descargas  files    .lesshtQ Plantillas
.bash_history  borrar  Documentos .gnupg   .local   .profile
.bash_logout   .cache  Escritorio Inágenes .mozilla Público
```

There is also a more useful `ls` flag, `-la` for long, which displays a detailed list of files in a long format. This will show you detailed information, starting from the left: file permissions, number of links, owner name, owner group, file size, last modified timestamp, and file/directory name.

Figure 8: Command ls -l

```
ivan@ivan-VirtualBox: ~$ ls -la
total 88
drwxr-xr-x 17 ivan ivan 4096 mar 26 17:26 .
drwxr-xr-x  3 root root 4096 mar 24 22:02 ..
-rw-r----- 1 ivan ivan 1656 mar 27 19:09 .bash_history
-rw-r--r--  1 ivan ivan  220 mar 24 22:02 .bash_logout
-rw-r--r--  1 ivan ivan 3771 mar 24 22:02 .bashrc
drwxrwxr-x  3 ivan ivan 4096 mar 26 14:57 books
-rw-rw-r--  1 ivan ivan   0 mar 26 16:12 borrar
drwxr-xr-x 15 ivan ivan 4096 mar 28 17:03 .cache
drwx----- 11 ivan ivan 4096 mar 28 18:07 .config
drwxr-xr-x  3 ivan ivan 4096 mar 28 17:58 Descargas
drwxr-xr-x  3 ivan ivan 4096 mar 26 21:45 Documentos
drwxr-xr-x  2 ivan ivan 4096 mar 24 23:57 Escritorio
-rw-rw-r--  1 ivan ivan   0 mar 26 15:18 file.py
-rw-rw-r--  1 ivan ivan   18 mar 26 15:19 filess
drwx----- 3 ivan ivan 4096 mar 24 23:57 .gnupg
drwxr-xr-x  2 ivan ivan 4096 mar 28 18:08 Imágenes
drwxrwxr-x  2 ivan ivan 4096 mar 25 17:24 Ivan
-rw-rw-r--  1 ivan ivan   0 mar 26 17:26 .lessshsQ
drwxr-xr-x  3 ivan ivan 4096 mar 24 23:57 .local
drwx----- 4 ivan ivan 4096 mar 25 17:21 .mozilla
drwxr-xr-x  2 ivan ivan 4096 mar 24 23:57 Música
```

2.0.5 touch

This Command allows us to create new empty files.

Figure 9: Command touch

```
ivan@ivan-VirtualBox:~$ touch Nuevo
ivan@ivan-VirtualBox:~$ ls
books  Descargas  Escritorio  filess  Ivan  Nuevo  Público
borrar  Documentos  file.py    Imágenes  Música  Plantillas  Videos
```

2.0.6 file

This Command will allow us to find out what type of file it is, and will show us a description of the file.

Figure 10: Command file

```
ivan@ivan-VirtualBox:~/Descargas$ file Modelado_Tarea_1.pdf
Modelado_Tarea_1.pdf: PDF document, version 1.5
ivan@ivan-VirtualBox:~/Descargas$ file Carpeta/
Carpeta/: directory
```

2.0.7 cat

A simple command to use is the cat command, short for concatenate, it not only displays the contents of the file, but can also combine multiple files and display the output from them.

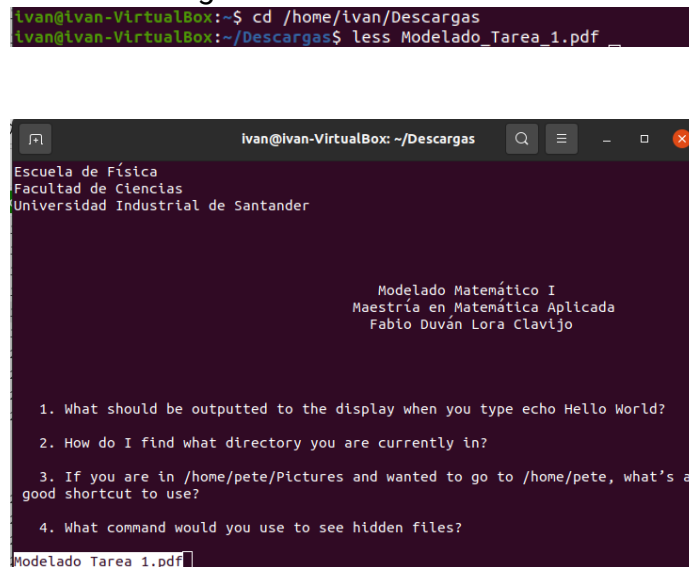
Figure 11: Command cat

