

# Compte rendu OUTILS LIBRES

REVEL Rémi

6 février 2022

## 1 efficacité de l'environnement de travail

### 1.1 Desactivation de la souris

voici les commande pour desactiver la souris :

1. xinput set-prop 4 "Device Enabled" 0
2. xinput set-prop 6 "Device Enabled" 0
3. xinput set-prop 7 "Device Enabled" 0

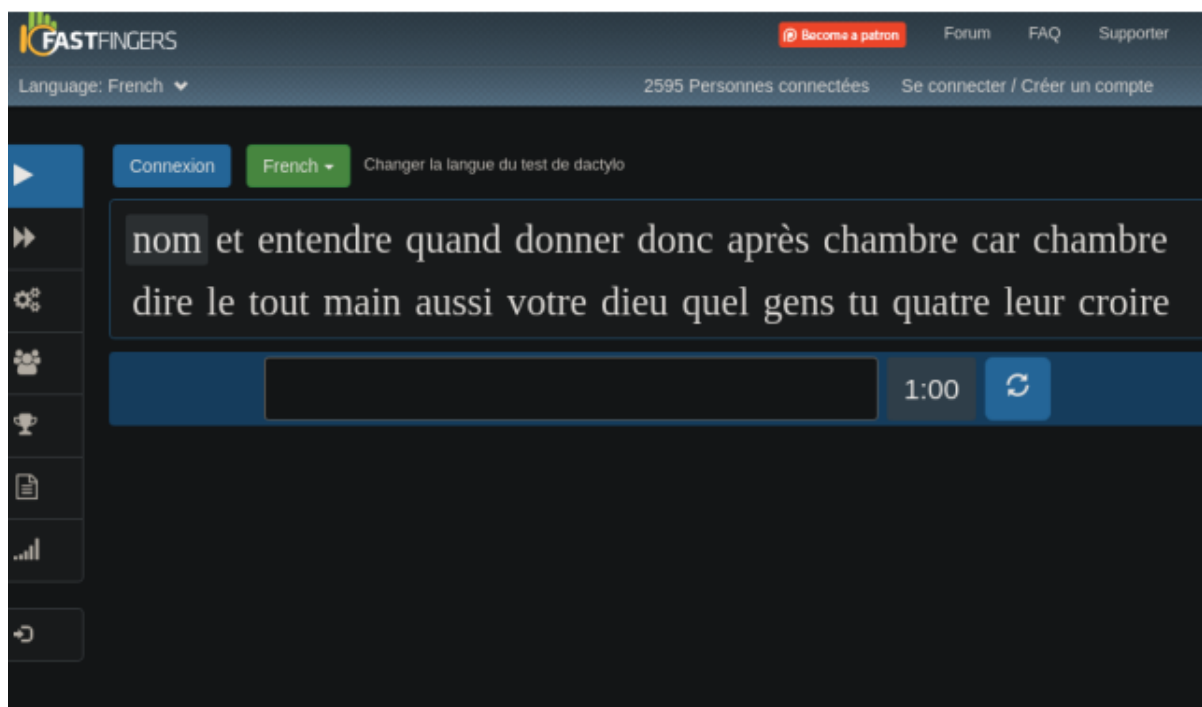
tableau de raccourcis clavier utile :

changer d'application	alt+tab ou windows+tab
gestionnaire d'application	Touche windows
Naviguer sur les elements cliquable d'une page web	tab
Fermer le navigateur	CTRL+W
Fermer une appliacation	ALT+F4
Changer d'onglet sous Brave	CTRL+1,2,3,...
Ouvrir un nouvelle onglet	CMD+T
faire une recherche	F6

### 1.2 S'ameliorer a la dactylographie

le site que j'ai retenu pour s'ameliorer en dactylographie est 10fastfingers.

On peut s'entrainer sur des mots aleatoire, ou sur nos propres texte, le site est disponible dans plusieurs langue.



