L'éditeur VI

Jean-Marc LICHTLE

11 juillet 2004

Table des matières

1 Introduction

1.1 Intérêt de VI

VI est l'éditeur de base en environnement UNIX, on le retrouve donc dans toutes les versions de ce système d'exploitation et donc dans tous les systèmes dérivés, qu'il s'agisse de LINUX, BSD ou d'autres produits apparentés. Alors qu'il est généralement en compagnie d'autres éditeurs, EMACS en particulier, dans des distributions musclées, on le retrouve souvent seul dans les micro distributions de LINUX, celles qui tiennent que sur une disquette, comme TOMSRTBT ou sur un mini live-cd comme DAMN SMALL LINUX.

L'apprentissage de VI, au moins des commandes de base, présente par ailleurs un avantage insoupçonné. Qui ne s'est jamais battu avec une page de manuel pour y chercher un mot particulier ? La commande de recherche de mot typique de VI fonctionne aussi lors des consultations de page man, comme elle fonctionne encore lorsqu'on visualise le contenu d'un fichier via more ou less. L'apport est le même dans l'utilisation de dselect, logiciel d'aide à l'installation des packages .deb de DEBIAN. Là aussi la syntaxe de recherche de VI s'applique. D'une façon plus générale, en présence d'un bon soft bien unixien, posez vous toujours la question de savoir si les commandes VI ne s'appliqueraient pas par hasard.

Enfin VI est utilisable dans une fenêtre TELNET ce qui le rend incontournable pour faire de la maintenance de matériel à distance. La question de savoir si son ergonomie est actuelle ou non ne se pose donc pas, VI est simplement incontournable. Pour ma part j'utilise couramment EMACS que je trouve plus agréable, mais j'ai été contraint de me "mettre" à VI pour travailler efficacement à distance sur un serveur LINUX et utilisant comme terminal un PC tournant sous un système d'exploitation tout à fait banal édité du côté de REDMOND (USA). A l'usage je suis obligé d'admettre que l'emploi de VI, même s'il reste moins intuitif que celui d'EMACS, peut être tout à fait agréable. Le présent document formalise donc mes notes et consigne les commandes qui, à mes yeux, sont indispensables pour aborder l'usage de VI.

1.2 Variantes de VI

Selon la distribution vous utiliserez VI ou VIM (improved). Un lien est souvent fait entre VI et VIM de telle sorte à lancer la version étendue de l'éditeur. Il existe par ailleurs des versions graphiques comme gvim, mon propos, même s'il est certainement généralisable à ce produit, sera essentiellement centré sur les versions fonctionnant en mode texte pur, celles qui pourraient, le cas échéans, vous sortir de l'ornière sur un système malade ne pouvant plus faire tourner son serveur graphique. Certaines des commandes décrites ici peuvent ne pas fonctionner sur les versions VI simples présentes sur des distributions plus anciennes ou des micro distributions. Dans le même ordre d'idée l'utilisation de VI via Telnet peut réduire la palette des options offertes.

2 Lancement

VI se lance directement en mode console ou dans une fenêtre terminal (rxvt, xterm ou autre). Une fois lancé VI occupe toute la fenêtre.
Une syntaxe très classique comme vi essai.txt lance VI en affichant le contenu du fichier essai.txt. Si ce fichier n'existe pas VI le crée puis affiche des lignes vides commençant par un symbole ~ . En fait ces lignes sont fictives, le remplissage est purement visuel. Selon la version de VI que vous utilisez et son paramétrage vous aurez peut-être une barre de menu dans le bas de l'écran qui vous donnera quelques (maigres) informations sur la session en cours, nom du fichier édité, position du curseur dans le texte.

3 VI, un éditeur modale

Il s'agit certainement l'aspect le plus déroutant pour le débutant. Au lancement VI se trouve automatiquement en mode commande. Tout de que vous taperez sera interprété comme une commande et non comme du texte à enregistrer . L'autre mode, appelé mode insertion et sert, lui, à entrer du texte. Selon la version de VI utilisée l'indication du mode courant n'apparaît pas clairement dans la barre de menu. Ceci est particulièrement vrai lorsqu'on utilise VI via TELNET. Une bonne pratique consiste donc à ne JAMAIS rester en mode insertion mais à toujours revenir au mode commande dès la fin de la frappe. Ce retour s'obtient en tapant Crtl-C ou ESC.