```
ivan@ivan-VirtualBox:~$ cat Música/
cat: Música/: Es un directorio
ivan@ivan-VirtualBox:~$ cat Descargas/
cat: Descargas/: Es un directorio
```


2.0.8 less

less is a Command that allows us to display text files, which allows us a more dynamic display allowing us to move forward and inside the text file.

Figure 12: Command less



```
ivan@ivan-VirtualBox:~$ cd /home/ivan/Descargas
ivan@ivan-VirtualBox:~/Descargas$ less Modelado_Tarea_1.pdf
```

Escuela de Física
Facultad de Ciencias
Universidad Industrial de Santander

Modelado Matemático I
Maestría en Matemática Aplicada
Fabio Duván Lora Clavijo

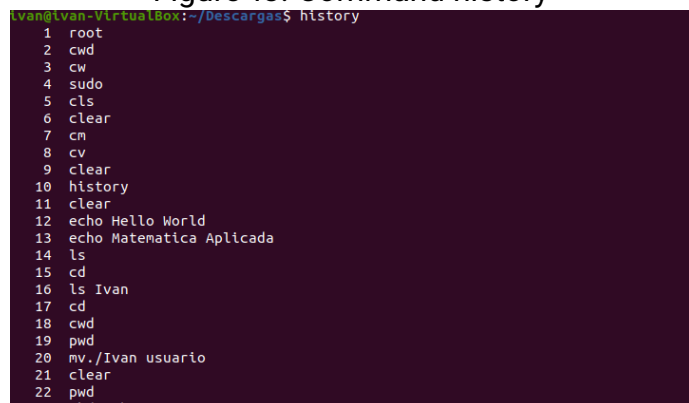
1. What should be outputted to the display when you type echo Hello World?
2. How do I find what directory you are currently in?
3. If you are in /home/pete/Pictures and wanted to go to /home/pete, what's a good shortcut to use?
4. What command would you use to see hidden files?

Modelado_Tarea_1.pdf

2.0.9 history

With the Command history we can access all previously used Commands in the shell console, which is quite useful when you want to find and execute a previously used Command without having to retype it.

Figure 13: Command history



```
ivan@ivan-VirtualBox:~/Descargas$ history
```

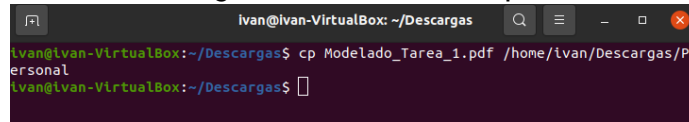
- 1 root
- 2 cwd
- 3 cw
- 4 sudo
- 5 cls
- 6 clear
- 7 cm
- 8 cv
- 9 clear
- 10 history
- 11 clear
- 12 echo Hello World
- 13 echo Matematica Aplicada
- 14 ls
- 15 cd
- 16 ls Ivan
- 17 cd
- 18 cwd
- 19 pwd
- 20 mv ./Ivan usuario
- 21 clear
- 22 pwd
- 23 cd /home/ivan/usuario

To clear history, run history -c.

2.0.10 cp (Copy)

Just like copying and pasting files in other operating systems, the shell gives us an even simpler way to do it.

Figure 14: Command cp

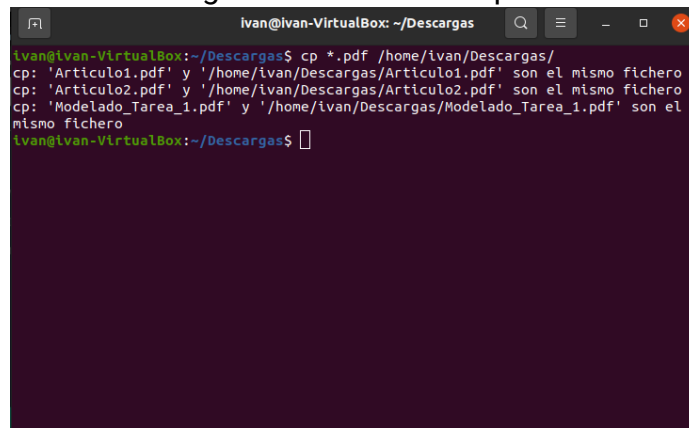


