

LE SHELL

116 101 114 109 105 110
97 108 32 109 97 99 79 83*

*Terminal macOS

Par Necib Mohamed-el-Amine



INTRODUCTION

Qu'est ce que le Shell?

Le shell(ou interface système en français) est un programme qui reçoit des commandes informatiques données par un utilisateur à partir de son clavier pour les envoyer au système d'exploitation qui se chargera de les exécuter.

Au début de l'informatique, le shell était la seule interface utilisateur disponible sur un système de type UNIX tel que Linux. De nos jours et avec l'arrivée de la souris, tous les ordinateurs utilisent des interfaces utilisateur graphiques (GUI, pour "graphical user interface") comme macOS ou Windows 11, en plus des interfaces de ligne de commande (CLI, pour "command line interface"), comme le shell.

Le mot "Shell" est employé ici comme une métaphore. Shell signifie enveloppe ou coque en français: à l'inverse du noyau d'un ordinateur, le shell désigne la couche la plus haute de toutes les interfaces des systèmes Unix (Linux, macOS).

A quoi sert un script Shell?

Sur la plupart des systèmes Linux, un programme appelé bash (qui signifie Bourne Again Shell, version améliorée du programme shell Unix d'origine, sh écrit par Steve Bourne) agit en tant que programme Shell. Sur macOS, le shell est accessible par l'application Terminal. Un terminal est un programme qui permet à l'utilisateur d'interagir avec le shell. Plusieurs émulateurs de terminaux existent sur Linux, comme gnome-terminal, konsole, xterm, rxvt, kvt, nxterm et eterm.

Tous les shell peuvent exécuter des commandes situées dans un fichier. Chaque fichier contenant des commandes destinées au shell se nomme un script (un programme écrit dans le langage propre au shell). Un script shell ressemble à un fichier texte simple exécutable par la machine mais qui doit être interprété par l'émulateur de terminal. Ce dernier a principalement pour objectif de lancer et de coordonner l'exécution de programmes sans passer par l'interface graphique (GUI Script). Un script shell peut également avoir d'autres utilisations : il peut être une méthode d'automatisation avec Python par exemple, ou encore générer des pages dynamiques en PHP ou JSP.



Utilisant un Macbook Air, j'ai effectué les exercices dans un premier sur l'OS Mac. Je me suis donc permis d'adapter les exercices demandés. Vous trouverez donc une référence au fichier zshrc plutôt que bashrc comme il était demandé dans l'intitulé.

JOB-01

Objectifs du Job:

1-Afficher le manuel de la commande ls

2-Afficher les fichiers cachés du home de votre utilisateur

3-Afficher les fichiers cachés plus les informations sur les droits sous forme de listes

Pour afficher le manuel de la commande ls il suffit simplement de taper la commande: man ls

Pour afficher les fichiers cachés du répertoire home il faudra taper la commande: ls -a

Pour afficher les fichiers cachés + les informations sur les droits sous forme de listes: ls -la

```

aminenecib@MacBook-Air-de-Amine:~
> ls -la
total 470224
drwxr-xr-x+ 86 aminenecib staff      2752 22 sep 13:57 .
drwxr-xr-x  5 root      admin       160 11 aoû 08:44 ..
-r-----  1 aminenecib staff        7 29 avr 2021 .CFUserTextEncoding
-rw-r--r--@ 1 aminenecib staff     10244 19 sep 15:38 .DS_Store
drwxr-xr-x+ 28 aminenecib staff      896 22 sep 10:26 .Trash
drwxr-xr-x  4 aminenecib staff     128 15 jul 2021 .android
drwxr-xr-x 14 aminenecib staff      448 15 sep 16:16 .atom
-rw-----  1 aminenecib staff      40  2 jul 2021 .bash_history
-rw-r--r--  1 aminenecib staff     417 21 sep 09:04 .bash_profile
-rw-r--r--  1 aminenecib staff     295 21 sep 09:04 .bashrc
drwxr-xr-x  8 aminenecib staff     256 22 sep 13:57 .cache
drwxr-xr-x  3 aminenecib staff      96 25 jul 2021 .cocoapods
drwx----- 7 aminenecib staff     224 21 sep 09:04 .config
drwxr-xr-x  3 aminenecib staff      96 18 mai 2021 .cups
drwxr-xr-x  7 aminenecib staff     224  9 sep 2021 .expo
drwxr-xr-x 16 aminenecib staff     512 22 sep 10:48 .fig
drwxr-xr-x  3 aminenecib staff      96 21 sep 09:04 .fig_dotfiles.bak
-rw-r--r--  1 aminenecib staff      91 24 jui 2021 .gitconfig
-rw-r--r--@ 1 aminenecib staff    6607 22 sep 13:16 .hyper.js
drwxr-xr-x  6 aminenecib staff     192 21 sep 16:30 .hyper_plugins
-rw-----  1 aminenecib staff      28  2 aoû 2021 .lessht
drwxr-xr-x  4 aminenecib staff     128 21 sep 09:04 .local
-rw-----  1 aminenecib staff     206 11 jui 2021 .netrc
-rw-----  1 aminenecib staff      27 25 mar 11:10 .node_repl_history
drwxr-xr-x  9 aminenecib staff     288 22 sep 11:46 .npm
drwxr-xr-x 22 aminenecib staff     704 20 sep 08:56 .oh-my-zsh

```

Il est possible de rajouter des options à une commande en ajoutant un tiret (-) (flag) plus l'option représentée par une lettre (ici le 'la').