3.1 Le mode commande

C'est le mode initial au chargement. Deux types de commandes peuvent être entrées, les commandes directes et les commandes " :".
Les commandes directes sont celles qui s'exécutent directement, sur la frappe d'une seule lettre, par exemple y pour copier, p pour coller, d pour effacer.
Les commandes " :" attendent des arguments éventuels et surtout la frappe de la touche entrée pour s'exécuter, exemple :w foo.txt qui demande la sauvegarde sous le nom foo.txt.
Aspect particulièrement précieux, le mode commande de vi présente cette analogie avec certains shells d'offrir un historique de commande consultable avec la flèche "haut". Cet aménagement permet souvent d'éviter de retaper une commande déjà saisie précédemment.

3.2 Le mode insertion

C'est le mode classique qui permet d'entrer du texte. La frappe de i ou de A permet de passer du mode commande au mode insertion. La différence essentielle est que i laisse le curseur à sa place alors que A déplace ce curseur en fin de ligne ce qui est très pratique pour réaliser un ajout. ESC ou CRTL-C permet de revenir en mode commande.

4 Déplacements du curseur

Les déplacements au moyen des touches fléchées donnent parfois, selon le terminal utilisé, des résultats assez déroutants. Un cas classique est l'utilisation de VI avec LINUX, par exemple Mandrake 9.1, sur un PC tout à fait ordinaire, clavier 102 touches etc. Dans ce genre de configuration :

– Les touches fléchées droite, gauche, haut et bas fonctionnent correctement aussi bien en mode commande qu'en mode insertion.
– Idem pour les touches page up et page down.
– Les touches home et fin ont une action limitée à la ligne en cours.

Très en gros ça fonctionne assez bien, hormis qu'il est impossible d'atteindre directement le début ou la fin du texte. Commençons donc par là, en mode commande uniquement :

– gg transporte le curseur au début du texte (home en bon jargon informatique)...
– et G à la fin du texte.

Nous voilà donc sauvés, nous avons retrouvé tous nos mouvements, du moins les plus courants. En plus de ces commandes simples et (relativement) intuitives, VI permet aussi le déplacement en mode commande au moyen des touches hjk et l :

– h pour gauche
– k pour haut
– j pour bas
– et l pour droite.

Évident non ? Non ? Bon, h et l ne devraient pas poser de problème, pour k et j je vais vous donner un truc, k déplace la ligne d'écriture vers le haut, j déplace vers le bas. Vu ? Non ? Simple, la barre du k va vers le haut, la patte du j vers le bas !

Par ailleurs (toujours en mode commande) :

– 0 (zéro) permet d'atteindre le début de ligne,
– $ - la fin de ligne,
– w permet de sauter au début du mot suivant,
– b - au début du mot précédent.

5 L'aide en ligne, la manipulation des fenêtres

L'aide en ligne ouvre une nouvelle fenêtre qui va diviser l'écran en deux parties égales. J'ai donc choisi de traiter ensemble les deux sujets, aide et manipulation des fenêtres.

5.1 Aide en ligne

Une aide est accessible en tapant :help en mode commande. Un petit mode d'emploi de l'aide s'affiche alors, malheureusement en langue anglaise. Pour quitter l'aide taper :q ou :q ! La recherche d'aide sur un sujet donné est possible en tapant :help sujet, par exemple :help textwidth. Si on ignore la syntaxe exacte de la commande qu'on recherche il est possible de faire suivre la frappe partielle du mot par Ctrl-d ce qui va afficher toutes les entrées possibles correspondant à le demande formulée. Par exemple :help width suivi de Ctrl-d va provoquer l'affichage d'une douzaine de possibilités parmi lesquelles textwidth, mais aussi screenwidth etc.