```
Ivan@Ivan-VirtualBox: ~/Descargas
Ivan@Ivan-VirtualBox:~/Descargas$ cp Modelado_Tarea_1.pdf /home/ivan/Descargas/Personal
Ivan@Ivan-VirtualBox:~/Descargas$
```

You can copy multiple files and directories, as well as use wildcards. A wildcard is a character that can be substituted for a pattern-based selection, giving you more flexibility with searches. You can use wildcards in each Command for more flexibility.

- * the wildcard of wildcards, is used to represent all single characters or any string.
- ? used to represent a character
- [] used to represent any character inside the brackets

Figure 15: Command cp *



```
Ivan@Ivan-VirtualBox: ~/Descargas
Ivan@Ivan-VirtualBox:~/Descargas$ cp *.pdf /home/ivan/Descargas/
cp: 'Articulo1.pdf' y '/home/ivan/Descargas/Articulo1.pdf' son el mismo fichero
cp: 'Articulo2.pdf' y '/home/ivan/Descargas/Articulo2.pdf' son el mismo fichero
cp: 'Modelado_Tarea_1.pdf' y '/home/ivan/Descargas/Modelado_Tarea_1.pdf' son el mismo fichero
Ivan@Ivan-VirtualBox:~/Descargas$
```

This will copy all files with the .pdf extension in your current directory to the Downloads directory.

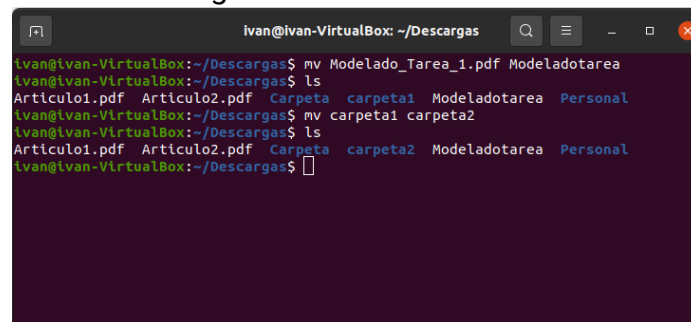
A useful Command is to use the -r flag, this will recursively copy the files and directories within a directory.

2.0.11 mv (Move)

This Command is used to move files and also to rename them. You can rename the files like this:

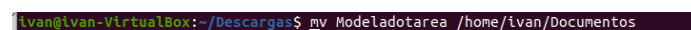
```
mv carpeta1 carpeta2
```

Figure 16: Command mv



```
ivan@ivan-VirtualBox: ~/Descargas
ivan@ivan-VirtualBox:~/Descargas$ mv Modelado_Tarea_1.pdf Modeladotarea
ivan@ivan-VirtualBox:~/Descargas$ ls
Articulo1.pdf  Articulo2.pdf  Carpeta  carpeta1  Modeladotarea  Personal
ivan@ivan-VirtualBox:~/Descargas$ mv carpeta1 carpeta2
ivan@ivan-VirtualBox:~/Descargas$ ls
Articulo1.pdf  Articulo2.pdf  Carpeta  carpeta2  Modeladotarea  Personal
ivan@ivan-VirtualBox:~/Descargas$
```

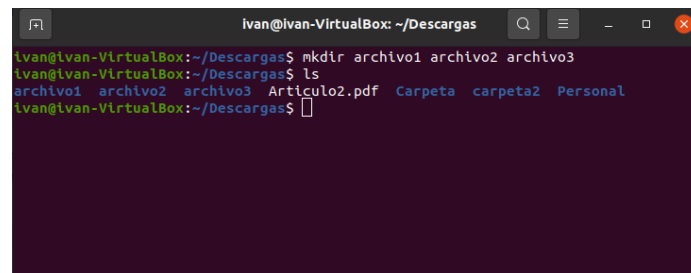
Or you can move one or more files to a different directory