## 1.3 Tutoriel pour VIM

insertion	i
enregister	:w
quitter	:q
aller au debut du fichier	:1
aller a la fin du fichier	:\$
annuler une action	u
recherche d'une occurrence	/occurrence
activer coloration syntaxique	:syntax on
templacer du texte	:s/origin/replacement/g

pour definir VIM comme editeur par default on a juste a rentrer cette commande :  
update-alternatives --set editor /bin/vim

## 1.4 Bash history

Mon mot de passe n'apparaît pas dans le bash history, donc il n'y a pas d'informations sensible.  
Les historiques sont propres a chaque shell utilisé.

Pour eviter de poluer notre historique avec des commande basique on peut les exclures avec cette commande :  
export HISTIGNORE="ls : cd : pwd"

## 1.5 Alias de fonction

Les commandes presentes ci dessous sont a placer dans le .bashrc

```
function mkcd {  
    mkdir -p $1 && cd $1  
}  
  
function gitemergency {  
    git add . && git commit -m "test" && git push  
}
```

## 1.6 Script

Ce script est fait pour la sauvegarde des données des utilisateurs de la machine.

```
#!/usr/bin/env bash  
  
BACKUP_DIR=/tmp/$(hostname)  
  
getent passwd | egrep ':[0-9]{4}:' | while IFS=: read -r name password uid gid gecos home shell; do  
    USER_BACKUP_DIR=$BACKUP_DIR/$name  
    mkdir -p $USER_BACKUP_DIR  
    sudo cp -R $home/. $USER_BACKUP_DIR/home/  
done  
  
sudo dpkg --get-selections > $BACKUP_DIR/package.list  
sudo cp /etc/{passwd,group,shadow} $BACKUP_DIR  
sudo cp -R /etc/apt/sources.list* $BACKUP_DIR  
sudo apt-key exportall > $BACKUP_DIR/repo.keys  
BACKUP_ARCHIVE_FILE=/tmp/backup_$(hostname)_$(date --iso-8601=seconds).tar.gz  
sudo tar -czvf $BACKUP_ARCHIVE_FILE $BACKUP_DIR  
DISTANT_BACKUP_DIR=/backup/data  
mkdir -p $DISTANT_BACKUP_DIR  
cp $BACKUP_ARCHIVE_FILE $DISTANT_BACKUP_DIR  
cd $DISTANT_B
```

## 1.7 customisation avec OH MY ZSH

Dans le fichier du theme oh y zsh que l'on a selectioné on y rajoute ceci pour pouvoir voir le status de nos Vagrant et notre statut git.

```
# RVM settings
if [[ -s ~/.rvm/scripts/rvm ]] ; then
  RPS1="%{$fg[yellow]}rvm:%{$reset_color}%{$fg[red]}\\$(~/.rvm/bin/rvm-prompt)%{$reset_color} $EPS1"
else
  if which rbenv &> /dev/null; then
    RPS1="%{$fg[yellow]}rbenv:%{$reset_color}%{$fg[red]}\\$(rbenv version | sed -e 's/ (set.*$//')'%{$reset_color} $EPS1"
  fi
fi

ZSH_THEME_GIT_PROMPT_PREFIX="%{$reset_color}%{$fg[green]}["
ZSH_THEME_GIT_PROMPT_SUFFIX="%{$reset_color%}"
ZSH_THEME_GIT_PROMPT_DIRTY="%{$fg[red]}*%{$reset_color%}"
ZSH_THEME_GIT_PROMPT_CLEAN=""
ZSH_THEME_VAGRANT_PROMPT_PREFIX="%{$fg_bold[blue]}["
ZSH_THEME_VAGRANT_PROMPT_SUFFIX="%{$fg_bold[blue]}]%{$reset_color%} "
ZSH_THEME_VAGRANT_PROMPT_RUNNING="%{$fg_no_bold[green]}●"
ZSH_THEME_VAGRANT_PROMPT_POWEROFF="%{$fg_no_bold[red]}●"
ZSH_THEME_VAGRANT_PROMPT_SUSPENDED="%{$fg_no_bold[yellow]}●"
ZSH_THEME_VAGRANT_PROMPT_NOT_CREATED="%{$fg_no_bold[white]}○"

# Customized git status, oh-my-zsh currently does not allow render dirty status before branch
git_custom_status() {
  local cb=$(git_current_branch)
  if [ -n "$cb" ]; then
    echo "$(parse_git_dirty)$ZSH_THEME_GIT_PROMPT_PREFIX$(git_current_branch)$ZSH_THEME_GIT_PROMPT_SUFFIX"
  fi
}

PROMPT='$(git_custom_status)%{$fg[cyan]}[%~% ]%{$reset_color}%B$b $(vagrant_prompt_info)$ (svn_prompt_info)$ (git_prompt_info)%(!.#.$) '
```