5.2 Manipulation des fenêtres

Pour passer de la fenêtre d'aide à la fenêtre courante (ou pour faire passer le curseur entre deux fenêtres d'édition) taper Ctrl-w j pour aller à la fenêtre en dessous, Ctrl-w k pour la fenêtre au-dessus.
Ces sauts permettent de copier très simplement des portions de texte d'une fenêtre à l'autre. D'autres commandes méritent un instant d'attention :
:split permet de créer une deuxième fenêtre dans l'écran actif.
:close ferme la fenêtre active.
:only ferme toutes les fenêtres sauf la fenêtre active.
Par défaut :split ouvre une fenêtre supplémentaire contenant le même fichier. Vous pouvez aussi préciser le nom du fichier à ouvrir à la suite de :split, par exemple :split essai.txt. Cette manipulation semble anodine mais c'est la façon la plus simple de récupérer une ou plusieurs lignes d'un fichier existant pour les copier dans le fichier en cours d'édition. Exemple : Vous êtes en train de taper du code et vous souhaitez récupérer quelques lignes de code bien tortueux que vous avez déjà mis au point dans un autre contexte. Procéder comme suit :

– Passez en mode commande (ESC).
– Tapez :split nom_fichier.php pour charger le fichier nommé nom_fichier.php dans une demi-fenêtre qui apparaitra à l'écran.
– Votre curseur est passé par défaut dans la nouvelle fenètre, sélectionnez la partie de texte souhaitée et copiez là.
– Tapez Ctrl-w j pour faire passer le curseur dans la fenêtre du bas, (Ctrl-w k pour le faire passer en haut) ou alors Ctrl-w w pour le faire changer de fenêtre, cette dernière commande fonctionnant en flip-flop c'est à dire que répétée une nouvelle fois elle fera revenir le curseur au point de départ.
– Coller le texte copié à l'endroit souhaité.
– Il suffit maintenant de repasser en mode commande et de taper :only pour que le fenêtre supplémentaire ouverte tout à l'heure se referme et que la fenêtre d'édition initiale reprenne sa taille d'origine.

Les commandes de copie et collage sont vue par ailleurs.

6 Lecture et enregistrement de fichier

:r foo.txt lit le fichier foo.txt et l'insère à la position courante du curseur
:w foo.txt enregistre l'état actuel du fichier dans foo.txt
:e foo.txt charge (ou recharge) le fichier foo.txt.
Cette dernière commande peut agir comme un puissant undo, surtout quand elle est associée au point d'exclamation qui provoque le rechargement forcé.

7 Les commandes d’édition de texte

7.1 Insertion et suppression

La commande i est la commande de base pour passer du mode commande au mode insertion. Elle présente deux variantes intéressantes, I et A. La première ajoute du texte en début de ligne, la seconde en fin de ligne. A est donc très pratique pour continuer une phrase en chantier en partant du mode commande.
d efface la ligne courante et la colle dans le tampon d’effacement. Cette commande est particulièrement utile pour supprimer une ligne blanche.
x efface le caractère situé sous le curseur
v permet de passer en mode visuel, en clair marque le début d’une zone de sélection qui pourra être effacée par d par ex.
y copie la zone sélectionnée
d efface la zone sélectionnée et la colle dans le tampon.
p colle le dernier tampon d’effacement
"3p colle le 3ème tampon (en remontant). Attention, cette frappe doit se faire en aveugle et en mode commande.
La touche INS permet de commuter entre le mode insertion et le mode "refrappe". Si nécessaire vous pouvez passer en mode refrappe en tapant R depuis le mode commande.

7.2 Annulation et répétition

:u ou :undo annule la dernière modification.
. (point) répète la dernière commande, plus précisément le séquence de frappe depuis le dernier passage en mode insertion. A utiliser avec doigté, exactement le genre de truc qui peut provoquer un bazar de première grandeur, n’oubliez pas :u qui peut vous sauver la mise le cas échéant.

7.3 Recherche de texte

/foo recherche de la séquence de caractère foo en avant
?foo recherche en arrière
n permet de recommencer la recherche, c’est à dire de trouver l’occurrence suivante.
N recherche l’occurrence précédente.
fx recherche d’un caractère (x) à droite sur la ligne.
Fx recherche d’un caractère à gauche sur la ligne.

7.4 Formatage de texte

7.4.1 Coupure de fin de ligne

Ce paragraphe ne concerne que vim.
Par défaut VI "engrange" le texte et le coupe tout simplement en bout de ligne sans se demander si la coupure affecte le milieu d’un mot ou non. Le moins que l’on puisse dire est que ce genre de

comportement est assez désagréable. Un petit coup de :set linebreak remet un peu d'ordre à tout cela. Ici aussi une version courte de la syntaxe est possible sous la forme :set lbr. L'effet de cette commande est immédiat et affecte l'intégralité du texte contenu dans le tampon d'édition. Cette modification de présentation n'a aucune conséquence sur le texte enregistré. A supposer qu'on édite à nouveau le texte quelques temps plus tard il faudra à nouveau taper cette commande, sauf à modifier le fichier de configuration de VI, mais c'est le sujet d'un autre paragraphe .