```
ivan@ivan-VirtualBox:~/Descargas$ mv Modeladotarea /home/ivan/Documentos
```

2.0.12 mkdir (Make Directory)

The mkdir (Make Directory) command is useful for creating new directories. You can even create multiple directories at the same time.



```
ivan@ivan-VirtualBox: ~/Descargas
ivan@ivan-VirtualBox:~/Descargas$ mkdir archivo1 archivo2 archivo3
ivan@ivan-VirtualBox:~/Descargas$ ls
archivo1  archivo2  archivo3  Articulo2.pdf  Carpeta  carpeta2  Personal
ivan@ivan-VirtualBox:~/Descargas$
```

You can also create subdirectories at the same time with -p (main flag).

```
mkdir -p fruits/apples
```

2.0.13 rm (Remove)

The Command **rm** (Remove) it is used to remove files and directories.

- **rm -f** Filename. The -f or force option tells rm to remove all files, whether they are write-protected or not, without prompting the user (provided they have the appropriate permissions).
- **rm -i file** Adding the -i flag like many of the other commands will tell you if you want to delete the files or directories.
- **rmdir directory.** You can remove a directory with the Command **rmdir**.

2.0.14 find

The command find can be used to find files or folders that match a given search pattern. Search recursively. An interesting thing to note is that find doesn't stop at the directory it's looking for, it will also search within any subdirectories that directory might have.

Figure 17: Command find

```
ivan@ivan-VirtualBox:~/Documentos$ find
./
./Data science in the Cahn Hilliard model.pdf
./tutorial0.pdf
./python.py
./Modeladotarea
./Myproject
./Myproject/modelado1
./Myproject/modelado1/Data science in the Cahn Hilliard model.pdf
./Myproject/modelado1/.git
./Myproject/modelado1/.git/HEAD
./Myproject/modelado1/.git/description
./Myproject/modelado1/.git/hooks
./Myproject/modelado1/.git/hooks/pre-rebase.sample
./Myproject/modelado1/.git/hooks/pre-push.sample
./Myproject/modelado1/.git/hooks/pre-receive.sample
./Myproject/modelado1/.git/hooks/update.sample
./Myproject/modelado1/.git/hooks/post-update.sample
./Myproject/modelado1/.git/hooks/prepare-commit-msg.sample
./Myproject/modelado1/.git/hooks/applypatch-msg.sample
./Myproject/modelado1/.git/hooks/commit-msg.sample
```

2.0.15 alias

An alias is a (usually short) name that the shell translates to another (usually longer) name or Command. The Alias allow you to define new Commands by substituting a string for the first token of a simple Command, which are created as follows:

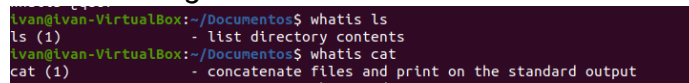
Figure 18: Command alias

```
ivan@ivan-VirtualBox:~/Documentos$ alias
alias alert='notify-send --urgency=low -i "${[ $? = 0 ]} && echo terminal || echo error" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[\;|]\s*alert$//'\''\`)'`
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alf'
alias ls='ls --color=auto'
```

2.0.16 whatis

The Command `whatis` provides a brief description of Command line programs.

Figure 19: Command `whatis`



