

Linux Utilisation

Module 07 – Éditer du texte sous Linux



1

Éditer du texte sous Linux



Objectifs

- Comprendre Vim et ses modes de fonctionnement
- Savoir ajouter ou supprimer du texte
- Utiliser les commandes spécifiques de Vim
- Gérer les différences des fichiers texte Windows et Linux

*Savoir écrire et modifier des
fichiers texte sous Unix Linux*



2

Éditeurs de texte sous Unix/Linux



3

Éditeurs de texte sous Unix/Linux Présentation des éditeurs de textes sous Linux

Nom	Présentation
Emacs	Éditeur, ou plutôt famille d'éditeurs, ouvert, extensible, riche de nombreuses bibliothèques. Il est destiné à des programmeurs.
Jed	Éditeur de texte semi-graphique avec coloration syntaxique. Émulation Emacs
Jedit	Éditeur de texte graphique avec coloration syntaxique et utilisant de multiples plugin. À destination des programmeurs
Joe	joe, acronyme récursif de Joe's Own Editor. Editeur en mode shell. Basé sur Emacs.
Nano	Éditeur léger, avec coloration syntaxique. Souvent apprécié des débutants, il est moins puissant qu'un Vim ou Emacs.
Geany, Gedit, Kate, Xed ...	Éditeurs en mode graphique, avec coloration syntaxique. En général implémentés dans diverses distributions et/ou environnements graphiques. Équivalent d'un notepad++
Vim	Éditeur natif de Linux, issu de son ancêtre Unix VI, vim est un éditeur puissant, utilisé autant par les administrateurs systèmes que les programmeurs. Implémentant nativement les commandes ex. Il est l'éditeur mis en avant dans cette formation.



4

- Natif dans tout système Unix/Linux
- Éditeur par défaut de certaines commandes
- Nommé VI sous Unix, les distributions Linux implémentent une déclinaison nommée Vim
- Vim apporte la coloration syntaxique, plus d'options pour l'espace de travail, la possibilité d'annuler plusieurs commandes

```
# some more ls aliases
alias ll='ls -l'
#alias la='ls -A'
#alias l='ls -CF'
alias vir='vim -R'

# Alias definitions.
# You may want to put all your additions into a separate
# file like ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc
# package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

-- INSERTION --      89,1      89%
```




Découverte de Vim et premières manipulations



vi [-R] [+n] [+/motif] <fichier> [<fichier2>] [<fichier3>]

vim [-R] [+n] [+/motif] <fichier> [<fichier2>] [<fichier3>]

- -R Ouvrir le fichier en lecture seule.
- +n Ouvrir le fichier et positionner le curseur en début de la ligne *n*
- +/motif Ouvrir le fichier et positionner le curseur sur la première occurrence du mot *motif*

 user30@deb:~\$ **vi** Edition



Zone de travail

```
Aboaf Maurice 244748
Adda Jen 239234
Allo Jean-Pierre 255398
Ben Raf 238899

Bernard Jean-Paul 234567
ChasserAt Paul 245678
Cousin Pascal 222222
Froideceaux Michel 252423
Gros Lucien 212121
Cayrol Patrice 123456
Beux Leon 567891
Dupont Jean 111111
Dupont Jean 111111
Dupont Patrice 222222
```

7,15 Haut

Ligne de
commande de vim



- Vim utilise deux modes principaux
 - Mode commande : Déplacement du curseur, utilisation du presse papier, appel à la ligne de commande
 - Mode Insertion : Saisie du texte par l'utilisateur.
Se remarque par une indication sur la ligne de commande

```
Aboaf Maurice 244748 Responsable Commercial
Adda Jen 239234
Allo Jean-Pierre 255398
Ben Raf 238899

Bernard Jean-Paul 234567
ChasserAt Paul 245678
Cousin Pascal 222222
Froideceaux Michel 252423
Gros Lucien 212121
Cayrol Patrice 123456
Beux Leon 567891
Dupont Jean 111111
Dupont Jean 111111
Dupont Patrice 222222
-- INSERTION --
```

1,44 Haut

9

Déplacer le curseur

cde	Résultat	cde	Résultat
← ou h *	Vers la gauche	w *	Début du prochain mot ou ponctuation
↓ ou j *	Vers le bas	b *	Début du mot précédent ou ponctuation
↑ ou k *	Vers le haut	e *	Fin du prochain mot ou ponctuation
→ ou l *	Vers la droite	:n	Se déplacer à la ligne n
0	Premier caractère de la ligne	gg ou :0	Début du fichier
^	Premier caractère du premier mot de la ligne	G ou :\$	Fin du fichier
\$	Dernier caractère de la ligne	%	Se déplacer à la prochaine accolade ou parenthèse