## 1.8 Raccourci clavier

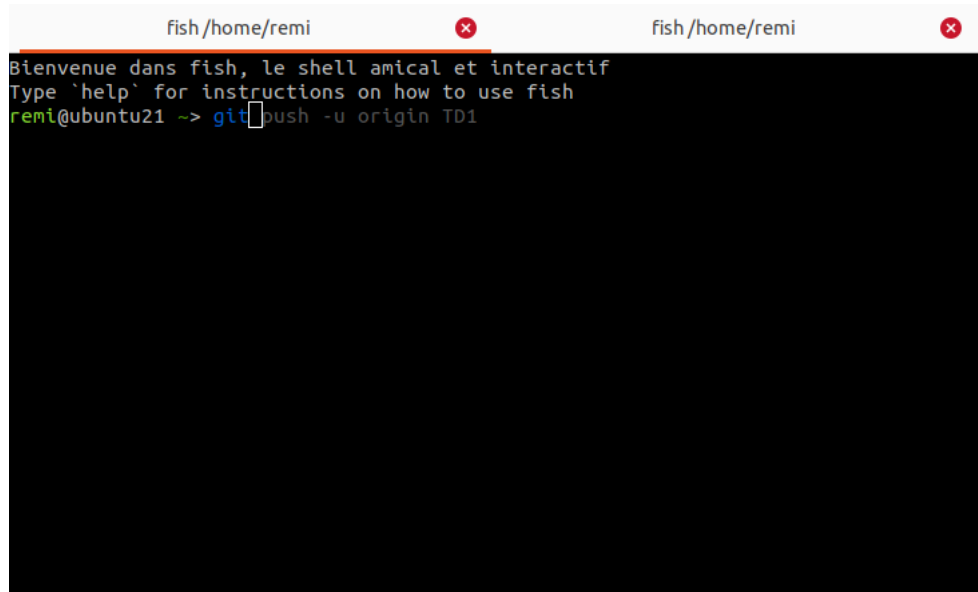
Pour faire un raccourcis clavier qui start ou stop apache en faisant CTRL+A, on place ce petit bout de code dans notre .bashrc

```
function ctrlSA()
{
  stat=`sudo service --status-all | grep apache2 | cut -d" " -f3`
  if [[ $stat == '+' ]]
  then
    systemctl stop apache2
    systemctl status apache2
  else
    systemctl start apache2
    systemctl status apache2
  fi
}
```

## 1.9 Emulateur de terminaux

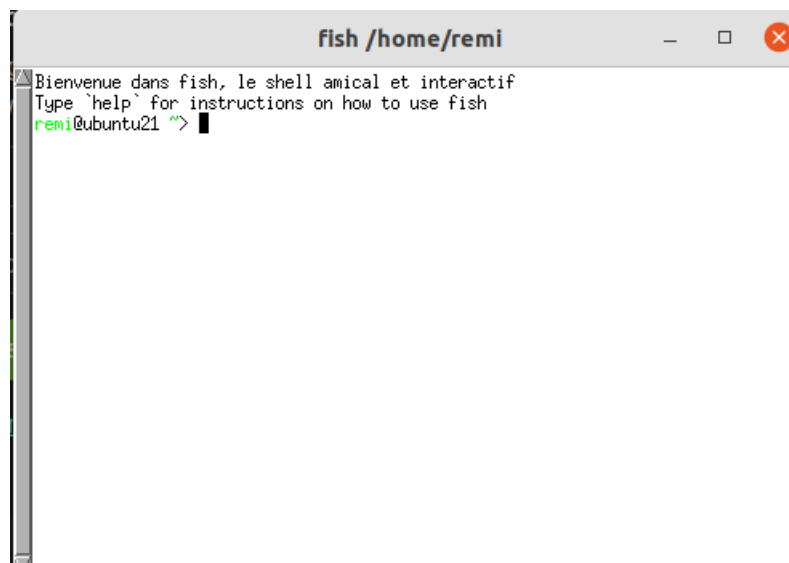
### Sakura :