Les deux syntaxes principales d'écritures des options pouvant être utilisées sont:

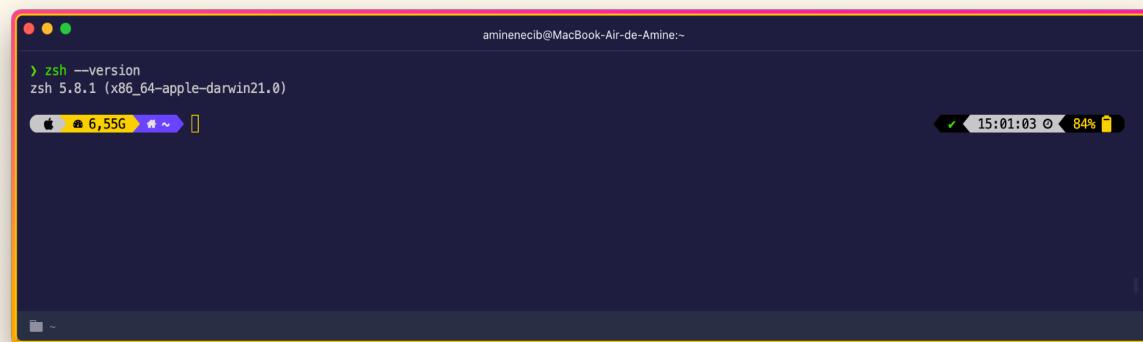
-Pour une commande avec une option contenant un seul caractère on ajoute un flag, exemple:
ls -a (1)

-Pour une commande avec une option contenant plus d'un caractère on ajoute alors deux flags, exemple: zsh --version (2)



```
aminenecib@MacBook-Air-de-Amine:~
> ls -a
.
..
.CFUserTextEncoding
.DS_Store
.Trash
.android
.atom
.bash_history
.bash_profile
.bashrc
.cache
.cocoapods
.yarnrc
.zcompdump-MacBook Air de Amine-5.8
.zcompdump-MacBook Air de Amine-5.8.1
.zcompdump-MacBook Air de Amine-5.8.1.zwc
.zprofile
.zprofile.swp
.zsh
.zsh_history
.zsh_sessions
.zshrc
Applications
```

(1)



```
aminenecib@MacBook-Air-de-Amine:~
> zsh --version
zsh 5.8.1 (x86_64-apple-darwin21.0)
Apple ~ 6,55G 15:01:03 84%
```

(2)

JOB-02

Objectifs du Job:

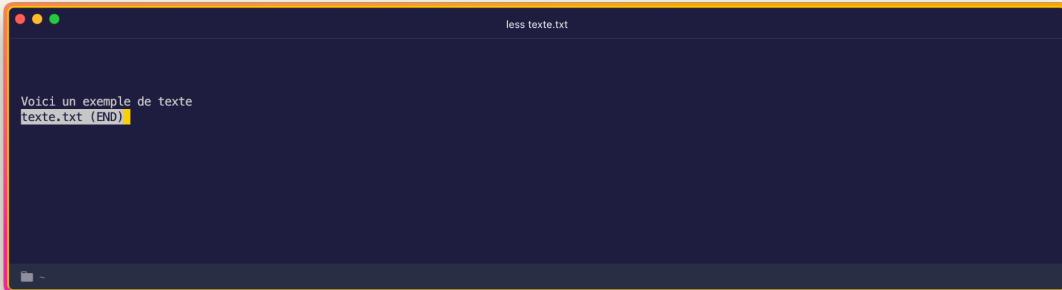
- Lire un fichier en utilisant une commande qui permet seulement de lire
- afficher les 10 premières lignes du fichier “.zshrc”
- afficher les 10 dernières lignes du fichier “.zshrc”
- afficher les 20 premières lignes du fichier “.zshrc”
- afficher les 20 dernières lignes du fichier “.zshrc”

Afin de lire un fichier on utilise la commande less + nom_du_fichier, on peut également utiliser cat.

(1) Less permet d' ouvrir un fichier contenant énormément de contenu et de le consulter en partant du début, page par page en défilant..

(2) Cat affichera tout le fichier ,malgré son volume, en nous propulsant à la fin de celui-ci.

(1)



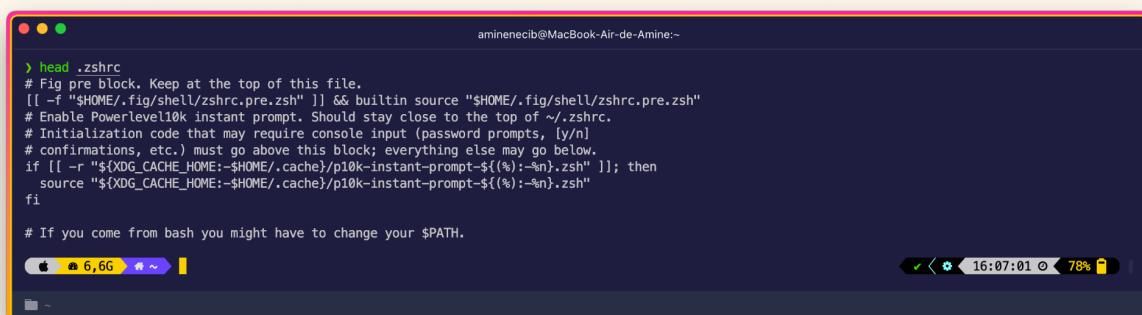
```
less texte.txt
Voici un exemple de texte
[END]
```



```
aminenecib@MacBook-Air-de-Amine:~
> cat texte.txt
Voici un exemple de texte
6,86G ~ 15:07:23 83%
```

(2)

(1) Pour afficher les dix premières lignes d'un fichier (dans le cas présent j'ai choisis le fichier zshrc car il contient énormément d'informations), il nous faut entrer la commande: head .zshrc