* : Ces commandes peuvent être multipliées par le nombre n .
5↓ pour descendre de 5 lignes par exemple

10

Rechercher un mot dans le texte

<code>/motif</code>	Rechercher le terme "motif"
<code>n</code>	Se déplacer vers la prochaine occurrence
<code>N</code>	Se déplacer vers l'occurrence précédente

Utiliser la ligne de commande

<code>:w</code>	Enregistrer les modifications
<code>:wq</code>	Enregistrer et quitter Vim
<code>:x</code>	Autre manière pour enregistrer et quitter
<code>:q</code>	Quitter un fichier sans modification
<code>:q!</code>	Quitter en ignorant les modifications



Passer en mode insertion

<code>a</code>	insertion après le caractère courant
<code>i</code>	insertion avant le caractère courant
<code>o</code>	insertion au début d'une nouvelle ligne suivant l'actuelle

<code>A</code>	insertion à la fin de la ligne courante
<code>I</code>	insertion au début de la ligne courante
<code>O</code>	insertion au début d'une nouvelle ligne précédant l'actuelle

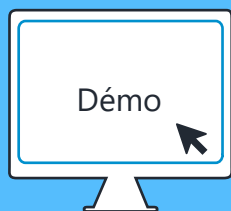
Sortir du mode insertion

Simplement par la touche [echap]



<code>u</code>	Annuler la dernière action
<code>[ctrl]+r</code>	Refaire la dernière action annulée
<code>:n</code>	Passer au prochain fichier ouvert
<code>:N</code>	Passer au précédent fichier ouvert
<code>:w fic01</code>	Enregistrer le contenu de l'espace de travail dans <code>fic01</code>
<code>:!cmd</code>	Exécuter la commande du shell <code>cmd</code> sans quitter Vim

Le caractère `:` place le focus sur la ligne de commande de vim. Il débute toute utilisation de celle-ci. Il est possible de naviguer dans l'historique des commandes comme on le ferait depuis le shell Bash.



Découverte de VI et de ces principaux modes



Copier, coller, remplacer ou supprimer du texte



15

- `nx` Suppression du ou des `n` caractères suivants
- `nX` Suppression du ou des `n` caractères précédents
- `D` Suppression du reste de la ligne à droite du curseur
- `ndw` Suppression de `n` mots
- `ndd` Suppression de `n` lignes à partir de la ligne courante



16

- np Coller n fois le contenu du tampon à la suite du curseur
- nP Coller n fois le contenu du tampon avant le curseur
- nYw Copier les n mots suivant le curseur
- nYY Copier les n lignes suivantes



- nx Couper n caractères suivants
- nX Couper n caractères précédents
- nD Couper le reste de la ligne
- dw Couper n mots
- ndd Couper n lignes



La dernière suppression est stockée dans un tampon mémoire appelé tampon d'annulation.



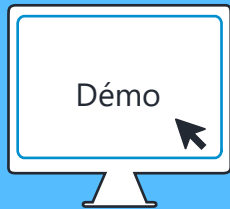
- Il est possible d'utiliser d'autres tampons d'annulation que celui par défaut en précisant des tampons de **a** à **z** précédés du caractère **"**.
- **"lyy** Copier la ligne courante dans le tampon **l**
- **"lp** Coller le contenu du tampon **l**
- Les tampons d'annulation sont internes à une instance de Vim
- Il est possible d'utiliser les tampons dans tous les fichiers ouverts dans une instance de Vim
- À la fermeture de Vim, ces mémoires sont vidées



- **ncl** Effacer(couper) **n** lettres et passer en mode insertion
- **ncw** Effacer (couper) **n** mots et passer en mode insertion
- **ncc** Effacer (couper) **n** lignes et passer en mode insertion
- **rl** Remplace le caractère sous le curseur par le caractère **l**
- **R** Passer en mode Remplacement (Mode insertion ou le texte saisi remplace le texte actuel)



Copier, coller, remplacer ou supprimer du texte



•
Manipulation de texte
•



21

Éditer du texte sous Linux

Le pouvoir de G



22

- Les commandes utilisent globalement la forme ci-dessous :
- `:<lignes> g/regex de recherche/commande de modification`
- La partie "commande de modification" peut prendre plusieurs formes

<code>s/regex/regex/g</code>	Substitution
<code>m n</code>	Déplacement
<code>d</code>	Suppression



