

Terminal Ubuntu :

First of all, the Ubuntu terminal is a terminal that run appart of windows.

So if you want to navigate trough your documents, you need to go to de root and begin by /mnt.

There's a few basics commands that can be useful :

- **cd** (change directory)
- **echo** (allows you to write content in a file > to erase content and add new, » to do a new line)
- **ls** (List of the elements inside a folder) [Equivalent on windows DIR]
- **pwd** (Show the path of the forlder we are) [CD on windows] Print working directory
- **cd ..** (To get back in the path)
- **mkdir** (create a folder at the place)
- **touch** (check if a file with the name exist, if not create one)
- **cat** (cat command allows you to create single or multiple files, view content of a file, concatenate files and redirect output in terminal or files.)
More explication with the command cat, if you use the command cat to create a file, cat > name.file you can write multiple lines in it directly after.
However you need to quit after with ctrl + d (or with ctrl + c but it's not recommanded).
- **mv** (allows you to move a file into a folder you can also use this command to rename a folder if you don't move it the second entrance is the new name) [exemple : mv one.txt story]
- **rm** (allow you to delete a file, you need to confirm that)
- **rm -rf** (delete the directory and all inside)
- **cp** (allows you to copy a file) [exemple : cp one.txt two.txt]
- **dirs / pushd / popd** (pushd allows you to add a path to a stack that will be keep in memory. dirs allows you to see the stack and popd to remove the last path added.)
If we want to add a path to the stack without going to this path, we can type **pushd -n « fullpath »**
You can use the options **-l** and **-v** with dirs to make the appears as an enumerate list.
So you can more easely use the pushd command with indices to go on the path you want.
You can also use popd with an indice to choose the path you want to delete from the stack.
- **ls -s** (allows you to see the owner and group permissions of the files)
To change owner of a file you need the admin privilege.
- **sudo** (Allows you to run command as root)
- **chown** (Allows you to change the owner of a file)
- **chmod** (control who can access file)
- **chgrp** (Allows you to change the group of the file)

} you cannot use **chown**, **chmod**, **chgrp** on windows files when you're in a linux terminal

— For the command **chmod** you can use these options :

Access class	Operator	Access Type
u (user)	+ (add access)	r (read)
g (group)	- (remove access)	w (write)
o (other)	= (set exact access)	x (execute)
a (all: u, g, and o)		

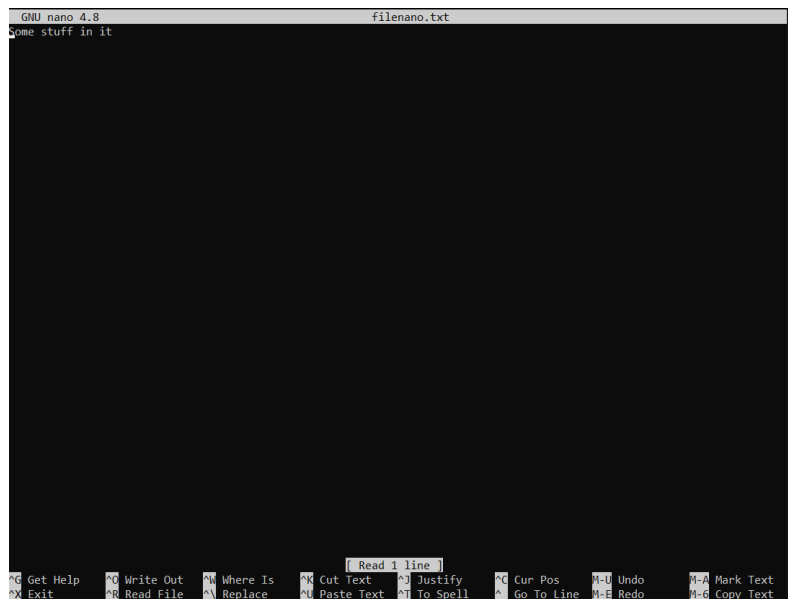
```
william@Polet-William:/$ cd home
william@Polet-William:/home$ ls
william
william@Polet-William:/home$ cd william
william@Polet-William:~$ ls
william@Polet-William:~$ touch file.txt
william@Polet-William:~$ ls -l
total 0
-rw-r--r-- 1 william william 0 May 11 15:36 file.txt
william@Polet-William:~$ sudo chown root file.txt
[sudo] password for william:
william@Polet-William:~$ ls -l
total 0
-rw-r--r-- 1 root william 0 May 11 15:36 file.txt
william@Polet-William:~$ sudo chgrp root file.txt
[sudo] password for william:
william@Polet-William:~$ ls -l
total 0
-rw-r--r-- 1 root root 0 May 11 15:36 file.txt
william@Polet-William:~$ git --version
git version 2.25.1
william@Polet-William:~$ sudo chmod ug+rw file.txt
[sudo] password for william:
william@Polet-William:~$ ls -l
total 0
-rwxrwxr-- 1 root root 0 May 11 15:36 file.txt
william@Polet-William:~$ sudo chmod o-r file.txt
william@Polet-William:~$ ls -l
total 0
-rwxrwx--- 1 root root 0 May 11 15:36 file.txt
william@Polet-William:~$ cat file.txt
cat: file.txt: Permission denied
william@Polet-William:~$ sudo cat file.txt
```