7.4.2 Largeur du texte

Fixer une largeur Ce paragraphe ne concerne que les utilisateurs de VIM.
Il peut arriver que l'on souhaite limiter la largeur d'un texte, par exemple à 50 colonnes. Pour cela taper :set textwidth=50. Cette syntaxe peut d'ailleurs être résumée en contractant textwidth en tw ce qui donne :set tw=50. Tout le texte qui sera tapé à partir de maintenant sera formaté à 50 colonnes.
Attention : il y a une différence fondamentale entre la coupure de ligne et le formatage en largeur et ignorer cette différence peut conduire à des "prises de tête" assez sévères. La commande linebreak ne fait rien d'autre que changer le comportement de l'affichage en coupant les phrases après des blancs pour ne pas sectionner les mots n'importe ou. En apparence textwidth fait de même mais l'apparence est trompeuse ! En fait textwidth va rajouter des sauts de lignes à la fin de chaque ligne. Ce sauts de lignes vont conduire à des modifications substantielles du comportement de VI :

– Si on utilise les touches flèchées le curseur va passer de ligne en ligne au lieu de passer de paragraphe en paragraphe comme il ne faisait d'habitude. Le saut de ligne est en effet interprété comme une fin de paragraphe
– Si on corrige le texte en ajoutant ou en supprimant des caractères, des mots, les sauts de lignes vont rester en place et mettre un b... (bazar) assez sensationnel !
– Si vous sauvegardez le texte et l'affichez avec cat (par exemple), vous vérifiez immédiatement que lbr est sans conséquence sur le texte alors que tw le modifie.

Il vaut donc mieux savoir ce qu'on veut faire avant de se lancer.
Ce point étant précisé avançons encore un peu dans l'étude de set tw. Pour formater un paragraphe placez le curseur dans le paragraphe et tapez :set tw=45 (exemple). Rien ne se passe dans l'immédiat. Il faut taper gqap pour que la nouvelle largeur soit prise en compte. Une autre possibilité, à mon avis plus sympathique est décrite ci-dessous :

– Tapez set tw=.. le curseur étant placé n'importe ou dans le document.
– Utilisez la commande v (Visuel) pour surligner le passage à modifier
– Tapez gq pour mettre en place la mise en forme

Notez que vous pouvez très bien modifier les largeurs de passages différents sans retaper set tw, la valeur étant mémorisée elle s'applique à chaque commande gq ou gqap.

Supprimer l'action de tw Panique à bord, les lignes sont coupées n'importe comment, plus rien ne va comme vous voulez, comment faire pour revenir au bon vieux texte qu'on s'était donné tant de mal à taper ? Du calme, la documentation de VI vous indique que vous pouvez fixer la valeur de tw à zéro pour annuler les effets précédents. Nouvelle déception, cette fois les lignes "wrappent" à 79 colonnes maxi ! Ma méthode dans ce cas est très rustique, n'ayant pas trouvé la

façon de me débarasser de tw je le porte simplement à une valeur loufoque, 9999 par exemple et je reformate le texte.

7.4.3 Supprimer un saut de ligne

Que vous soyez dans le cas décrit plus haut ou que vous souhaitiez simplement "raccorder" entre elles deux lignes séparées par un saut de ligne (une frappe sur la touche ENTER) les méthodes suivantes vont vous rendre service.
En mode insertion il suffit de se placer en fin de ligne et de taper SUPPR pour effacer le saut de ligne non désiré.
En mode commande c'est encore plus simple puisqu'il suffit, en positionnant le curseur n'importe ou sur la première ligne, de taper "J" (majuscule) pour que le raccordement se fasse automatiquement ! Nul besoin d'aller à la fin de la ligne. Dans le cas décrit ci-dessus il suffit donc simplement, en étant sur la première ligne de texte "amputé" par tw, de taper "J" sur la première ligne et de garder le doigt enfoncé jusqu'à ce que toutes les lignes aient recollé au peloton.