```
ivan@ivan-VirtualBox:~/Documentos$ whatis ls
ls (1)             - list directory contents
ivan@ivan-VirtualBox:~/Documentos$ whatis cat
cat (1)            - concatenate files and print on the standard output
```

The description is obtained from the manual page of each Command.

2.0.17 Command `tar`

The command `tar` in Linux is often used to create archives of `.tar.gz` or `.tgz` archives, also called "tarballs". This command has a lot of options, but you only need to remember a few letters to quickly create archives with `tar`. The Command `tar` can also extract the resulting files

Use the following Command to compress an entire directory or a single file on Linux. It will also zip all other directories within a directory you specify; in other words, it will work recursively.

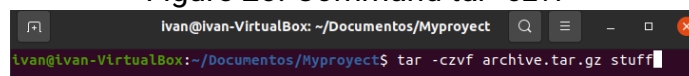
```
tar -czvf filename.tar.gz / path / to / directory-or-file
```

Here's what those switches mean:

- `-c`: create a file.
- `-z`: Compress the file with `g` zip.
- `-v`: View the progress in the terminal while creating the file, also known as "verbose" mode. The `v` is always optional in these Commands, but it is useful.
- `-f`: allows you to specify the `f` filename of the file.

Let's say you have a directory called "stuff" in the current directory and you want to save it to a file called `archive.tar.gz`. I would execute the following Command:

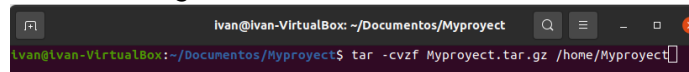
Figure 20: Command `tar -czvf`



```
ivan@ivan-VirtualBox: ~/Documentos/Myproyect
ivan@ivan-VirtualBox:~/Documentos/Myproyect$ tar -czvf archive.tar.gz stuff
```

- Crear un archivo .tar.gz en Linux
Si deseas una mejor compresión, también puedes usar .tar.gz. Un ejemplo de esto es:

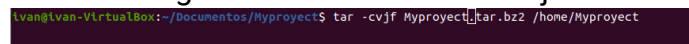
Figure 21: Command tar -cvzf



- The additional option z represents gzip compression.
- Create a .tar.bz2 file on Linux

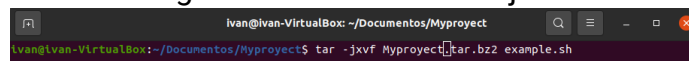
The .bz2 file provides more compression compared to gzip. However, this alternative will take more time to compress and decompress. To use it, you must use the -j option. An example of what the operation would look like is as follows:

Figure 22: Command tar -cvjf



- To extract a single file from a .tar.bz2 archive you can use a Command like this:

Figure 23: Command tar -jxvf



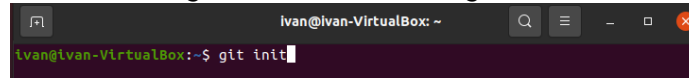
2.0.18 git commands

GIT is the most widely used open source VCS (version control system) that allows you to track changes made to files

- **git init** will create a new local Git repository. The following Git Command will create a repository in the current directory:

- Alternatively, you can create a repository inside a new directory by specifying the project name:

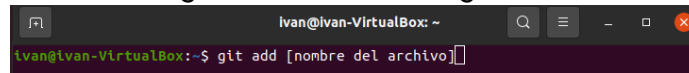
Figure 24: Command git init



`git init [project name]`

- **git add** is used to add files to the staging area. For example, the following basic Git Command will index the temp.txt file:

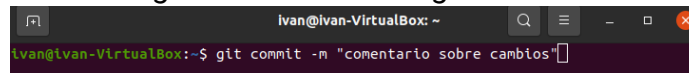
Figure 25: Command git add



`git add <temp.txt>`

- **git commit** will create a snapshot of the changes and save it to the git directory.

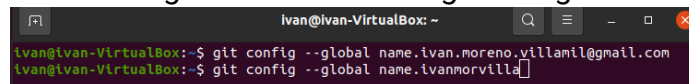
Figure 26: Command git commit



`git commit -m "The message that accompanies the commit goes here"`

- **git config** can be used to set user-specific settings, such as email, username and format type, etc. For example, the following Command is used to establish an email:

Figure 27: Command git config



`git config --global user.email tuemail@ejemplo.com`

- The -global option tells Git that you are going to use that email for all local repositories. If you want to use different emails for different repositories, use the following Command:

`git config --local user.email tuemail@ejemplo.com`

- **git status** displays the list of files that have been changed along with files that are about to be staged or committed.

Figure 28: Command git status