It's normal if there's one line missing that's because the file was empty so the cat command return nothing.

Nano and Vim :

Nano and Vim are text editors that can be used inside the terminal.

To create a file with Nano, type `nano « name.extension »` and then a windows appears.



You can write in it, to save it you need to use `ctrl + O` and press enter to confirm, after you quit with `ctrl + X`

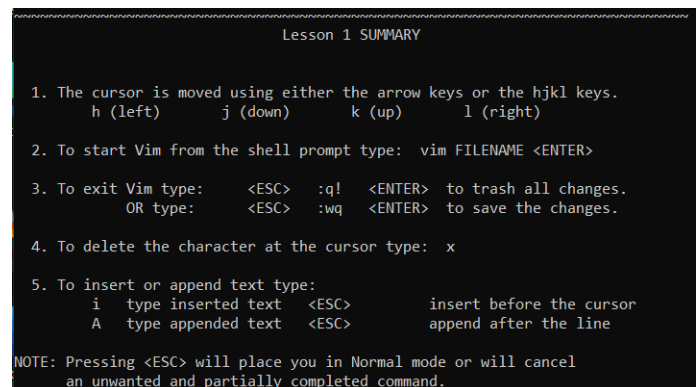
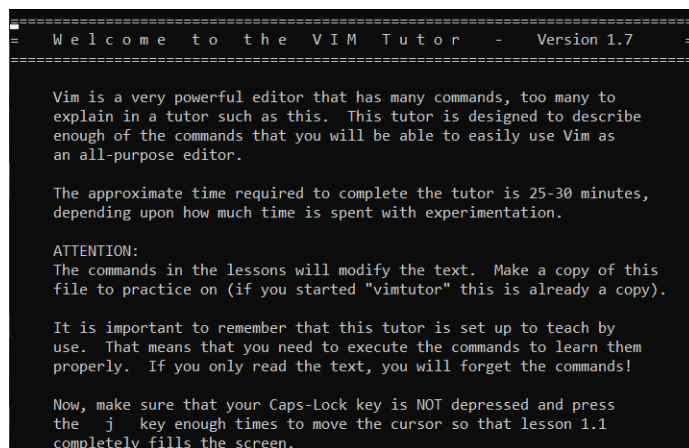
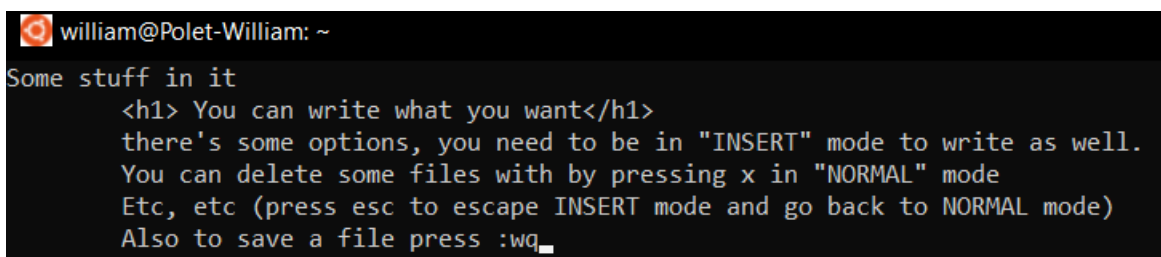
To create a file with Vim use the same way, type `vim « name.extension »`

You will have a window that appears and to **write** in it you need to press `i` to enter in insert mode. When you have finish you can press `esc` to stop editing.