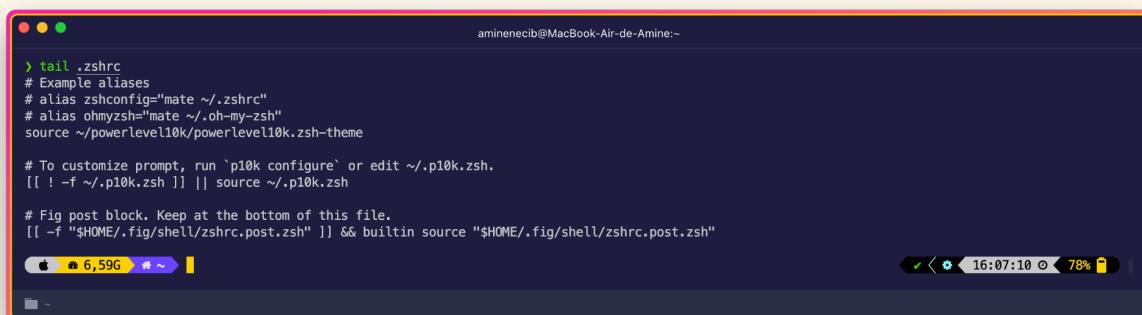
```
aminenecib@MacBook-Air-de-Amine:~
```

```
> head .zshrc
# Fig pre block. Keep at the top of this file.
[[ -f "$HOME/.fig/shell/zshrc.pre.zsh" ]] && builtin source "$HOME/.fig/shell/zshrc.pre.zsh"
# Enable Powerlevel10k instant prompt. Should stay close to the top of ~/.zshrc.
# Initialization code that may require console input (password prompts, [y/n]
# confirmations, etc.) must go above this block; everything else may go below.
if [[ -r "${XDG_CACHE_HOME:-$HOME/.cache}/p10k-instant-prompt-${(%):-%n}.zsh" ]]; then
    source "${XDG_CACHE_HOME:-$HOME/.cache}/p10k-instant-prompt-${(%):-%n}.zsh"
fi

# If you come from bash you might have to change your $PATH.


```

(2) Pour afficher les dix dernières lignes d'un fichier il nous faut entrer la commande: tail .zshrc



```
aminenecib@MacBook-Air-de-Amine:~
```

```
> tail .zshrc
# Example aliases
# alias zshconfig="mate ~/.zshrc"
# alias ohmyzsh="mate ~/.oh-my-zsh"
source ~/powerlevel10k/powerlevel10k.zsh-theme

# To customize prompt, run `p10k configure` or edit ~/.p10k.zsh.
[[ ! -f ~/.p10k.zsh ]] || source ~/.p10k.zsh

# Fig post block. Keep at the bottom of this file.
[[ -f "$HOME/.fig/shell/zshrc.post.zsh" ]] && builtin source "$HOME/.fig/shell/zshrc.post.zsh"
```

Les commandes head et tail affichent par défaut respectivement les 10 premières et 10 dernières lignes d'un fichier.

(3) Pour afficher un nombre donné de lignes d'un fichier il faut utiliser l'option -n suivi du nombre de lignes que l'on veut faire apparaître. Dans le cas présent on souhaite afficher les 20 premières lignes du fichier zshrc, on utilisera donc: head -n 20 .zshrc

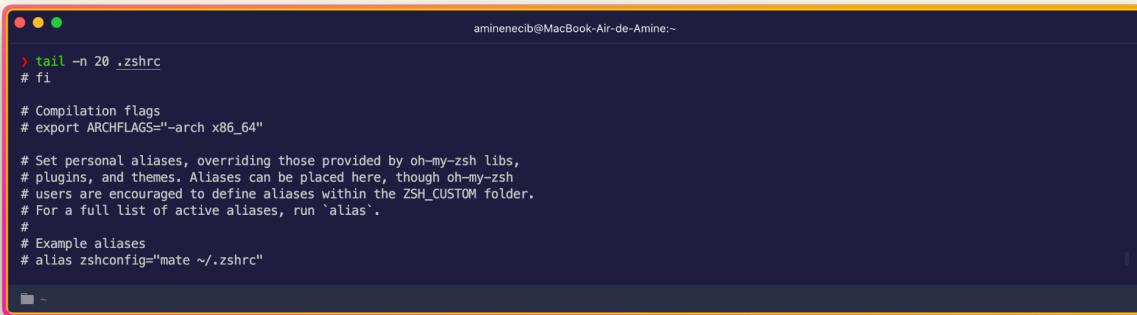


```
aminenecib@MacBook-Air-de-Amine:~
```

```
> head -n 20 .zshrc
# Fig pre block. Keep at the top of this file.
[[ -f "$HOME/.fig/shell/zshrc.pre.zsh" ]] && builtin source "$HOME/.fig/shell/zshrc.pre.zsh"
# Enable Powerlevel10k instant prompt. Should stay close to the top of ~/.zshrc.
# Initialization code that may require console input (password prompts, [y/n]
# confirmations, etc.) must go above this block; everything else may go below.
if [[ -r "${XDG_CACHE_HOME:-$HOME/.cache}/p10k-instant-prompt-${(%):-%n}.zsh" ]]; then
    source "${XDG_CACHE_HOME:-$HOME/.cache}/p10k-instant-prompt-${(%):-%n}.zsh"
fi

# If you come from bash you might have to change your $PATH.
# export PATH=$HOME/bin:/usr/local/bin:$PATH
```

(4) Ainsi pour afficher les 20 dernières lignes du fichier on utilisera: tail -n 20 .zshrc



```
aminenecib@MacBook-Air-de-Amine:~
```

```
> tail -n 20 .zshrc
# fi

# Compilation flags
# export ARCHFLAGS="-arch x86_64"

# Set personal aliases, overriding those provided by oh-my-zsh libs,
# plugins, and themes. Aliases can be placed here, though oh-my-zsh
# users are encouraged to define aliases within the ZSH_CUSTOM folder.
# For a full list of active aliases, run `alias`.
#
# Example aliases
# alias zshconfig="mate ~/.zshrc"
```

JOB-03

Objectifs du Job:

- Installer le paquet “cmatrix”
- lancer le paquet que vous venez d’installer
- Mettre à jour son gestionnaire de paquets
- Mettre à jour ses différents logiciels
- Télécharger les internets : Google
- Redémarrer votre machine
- éteindre votre machine

Pour installer le paquet “cmatrix” il m’aura fallu d’abord installer homebrew.