- Pour supprimer une ligne en fonction de la recherche

 `:g/motif/d`

- Même chose mais uniquement des lignes 15 à 17

 `:15,17 g/motif/d`

- Pour déplacer le résultat de la recherche à la ligne 10

 `:g/motif/m 10`



- Pour la commande de substitution, la syntaxe complète est celle-ci

:<lignes> g/rechsign/s/rechmod/nouveau/g

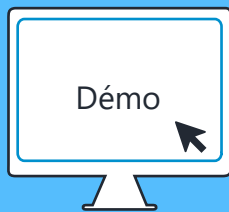
:30,45 g/demi-dieu/s/heracles/achille/g

- Des versions plus courtes

:g/recherche/s//nouveau</g>

:g/Nantes/s//Rennes/g

:g/Niort/s/^#//



Le pouvoir du G



Gérer l'environnement de Vim



27

- `:set` Afficher les options personnalisées
- `:set all` Afficher toutes les options disponibles
- `:set option` Activer une option
- `:set nooption` Désactiver une option
- `:set option?` Afficher le statut d'une option



28

Quelques options communes de Vim

- `compatible` Activer le mode compatible de vi pour vim
- `showmode` Afficher le mode de travail sur la ligne de commande
- `autoindent` Indentation automatique, utile en programmation/scripting
- `number` Afficher les numéros de lignes
- `ignorecase` Pas de distinction minuscule / majuscule lors des recherches
- `tabstop` Définir le nombre de caractères d'une tabulation (par défaut 8)



29

Coloration syntaxique

- `:syntax on` (uniquement) Activer la coloration syntaxique (Vim)
- `:syntax off` Désactiver la coloration syntaxique

Fichier de paramètres

Le fichier `~/.vimrc` (sans `:"`) Résultat (exemple d'un script)

```
set nocompatible
set number
syntax on
```

```
1 #!/bin/bash
2 echo "Hello World!"
3 if [[ $LOGNAME != "root" ]] ; then
4     echo "Bonjour $LOGNAME"
5 fi
~ -- INSERTION -- 6,1 Tout
```



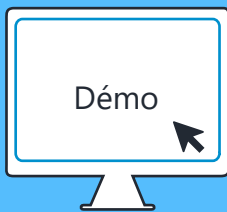
30

- La commande `:help` (version courte `:h`) permet de se rendre dans l'aide (en anglais) de Vim.
- Lancée seule, la commande nous amène sur la page "VIM - main help file"
- En général, on l'utilise avec un terme à rechercher : `:help insert`
- Pour sortir, comme dans Vim : `:q`

 <https://vimhelp.org/>



31



•
Personnaliser Vim
•



32

Les fins de lignes dans les fichiers textes



33

Les fins de ligne dans les fichiers de textes
Linux vs Windows



[CR] [LF]
\r\n



[LF]
\n



34

- Dans .vimrc activer l'option `fileformats=unix` (version courte `ffs=unix`) pour visualiser ce caractère. S'il est présent, il sera représenté par les caractères `^M`

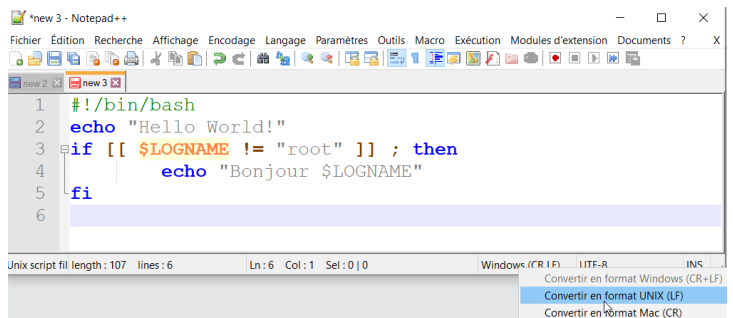
```
Module 2 et Module 3^M
Demo^M
Fiches TP^M
```

- Passer le fichier au format Unix Linux
`:set fileformat=unix` (version courte `ff=unix`)



- Deux groupes de commandes linux permettent de convertir les fichiers
`dos2unix` et `unix2dos` du paquet `dos2unix`
`fromdos` et `todos` du paquet `tofrodos`

- Sous Windows la plupart des éditeurs de textes proposent de faire la conversion des fichiers directement





TP

•

•



- Vous savez éditer des fichiers avec Vim
- Vous savez utiliser la recherche/substitution de la ligne de commande ex dans Vim
- Vous savez transformer un fichier texte au format Windows en format Unix/Linux