```
ivan@ivan-VirtualBox:~/Documentos$ git status
En la rama master

No hay commits todavia

Cambios a ser confirmados:
  (usa "git rm --cached <archivo>..." para sacar del área de stage)
nuevos archivos: hola.html

Archivos sin seguimiento:
  (usa "git add <archivo>..." para incluirlo a lo que se será confirmado)
Administracion_basica_de_Guadalupe.pdf
data science in the Cahn Hilliard model.pdf
Modeladotarea
Myproyect/
Myproyecto/
puntofijo.py
pyhton.py
pyhton.py.save
secante.py
tutorial0.pdf
```

- **git push** is used to push local commits to the master branch of the remote repository. Here is the basic structure of the code:

Figure 29: Command git push

```
ivan@ivan-VirtualBox: ~/Documentos/Myproyect
ivan@ivan-VirtualBox:~/Documentos/Myproyect$ git push origin <name branch>
```

git push origin <master>

- Replace <master> with the branch you want to push changes to when you don't want to push changes to the master branch.
- **git checkout** creates branches and helps you navigate between them. For example, the following Command creates a new one and automatically switches to it:

Figure 30: Command git checkout

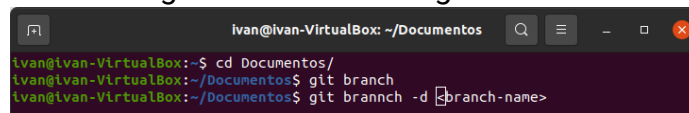
```
ivan@ivan-VirtualBox: ~/Documentos
ivan@ivan-VirtualBox:~/Documentos$ git checkout -b <branch name>
```

command git checkout -b <branch-name>

- To switch from one branch to another, just use:
git checkout <branch-name>

- **git branch** is used to list, create, or delete branches. For example, if you want to list all the branches present in the repository, the Command should look like this:

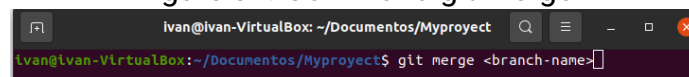
Figure 31: Command git branch



```
ivan@ivan-VirtualBox: ~/Documentos
ivan@ivan-VirtualBox:~$ cd Documentos/
ivan@ivan-VirtualBox:~/Documentos$ git branch
ivan@ivan-VirtualBox:~/Documentos$ git branch -d <branch-name>
```

- To switch from one branch to another, just use:
git checkout <branch-name> item If you want to delete a branch, use:
git branch -d <branch-name>
- **git pull** merges all the changes that have been made in the remote repository with the local working directory.
git pull
- **git merge** is used to merge a branch with another active branch:

Figure 32: Command git merge

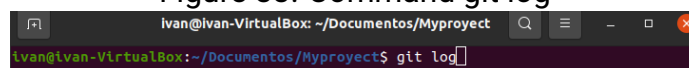


```
ivan@ivan-VirtualBox: ~/Documentos/Myproject
ivan@ivan-VirtualBox:~/Documentos/Myproject$ git merge <branch-name>
```

git merge <branch-name>

- **git log** is used to view the history of the repository listing certain commit details. When executing the Command you get an output like this:

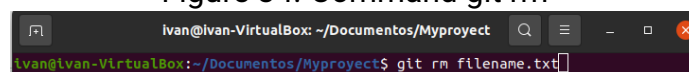
Figure 33: Command git log



```
ivan@ivan-VirtualBox: ~/Documentos/Myproject
ivan@ivan-VirtualBox:~/Documentos/Myproject$ git log
```

- **git rm** can be used to remove files from the index and working directory.

Figure 34: Command git rm



```
ivan@ivan-VirtualBox: ~/Documentos/Myproject
ivan@ivan-VirtualBox:~/Documentos/Myproject$ git rm filename.txt
```

git rm filename.txt