To save the file type `:wq` (to save and exit the file) to only exit the file type `:q!`

There's a lot of things to learn from Vim, for a little preview you can type the command `vimtutor`.

For exemple you can use the command `dw` in normal mode to delete an entire word, or just use the `x` to delete a character.



Markdown Langage :

Markdown is a lightweight language for creating formatted text.

You can do it with any text editor. You just need to respect the syntax and use the extension .md

Few exemple of the syntax :

- # is to make a title. You can put up 1 to 6 # (1 for the largest title and 6 for the smallest)
- To create a paragraph just write, if you want to go to the line, finish your line with 2 spaces.
- To make a list, just do a list who begin with 1. continue with 2. and if you want to do a sublist repeat using 1. with an indentation.
- If you want to instert a gif use and the link of the gif between the parenthesis.

```
william@Polet-William: /mnt/c/users/willi/desktop
# William Polet (student to become a dev junior)

I somebody who like to learn and help people to learn (I was a teacher before).
I like to play video games as well (actually on Dofus there's a time limited server
it's very fun ;p)
And if you want to know more about me, just ask !
## Fun facts about me :
1. I can solve a 3x3 Rubik's cube under 1min (maybe with a little practice, it's be
en a very very long time !) my PB is between 30 and 40 sec, can't remember lol.
2. I really enjoy playing video games but I don't finish half of the games I buy...

3. If you ask me for help, I will help you with pleasure if I'm capable.
## Three things I like the most :
1. My girlfriend of course <3
2. Free time (to play video games or to do sport)
3. Learning new things !
## What I want to achieve at BeCode :
Honestly for now, I don't know. I just want to learn new thins, to change my carree
r. I know that I want to do a backend developper bc frontend is not my cup of tea.
Maybe I will go on JAVA, maybe not. Keep you in touch !
## The fears I might have :
NONE (just kidding, but I don't really have fears so I don't know what answer to th
at)
## The things I look forward too :
Just learning new things and be able to work as a developper because I like it. I d
on't want to do a job that I don't like.
## Three things I value the most to work as a team :
1. Communication, the most important skill !
2. Ask for help if needed, it's never a shame to ask someone's help, the shame is t
o not ask and loose time and energy for nothing.
3. Explain things calmly and clearly so you will not have to repeat the same thing
over ans over.
## My favorite gif :
I don't have a favorite gif, not a huge fan of gifs.
By the way, how do you pronounce "gif" ?
If I had to put a gif it will be this one maybe :

For those who wants to know I was looking for a gif about my favourites games, but
could not find one that I really liked, so I put one of the "Cra" Goddess for Dofus
/Wakfu.

~
"Markdown.md" [dos] 29L, 2081C                               29,173      All
```

And you can see the result by clicking on [this link](#).

Git and Github :

Github is a service to store your code, git it's the technology that has two purpose. First one is to connect to the services (Github or BitBucket)

Second you will be able to use git to do some version control, if you are writing a piece of code for 3 days for example. You need to add something with no link, and we need to be sure that you don't have any issues, for that you can do a copy of your code and work on the copy.

Github online, git on your machine.

A repository is the place where you're going to push your code on Github.

It's nice to separate things, so one challenge, one repo.

Create a new repository :

- Choose repository name.
- Write a little description (optional)
- choose if it's public or private
- Add a README file, are the files that contains the description of the repo.
- Git ignore, just ignore certain files that you don't want to be on Github. (very important)
- Choose a license (MIT license is good), it's not a mandatory, it's a sentence that will protect your code (law?)

(Github copilot is a help to code when we write, it's a paid service based on IA, that use the code on Github to perfect the IA and predict the code you will write.)

So now we have a repo with a README file, but it's only on github, we want to have it on our computer.

To do that, you need to create a link between your computer and the github repo. For that go on your repository and clone your repository, by HTTPS, SSH or Github CLI way.