Homebrew est un gestionnaire de paquets pour mac OS, gratuit et open-source écrit en Ruby. Son but est de simplifier l’installation de programmes.



Pour installer Homebrew c’est très simple, on se réfère à la documentation du créateur et l’on suit les étapes. Dans un premier temps on utilise dans notre terminal la ligne de commande:

Installer Homebrew

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Le gestionnaire de paquets étant maintenant installé, nous devons installer le programme cmatrix. Pour ce faire on utilisera l'invite de commande \$ brew install cmatrix.



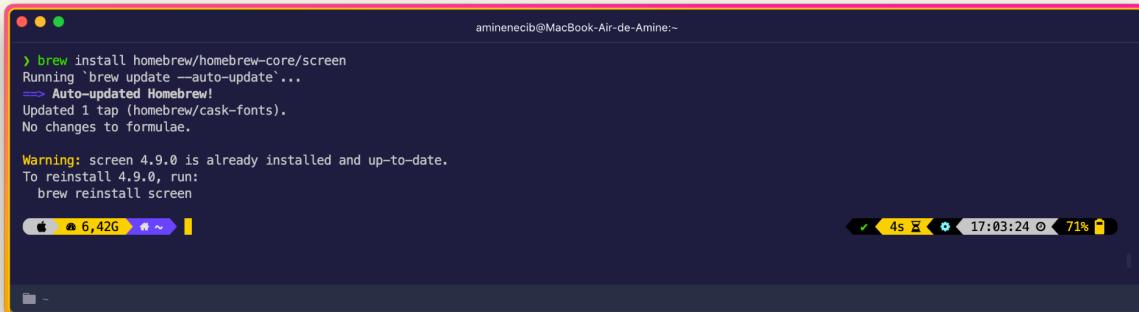
```
aminenecib@MacBook-Air-de-Amine:~
> brew install cmatrix
Running `brew update --auto-update`...
=> Auto-updated Homebrew!
Updated 2 taps (homebrew/core and homebrew/cask).
=> New Casks
lunacy

Warning: cmatrix 2.0 is already installed and up-to-date.
To reinstall 2.0, run:
  brew reinstall cmatrix

  6,27G 5s 16:42:03 73%
```

Pour faire fonctionner cmatrix qui est distribué sous licence GNU nous allons avoir besoin de GNU SCREEN pour autoriser notre terminal. En effet Screen est une console virtuelle qui permet de partager un terminal en plusieurs processus. C'est lui aussi un logiciel libre distribué par le projet GNU selon les termes de la licence GPL.

On utilise donc l'invite de commande suivante pour l'installer :

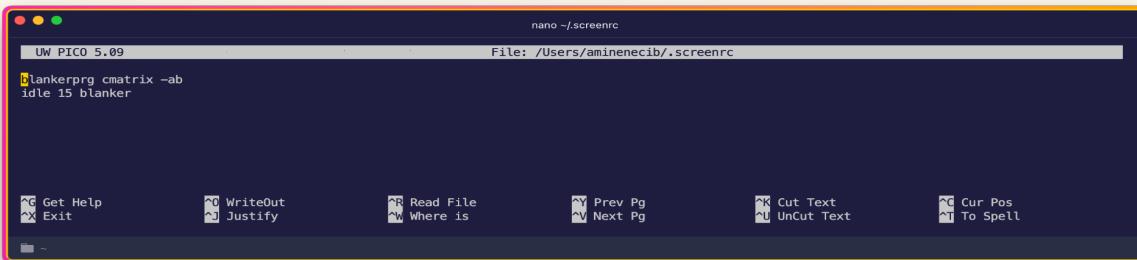
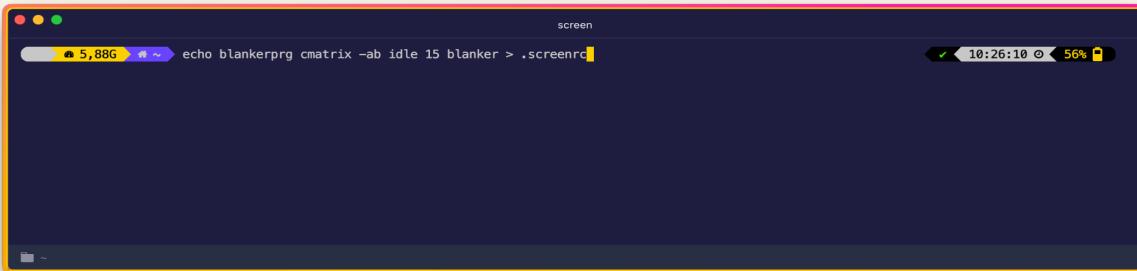


```
aminenecib@MacBook-Air-de-Amine:~
> brew install homebrew/homebrew-core/screen
Running `brew update --auto-update`...
=> Auto-updated Homebrew!
Updated 1 tap (homebrew/cask-fonts).
No changes to formulae.

Warning: screen 4.9.0 is already installed and up-to-date.
To reinstall 4.9.0, run:
  brew reinstall screen

  6,42G 4s 17:03:24 71%
```

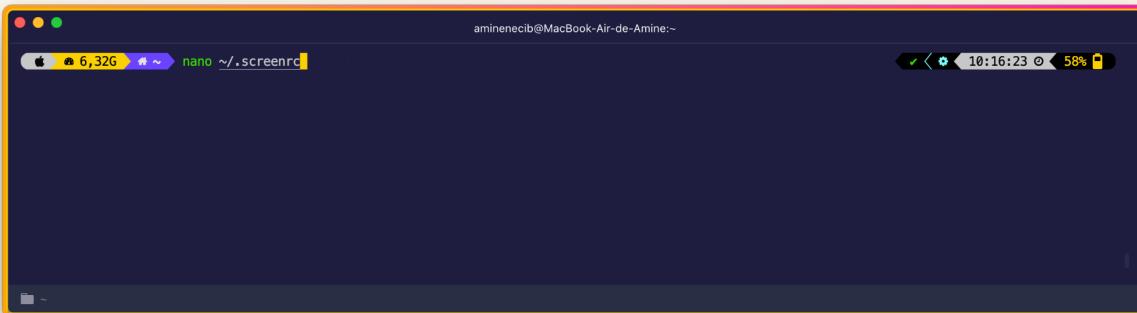
On crée ensuite un fichier .screenrc. Celui-ci sert à configurer l'initialisation du programme screen pour la suite. On peut créer ce fichier en y intégrant directement les paramètres que l'on veut lui attribuer en une seule ligne de commande. Pour ce faire on execute la commande suivante:



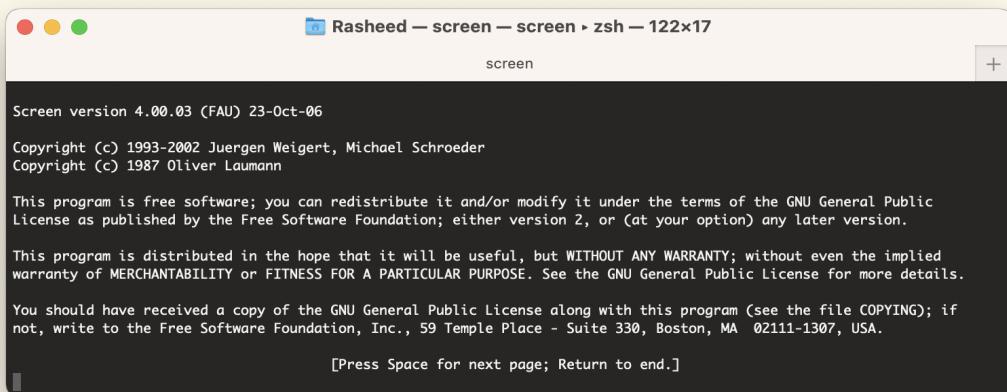
blankerprg cmatrix -ab => Demande au programme screen d'utiliser cmatrix comme screenblanker. Il est dans un premier temps sollicité pour occulter l'écran puis Cmatrix prend le relais pour y afficher ses outputs qui apparaîtront sur l'écran.

idle 15 blanker => Active l'occultant d'écran après un délai de 15 secondes d'inactivités. Vous pouvez quitter cet écran en pressant n'importe quelle touche par la suite. Vous pouvez paramétriser le délai de lancement du processus en modifiant le temps à votre guise.

Pour entrer dans le fichier et y modifier les paramètres il vous suffit de taper la commande suivante:



Enfin pour initier notre écran de veille, il suffira de lancer une fois le programme screen en tapant simplement screen dans notre terminal. Il affichera ensuite cette page:



```

Screen version 4.00.03 (FAU) 23-Oct-06
Copyright (c) 1993-2002 Juergen Weigert, Michael Schroeder
Copyright (c) 1987 Oliver Laumann

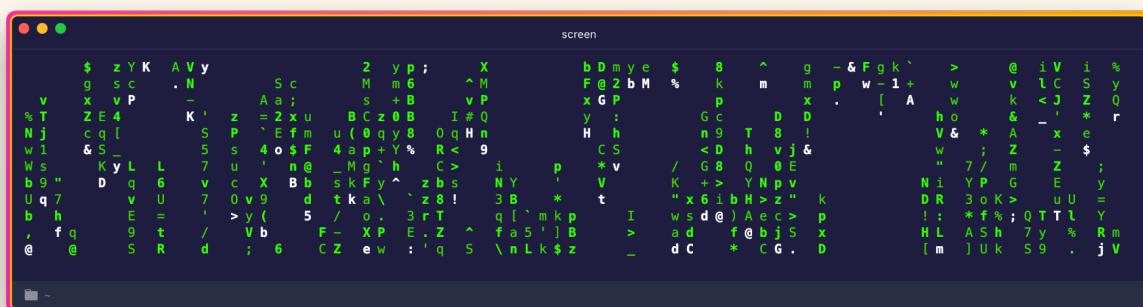
This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public
License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program (see the file COPYING); if
not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

[Press Space for next page; Return to end.]
```

Au bout de 15 secondes comme paramétré vous pourrez constater un sublime affiche sur le thème du célèbre film Matrix

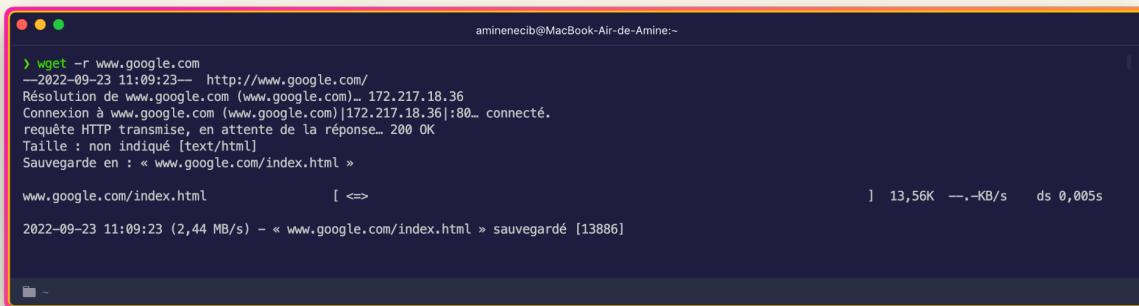


Pour mettre à jour mon gestionnaire de paquets sur macOS on tapera la commande => brew update.