7.5 Copier et coller du texte

Le copier / coller est certainement la fonction la plus utile dans un éditeur de texte. Elle est donc bien évidemment bien représentée dans VI.

7.5.1 Copier

La suppression d'une ligne au moyen de la commande dd efface bien entendu la ligne visée, mais elle copie aussi son contenu dans un tampon qui pourra être récupéré par la suite. A ce titre dd est donc la permière méthode de copie. Pour le reste je propose d'utiliser le mode VISUAL pour surligner le texte à copier après quoi "d" ou "y" copie la partie surlignée dans le tampon, avec suppression du texte orginal dans le cas de "d". L'emploi de VISUAL est très simple, passer en mode commande, déplacer le curseur sur le premier caractère à copier, taper "v" pour signifier l'entrée en visual, déplacer le curseur jusqu'au dernier caractère à copier. Attention, le dernier caractère qui sera copié est celui sur lequel clignote le curseur. C'est un peu déroutant au début et puis on s'y fait.

7.5.2 Coller

Le collage se fait en partant du mode commande. Il suffit de positionner le curseur sur le caractère qui précède l'endroit ou on souhaite coller le texte après quoi une frappe sur "p" colle le texte.
Toute le problématique du copier coller est donc de se souvenir que le texte est enlevé (ou copié) JUSQU'AU curseur clignotant et qu'il est collé APRES le caractère sur lequel on arrête le curseur pour positionner le collage. Si vous gardez à l'esprit que avec VI la lettre y (yank) est utilisée pour la copie alors que, couramment on l'utilise plutôt pour le collage (EMACS) alors vous serez paré pour le sujet.

8 Quitter et enregistrer

:w enregistre le fichier avec son nom par défaut.
:q quitte, sauf si le fichier n'a pas été sauvegardé auquel cas VI émet un message d'erreur.
:q ! force l'abandon de l'édition non enregistrée et ferme l'application.

9 Sortie provisoire de VI pour accéder au shell

L'accès au shell est possible à tout moment au moyen de la commande :sh ou :shell. VI va éventuellement vous adresser un message vous signalant que votre fichier de travail n'est pas encore sauvegardé. Une fois l'activité shell terminée vous pouvez reprendre le cours de votre travail d'édition en tapant simplement exit dans la fenêtre ainsi ouverte.

10 Paramétrage permanent de VI

Le paramétrage de VI s'effectue soit dans un fichier global valable pour tous les utilisateurs et généralement enregistré sous /etc/vimrc (/usr/share/vim/vimrc avec Mandrake 9.2, /etc/vim/vimrc avec Debian Woody). Un utilisateur peut installer un fichier .vimrc dans son répertoire personnel de façon à y enregistrer ses préférences. Il suffit de copier vimrc en le renommant .vimrc. Le '.' est important, sans lui le fichier n'est pas lu. Pour illustrer mon propos je suggère d'ajouter la ligne :

```
set linebreak
```

en fin de fichier. Cette simple modification suffit à rendre permanente la modification de paramétrage qui conduit à des coupures de lignes propres comme il a été exposé plus haut au paragraphe 7.4.1.

11 L'auteur

Jean-Marc LICHTLE, ingénieur Arts et Métiers promotion CH73. J'avance pas à pas dans la découverte de LINUX en mettant un point d'honneur à mettre au propre mes notes de travail. C'est ainsi que j'ai rédigé différentes contributions sur des thèmes aussi variées que :
– l'utilisation de la commande rpm
– l'installation et la sécurisation d'un serveur Apache PHP MySQL
– l'utilisation d'un Minitel comme terminal de secours
– la mise en oeuvre de Gnuplot
– une introduction à LaTeX
– une étude sur la programmation des micro contrôleurs PIC16F84 avec une machine LINUX
et quelques autres documents de la même eau. Mon plaisir ? Recevoir un soir un mail d'un lecteur à qui mon travail a rendu service. Ce qui me met en rage ? Trouver un de mes papiers copié in extenso à ceci près que mon nom a été simplement remplacé par celui du copieur. Depuis ce temps il a droit à quelques lignes dans chacune de mes contributions.

La copie partielle ou totale de ce document est autorisée avec cette simple restriction, plus une marque de politesse en fait, que mon nom reste associé au texte que j'ai créé surtout si la copie réalisée représente une portion importante du travail que j'offre à la communauté.

Index