CLI is a webservice software that you can install to your terminal to communicate for Github. It's a proprietary version of Git.

Copy the HTTPS link, write `git clone « link from Github »`

Now we have a folder with the README file and in the file the title of the repo.

We can modify the README file, by adding some content. However it's only modify on the computer and not on Github.

We need to keep the modification, so we want to put it on Github.

The folder (copy of your repo) is tracked by Git, actually on the main branch, the only one actually.

If you want to check if you modified the code you can type the command `git status` it gives you all the files modified.

`git add .` the `.` is to say that I want to add everything.

(It put all the content in the bus and we can check if it has been added with `git status`)

`git commit -m` here `-m` is a flag (option to be more precise on the command)

When we do a `git commit -m` we need to put a message with it, we can for example say what we did in the repo or code.

(Now the little cross disappear but the bus is ready to go to Github)

`git push` It's the command to send the work on Github, but we need to tell where it needs to go.

For that we use `git remote -v` to have the address (that we copy earlier from Github) of the repo, but it's also called « origin »

So we can do : `git push origin main` `main` is to say that the content need to go on the main branch.

Let's do a little recap :

- do a clone of the repo. (`git clone`) (only if you have created a new repo)
- add content in the copy, new file, modify files who already exist, etc.
- add all the content to be send. (`git add .`)
- give a message. (`git commit -m`)
- send it to Github. (`git push origin main`)

With `git add`, we can choose the files that we add. (But it not happen in general)

Difference between Master and Main, Master is just an old word for Main.

Initiate a repo with git init :

To initialize a repo with git init, go to the folder in your terminal, write `git init`.

At every time remember that you can use the command `git status` to verify the status of your files.

To track your files, use as usual the command `git add .`, after that you can `git commit -m "your message"`

First time it will ask you to login and run the commands `git config --global user.email "youremail@email.com"` and the command `git config --global user.name "Your Name"`, personally for the name I put my Github UserName

Next you need to change the name of the branch : `git branch -M main` and add the link to the repository : `git remote add origin git@github.com :WillPolet/RepoName.git`

After that you can `git push origin main`, but be aware, if there was a README file into your repo created on github, this will cause a problem, here's how to solve it :

Type these six commands in your terminal to install gh.

```
type -p curl >/dev/null || (sudo apt update && sudo apt install curl -y)
```

```
curl -fsSL https://cli.github.com/packages/githubcli-archive-keyring.gpg | sudo dd of=/usr/share/keyrings/githubcli-archive-keyring.gpg \
```

```
&& sudo chmod go+r /usr/share/keyrings/githubcli-archive-keyring.gpg \
```

```
&& echo "deb \[arch=\$(dpkg --print-architecture) signed-by=/usr/share/keyrings/githubcli-archive-keyring.gpg\] https://cli.github.com/packages stable main" | sudo tee /etc/apt/sources.list.d/github-cli.list > /dev/null \
```

```
&& sudo apt update \
```

```
&& sudo apt install gh -y
```

After you need to authenticate with gh with `gh auth login`, choose to login through HTTPS or SSH (had an issue with HTTPS so I did it with SSH after).

Very important, if you already had a README file, now you need to pull it by using this command : `git pull origin main --allow-unrelated-histories`.

And now you should be able to `git push origin main`

So if there's no troubles, follow the instructions below, if there's a problem search in this document, if the problem persist go on google.

- Create a README.md file in your folder (optionnal)
- `git init`
- `git add .`
- `git commit -m`
- `git branch -M main`
- `git remote add origin git@github.com :WillPolet/RepoName.git`
- `git push -u main origin` -u is optionnal

If you don't have create your repo on Github yet, you can type this command after your commit :

```
gh repo create my-newrepo --public --source=. --remote=upstream --push
```

After that no need to do another command line, your files will be on your fresh new Github repo!

HTML sementic :

Sementic HTML is a way to build an HTML file with html tag that describes what it does.
Exemple of tags that you will use in these kind of document :

- | | | | |
|--------------|----------|-----------|------|
| • h1 | • q | • caption | • td |
| • h2 | • img | • table | • ul |
| • blockquote | • hr | • th | • ol |
| • q | • figure | • tr | • li |

And there's way more!