Pour mettre à jour les différents logiciels on utilisera ,comme précédemment, la commande brew suivi d'upgrade cette fois. => brew upgrade

Utilisant un mac, la commande wget n'est pas présente par défaut sur mon terminal, il m'a donc fallu l'installer pour pouvoir l'utiliser. Pour ce faire, rien de plus simple, j'ai utilisé la commande => brew install wget

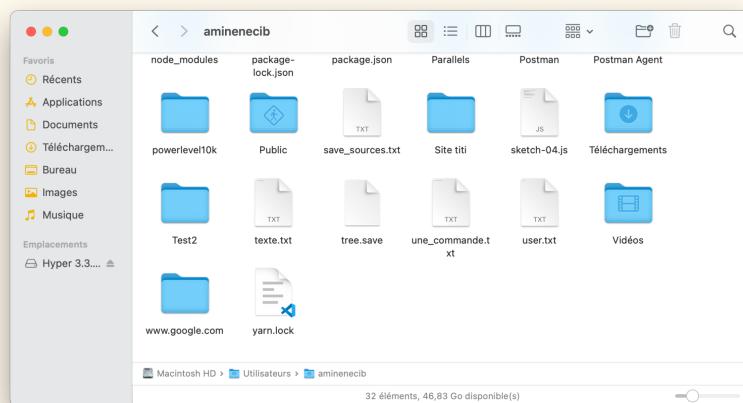
Une fois installée j'ai pu télécharger les internets grâce à la commande => wget -r www.google.com



```
aminenecib@MacBook-Air-de-Amine:~$ > wget -r www.google.com
--2022-09-23 11:09:23--  http://www.google.com/
Résolution de www.google.com (www.google.com)... 172.217.18.36
Connexion à www.google.com (www.google.com)|172.217.18.36|:80... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : non indiqué [text/html]
Sauvegarde en : < www.google.com/index.html >
www.google.com/index.html [ <> ] 13,56K --.-KB/s  ds 0,005s
2022-09-23 11:09:23 (2,44 MB/s) - < www.google.com/index.html > sauvegardé [13886]
```

-r étant l'option qui renvoi à la récursivité, en d'autres termes c'est une démarche dont la description mène à la répétition d'une même règles.

Un fichier www.google.com se crée ensuite dans le dossier Home de l'utilisateur. Ce dossier contient toutes les informations présentes sur la page internet.



Pour redémarrer notre ordinateur via le terminal il nous suffit de taper la commande => sudo shutdown -r now

Pour éteindre notre ordinateur via le terminal il faut cette fois taper la commande => sudo shutdown -h now

JOB-04

Objectifs du Job:

- Créer un groupe appelé “Plateformeurs”
- Créer un utilisateur appelé “User1”
- Créer un utilisateur appelé “User2”
- Ajouter “User2” au groupe "Plateformeurs"
- Copier votre “users.txt” dans un fichier “droits.txt”
- Copier votre “users.txt” dans un fichier “groupes.txt”
- Changer le propriétaire du fichier “droits.txt” pour mettre “User1”
- Changer les droits du fichier “droits.txt” pour que “User2” ai accès seulement en lecture
- Changer les droits du fichier “groupes.txt” pour que les utilisateurs puissent accéder au fichier en lecture uniquement
- Changer les droits du fichier pour que le groupe “Plateformeurs” puissent y accéder en lecture/écriture.

Pour créer un groupe nommé “Plateformeurs” nous allons devoir taper la commande => *dscl . -create /Groups/nom_du_groupe*

Pour créer un utilisateur appelé “User1”, la commande est similaire mais l'on remplace Groups par Users comme ceci => *dscl . -create /Users/nom_du_user*

Pour pouvoir ajouter un utilisateur à mon groupe “Plateformeurs”, il faut au préalable que je crée un ID à ce même utilisateur. L' ID est choisi par le créateur mais il doit être unique. Le

terminal sur Mac ne permettant pas de déplacer des Utilisateurs par leurs noms.

Pour ce faire on utilise la commande => `sudo dscl . -create /Users/nom_du_user UniqueID "ID"`

Attention: ne pas oublier le sudo, car nous devons passer en mode superutilisateurs pour bénéficier des droits admin.

L'ID de l'utilisateur étant créé, on peut maintenant déplacer l'utilisateur vers le groupe que l'on souhaite (ici "Plateformeurs"). Pour cela on utilise la commande => `sudo dseditgroup -o edit -a user3 -t "1804" Plateformeurs`

Pour copier le fichiers users.txt dans le fichier droits.txt il nous suffit de taper la ligne de commande => `cp users.txt droits.txt`

Cette commande copiera directement le contenu du fichier users.txt et le collera dans le fichier droits.txt

De la même manière on copiera donc le fichier users.txt dans le fichier groupes.txt à l'aide de la commande => `cp users.txt groupes.txt`

Pour changer le propriétaire du fichier droits.txt il faudra utiliser la commande => `sudo chown id_du_user droits.txt`

Pour changer les droits du fichier droits.txt et les attribués seulement à la lecture pour le user2 on utilisera la commande => `sudo chmod 744 droits.txt`

chmod signifie *change file modes or access control lists*. Dans le cas présent on attribuera tous les droits à l'administrateur mais que les membres d'un groupe ainsi que tous les autres utilisateurs n'ai que le droit de lecture on utilisera 744. Chaque chiffre représentent un droit, le total de tous ces droits fera un total de 7. Cette valeur s'appelle la valeur Octale, elle peut être représentée par des lettres : rwx ou - si nul.