C'est un terminal assez simple, il est léger, il fait rien de plus que le terminal qui est présent de base sur debian, son avantage un noir profond ce qui fait bien ressortir les couleurs, on a la possibilité d'avoir plusieurs onglets. Couplé au shell fish qui possède une coloration syntaxique et de l'auto complétion basé sur l'historique, ça rend le terminal intéressant.





### urxvt :

C'est un terminal très léger, il prend très peu de ressource, il n'y a pas de possibilité de faire un nouvel onglet. Il est pas très jolie à voir









Tilix :



C'est un terminal beaucoup plus avancé que les autres, on peut le customiser comme on veut, on peut aussi modifier les raccourcis clavier. Les fenetres de ce terminal peuvent etre logé sous forme de mosaïque. Il y a la possibilité que les commandes que l'on tape soit répliqué sur d'autres terminaux. Pour un admin sys, c'est un tres bon terminal.

1 / 1 +  

Tilix: fish /home/remi

  -  



1: top /home/remi  


2: fish /home/remi  

top - 21:51:41 up 9:50, 2 users, load a  
Tâches: 422 total, 1 en cours, 420 en ve  
%Cpu(s): 0,7 ut, 0,4 sy, 0,0 ni, 98,9 i  
MiB Mem : 15940,5 total, 897,2 libr,  
MiB Éch: 2048,0 total, 2043,7 libr,

PID	UTIL.	PR	NI	VIRT	RES
1522	remi	20	0	6427216	385276
69342	remi	20	0	667540	75252
36296	remi	20	0	4286616	241436
68428	remi	20	0	24,5g	187196
1411	remi	9	-11	2801144	26940
3261	remi	20	0	16,6g	464848
66674	root	20	0	0	0
698	root	-2	0	0	0
3318	remi	20	0	323460	11732
3416	remi	20	0	16,7g	209884
3420	remi	20	0	16,4g	101892
5034	remi	20	0	36,8g	166040
5217	remi	20	0	41,1g	367236
69580	remi	20	0	22064	4560
1	root	20	0	164944	11264
2	root	20	0	0	0
3	root	0	-20	0	0
4	root	0	-20	0	0

tmpfs7,8G269M7,6G4  
% /dev/shm  
tmpfs5,0M4,0K5,0M1  
% /run/lock  
tmpfs7,8G07,8G0  
% /run/qemu  
/dev/nvme0n1p296M31M66M32  
% /boot/efi  
tmpfs1,6G18M1,6G2  
% /run/user/1000  
remi@ubuntu21 ~>

3: fish /home/remi  



Bienvenue dans fish, le shell amical et interactif  
Type `help` for instructions on how to use fish  
remi@ubuntu21 ~>

## 2 SSH

### 2.1 Première connection en SSH

Pour se connecter en ssh a srv on entre cette commande :

```
sudo ssh carol@10.0.0.3
```

on est bien en ssh sur cette machine car ce n'est pas mon ip qui est affiché

```
carol@srv:~$ ip addr
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:29:80:01 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.3/24 brd 10.0.0.255 scope global eth1
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe29:8001/64 scope link
        valid_lft forever preferred_lft forever
```

### 2.2 Public key

ssh-keygen -b 4096 va nous créer une clé, on lui indique que l'on veut la stocker dans un fichier spécifique, dans mon cas le fichier id\_rsa.

Cette clé une fois créée on la transfère à la machine de destination scp

```
.ssh/id_rsa.pub bob@10.0.0.3 ./clepub.txt
```

 il ne reste plus qu'à copier cette clé dans le fichier .ssh/authorized\_keys

```
cat clepub.txt > .ssh/authorized_keys
```

Une passphrase c'est une phrase qui est demandée lors de notre connexion en ssh, elle permet de protéger la clé publique