Octal	Decimal	Permission	Representation
000	0 (0+0+0)	No Permission	---
001	1 (0+0+1)	Execute	--x
010	2 (0+2+0)	Write	-w-
011	3 (0+2+1)	Write + Execute	-wx
100	4 (4+0+0)	Read	r--
101	5 (4+0+1)	Read + Execute	r-x
110	6 (4+2+0)	Read + Write	r-w-
111	7 (4+2+1)	Read + Write + Execute	rwx

Pour changer les droits du fichier groupes.txt pour que les utilisateurs puissent accéder au fichier en lecture uniquement il faut utiliser la commande suivante => *sudo chmod 665 groupes.txt*

On peut contrôler ensuite que le groupe Plateformeurs a les droits de lecture et d'écriture sur le fichier en utilisant la commande => *ls -l*

JOB-05

Objectifs du job:

- Ajouter un alias qui permettra de lancer la commande “ls -la” en tapant “la”
- Ajouter un alias qui permettra de lancer la commande “apt-get update” en tapant “update”
- Ajouter un alias qui permettra de lancer la commande “apt-get upgrade” en tapant “upgrade”
- Ajouter une variable d'environnement qui se nommera “USER” et qui sera égale à votre nom d'utilisateur
- Mettre à jour les modifications de votre bashrc dans votre shell actuel
- Afficher les variables d'environnement
- Ajouter à votre Path le chemin ”/home/'votre utilisateur'/Bureau”

Pour ajouter un alias qui permet de lancer la commande *ls -la* en tapant seulement *la*, il suffit d'utiliser la commande suivante => *alias la="ls -la"*

Pour ajouter un alias qui permet de lancer la commande *brew update* en tapant seulement *update* on aura juste à créer comme précédemment un alias comme ceci => *alias update="brew update"*

Pour ajouter un alias qui permet de lancer la commande *brew upgrade* en tapant seulement *upgrade* on aura juste à créer comme précédemment un alias comme celui => *alias upgrade="brew upgrade"*

Pour créer une variable d'environnement sur macOS il suffit de taper => *export USER=user1*

Pour ajouter un chemin dans une variable d'environnement qui n'est pas temporaire il faut aller dans le fichier *.zshrc* et rajouter le chemin dans la variable souhaitée.

Pour pouvoir voir toutes les variables d'environnement on tapera => *env*

Pour mettre à jour les modifications de notre fichier *zshrc* dans notre shell actuel il suffit d'utiliser la commande => *source ~/.zshrc*

JOB-06

Objectifs du job:

Vous devez télécharger l'archive suivante et la désarchiver seulement avec le terminal.

Cette manipulation vous permettra d'accéder à la suite du sujet.

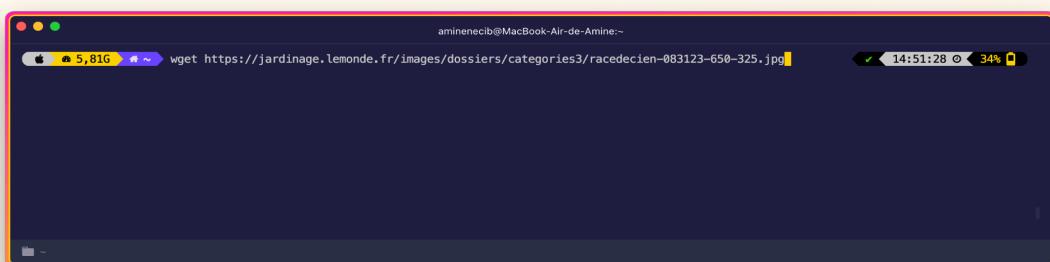
Etant donné que le lien semblé ne pas fonctionner nous n'avons pas pu faire cette exercice mais nous avons quand même essayé de le faire à partir d'un fichier quelconque.

Pour exemple nous avons pris une image d'un chien trouvé sur le site du monde (lien: <https://jardinage.lemonde.fr/images/dossiers/categories3/racedecien-083123-650-325.jpg>) et nous avons essayé de la télécharger via le terminal.

Pour ce faire nous avons utilisé la commande => *wget*

<https://jardinage.lemonde.fr/images/dossiers/categories3/racedecien-083123-650-325.jpg>

Ainsi nous avons récupéré l'image de ce magnifique labrador :D



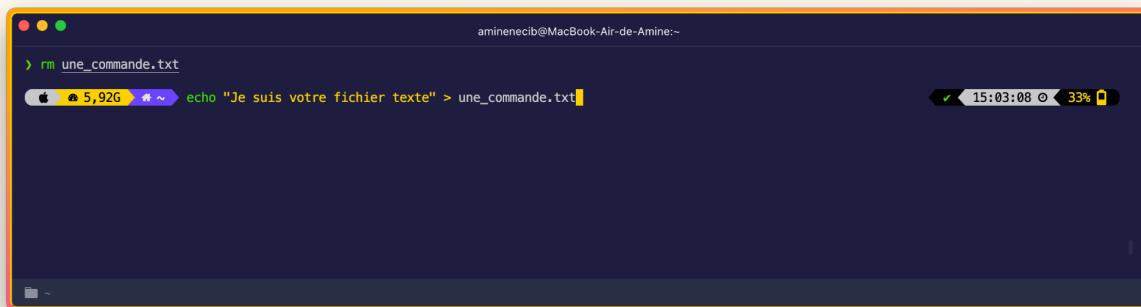
Si le fichier est compressé on ajoutera la commande tar -xvf nom_du_fichier.tgz. Le paramètre x permet d'extraire l'archive, le z de la décompresser et le f indique la donnée à décompresser.

JOB-07

Objectifs du Job:

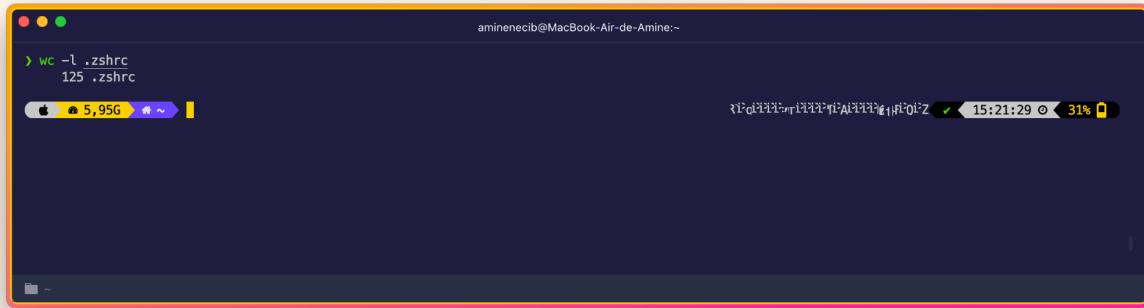
- Créer un fichier “une_commande.txt” avec le texte suivant “Je suis votre fichier texte”
- Compter le nombre de lignes présentes dans votre fichier de source apt et les enregistrer dans un fichier nommé “nb_lignes.txt”
- Afficher le contenu du fichier source apt et l'enregistrer dans un autre fichier appelé “save_sources”
- Faites une recherche des fichiers commençant par “.” tout en cherchant le mot alias qui sera utilisé depuis un fichier

Pour créer un fichier avec le texte suivant “Je suis votre fichier texte”, il nous suffit de taper la commande suivante => echo “je suis votre fichier texte” > une_commande.txt



The screenshot shows a terminal window on a Mac OS X desktop. The window title is 'aminenecib@MacBook-Air-de-Amine:~'. The command entered is 'echo "Je suis votre fichier texte" > une_commande.txt'. The terminal interface includes a status bar at the bottom showing the date and battery level.

Afin de compter le nombre de lignes j'ai choisis de tester la manipulation sur le fichier *zshrc*, et de l'enregistrer dans un fichier nommé “nb_lignes.txt”. Pour ce faire j'ai utilisé la commande suivante => *wc -l .zshrc*



The screenshot shows a terminal window on a Mac OS X desktop. The window title is 'aminenecib@MacBook-Air-de-Amine:~'. The command entered is 'wc -l .zshrc', which outputs '125 .zshrc'. The terminal has a dark blue background. At the bottom, there's a status bar with various icons and text, including '5,95G' (disk usage), '15:21:29' (time), and '31%' (battery level).

On peut voir ici que j'ai 125 lignes dans mon fichier *zshrc*

Pour afficher le contenu du fichier *zshrc* et l'enregistrer dans un fichier *save_sources* j'ai pu utiliser la commande => cat .zshrc | tee save_sources

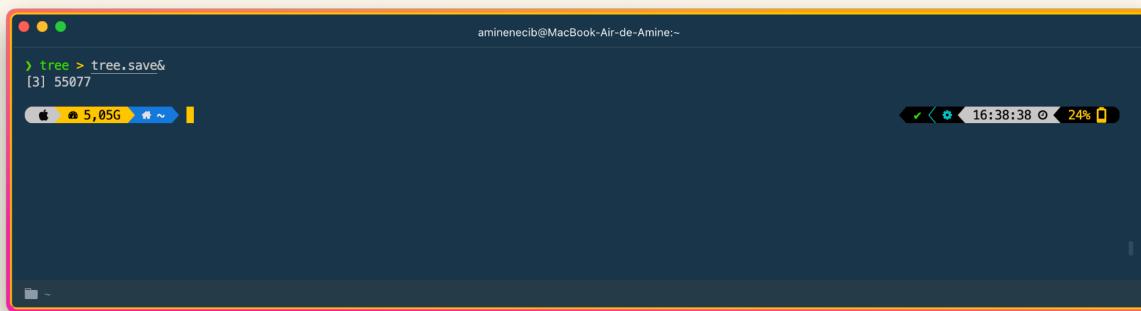
Pour faire une recherche des fichiers commençant par “.” tout en cherchant le mot “alias” qui sera utilisé depuis un fichier on utilise la commande => ls -a .* | grep -r alias

Pour aller plus loin

- Installer la commande tree
- Lancer la commande tree en arrière-plan qui aura pour but d'afficher toute l'arborescence en de votre / en enregistrant le résultat dans un fichier "tree.save"
- lister les éléments présents dans le dossier courant est utilisé directement le résultat de votre première commande pour compter le nombre d'éléments trouvés
- Lancer une commande pour update vos paquets, si l'update réussi alors, vous devrez lancer un upgrade de vos paquets. Si l'update échoue, votre upgrade ne se lancera pas

Pour installer la commande tree, rien de plus simple => *brew install tree*

Pour lancer la commande tree en arrière-plan et enregistrer son résultat dans un fichier *tree.save*, il suffit de taper la commande suivante => *tree >tree.save &*



Cette commande permet aussi d'éviter d'afficher un grand nombre d'informations sur notre terminal et de devoir scroller indéfiniment.

On peut vérifier le fichier en tapant `cat tree.save`.

```
aminenecib@MacBook-Air-de-Amine:~
```

```

|-- RpcProvider.d.ts
|-- RpcProvider.js
|-- RpcProviderInterface.d.ts
|-- RpcProviderInterface.js
|-- index.d.ts
|-- index.js
-- package.json
src
|-- RpcProvider.ts
|-- RpcProviderInterface.ts
-- index.ts
test

```

Pour lister le nombre d'éléments du dossier courant on utilise la commande `ls | wc -l`

```
aminenecib@MacBook-Air-de-Amine:~
```

```

|__ url?q=https://support.google.com/websearch?p=fr_consumer_info&hl=fr&fg=1&sa=U&ved=0ahUKEw19y5rgyqr6AhXSRzABHfPUBPEQ9NMDCAU&usg=A0Vvaw39
bRflB8247D3hm1AstW2
|__ url?q=https://support.google.com/websearch?p=fr_consumer_info&hl=fr&fg=1&sa=U&ved=0ahUKEw1V3PPbyqr6AhXdk4kEHMaAfQ9NMDCAU&usg=A0Vvaw0v8
FatkG52eLj-1v-69mNz
└── webhip?tab=ww
yarn.lock

```

```
344845 directories, 2049659 files
> ls | wc -l
33
```

Pour lancer une update et si celle-ci est valide upgrade : `update && upgrade`.

77 101 114 99 105 32 100
 39 97 118 111 105 114 32
 108 117 32 99 101 32 100

111 115 115 105 101 114
32 101 116 32 100 39 121
32 97 118 111 105 114 32
112 111 114 116 101 114
32 117 110 32 105 110 116
101 114 101 116 32