Guide de survie sous VI
ou
VI par l'exemple.

Charles VIDAL
vidalc@univ-mlv.fr

14 juin 1996

1 VI KESAKO???

Vi est un éditeur visuel (prononcé Viaï), à l'opposé des éditeurs non visuels, par exemple sed, ex, ed, edlin.
Vi a été écrit par William Joy comme une part de la distribution BSD d'UNIX. Puis At&T l'a utilisé, et maintenant c'est devenu un standard sous Unix.
Cette standardisation de l'éditeur VI permet donc de retrouver des versions de VI sur tout les systèmes actuels :
Unix, AmigaOS, ST-OS, DOS, OS-2, SYSTEM 7, WIN3.1, WIN95, et bien d'autres futures (BE-OS).
Ceci dit les versions de VI varient un peu, certaines commandes ne marchent pas forcément de la même manière partout.
Pour apporter de l'eau à mon moulin, j'ai utilisé VI sur un terminal X, un terminal vt100, Win95, DOS, et même un minitel (c'est dire).

2 Pourquoi apprendre VI?

C'est vrai, VI est pénible à apprendre, surtout aujourd'hui ou les nouveaux éditeurs sont tous munis d'un minimum d'interface graphique.
Mais la puissance de VI surpasse de beaucoup d'autres éditeurs plus conviviaux.
On peut quasiment tout faire sous VI, la difficulté est de savoir comment VI peut le faire.
Je suis un peu prosélyte à propos de VI (je pourrais l'être a propos d'Emacs (Emacs make another computer slow), mais là n'est pas le propos), et voyant un ami utilisant un éditeur graphique (MOTIF/OSF), gourmand en ressources, je lui dis qu'il devrait utiliser VI.
Ajoutant que j'utilisais VI sous DOS, il me rit au nez, traite VI des pires maux, et ose même le comparer à edit (editeur du DOS).
Aprés 10 minutes, lui montrant une petite partie des possibilités de VI, fonctionnalités impossibles sous edit, et même sous la pluspart des éditeurs DOS même borland, il abdique, et me demande ou se procurer la version DOS.

J'espère vous aider à appréhender VI, et vous faire votre propre idée.

3 L'utilisation et les piéges à éviter de VI

Attention, au premier contact de VI, il est probable que vous soyez déconcerté, totalement perdu.
Il y a certaines choses à connaître.
L'interface avec l'utilisateur se fait uniquement par le biais du clavier. Donc, ne vous énervez pas sur celui-ci, la situation ne fera qu'empirer .
la touche DEL n'efface pas les caractères, ça trouble beaucoup au début.
VI a deux modes :

- *un mode* **INSERT** on tape 'v', un 'v' s'insère dans le texte .
- *un mode* **COMMAND** on tape ':q', on quitte vi .

Toute la difficulté du débutant est de savoir dans quel mode il est.
En cas désespéré, appuyez sur ESC puis tapez ":se smd", ou ":se showmode".
Cette commande, montre en bas à droite, le mode où l'on se trouve.

Au lancement de VI, on est dans le mode **COMMAND**, par défaut.
Pour passer du mode **COMMAND** au mode **INSERT**, il y a plein de touches, mais la plus utilisée est 'i' pour mode 'I'NSERT !
Pour passer du mode **INSERT** au mode **COMMAND**, ici une seule touche ESC (ou CTRL-C).
Une fois en mode **COMMAND**, on peut éxecuter une commande de différentes façons.

Exemples:
dd supprime la ligne courante.
5dd supprime 5 lignes à partir de la ligne courante.
:3,5dd supprime de la ligne 3 à la ligne 5 .
Après une commande commençant par ":" ou "\", appuyer sur ENTER pour l'exécuter.
Beaucoup de commandes peuvent s'exécuter de la même façon.
Il y a des exceptions :
Par Exemple : w vous positionne sur le prochain mot, alors que ":w" sauve le fichier.

Attends, mais si j'ai pas de touche ESC sur mon clavier,comment utiliser VI?

Essaie CTRL-[, CTRL-3 ,ou CTRL-C, sous DEC terminal, F11 est utilise comme touche ESC.

4 Premier pas...

lancez le programme vi (elvis ou autre clone).

l'écran s'efface.
appuyez sur 'i' (pour passer en mode **INSERT**), si en bas de l'écran s'affiche insert, sautez la ligne suivante.
Sinon appuyer sur ESC puis tapez ":se showmode" ENTER puis sur i.
Frapper x, un "x" apparait à la première ligne (sinon recommencer depuis le début) frapper ensuite ESC (maintenant vous êtes en mode **COMMAND**), puis x, le "x", qui a été précédement inséré, disparait.
Vous devez comprendre, maintenant, la différence entre le mode **COMMAND** et le mode **INSERT**.
En mode **INSERT**, x insère le caractere "x".
En mode **COMMAND**, x, qui n'est plus le caractere mais une commande, supprime le caractere courant. Pour finir, étant toujours en mode **COMMAND** frapper ":q!" .Voila vous êtes sorti de VI
BRAVO
Faites la même operation mais à la fin taper ":w" toto puis ":q", le fichier toto sera alors créé.

voici quelques commandes

Commandes avec passage en mode **INSERT**	
a	append ajoute après la lettre courante.
A	ajoute après la ligne courante.
o	ajoute une ligne en dessous de la position courante
I	insère en début de ligne.
cw	change le mot sur lequel on se trouve
Commande sans passage en mode **INSERT**	
x	en mode **COMMAND** enléve le caractere à la position courante
:10	la ligne courante devient la ligne 10
dw	supprime le mot sur lequel on se trouve
:q	quitte
:w	sauve
:q!	quitte sans sauver..

Cette liste n'est pas exhaustive, c'est un minimum pour débuter.
D'autres commandes seront données dans la suite du document.
Mais au début, ne soyez pas trop ambitieux, maitrisez les comandes ci-dessus, vous serez déja très bon.

5 Exemples d'utilisation

Envoi

vi	édite un nouveau fichier sans nom.
vi toto.c	édite le fichier toto.c, s'il n'existe pas il sera créé à la prochaine sauvegarde.
vi *.txt	édite tous les fichiers *.txt. Pour passer au suivant faire en mode commande :n et :N pour passer au précédent.
vi -R fichier1	Lit le fichier fichier1 en READ ONLY.
vi -r	Edite le fichier édité lors du crash du systême.
vi + fichier1	Ouvre le fichier fichier1 à la dernière ligne.
vi +n fichier1	Ouvre le fichier fichier1 à la ligne n.
vi +/pattern fichier1	Ouvre le fichier fichier1 à la 1ere occurence de pattern n.

———————————————————————————

Remarque

Un grand avantage de VI est qu'il est très rapide à s'exécuter. Sur HP-UX, 1 minute pour éditer 15 Mega.

copier-coller

:10,20t 100	copie les lignes de 10 à 20 à la 100 eme ligne .
:10,20m 100	déplace les lignes de 10 à 20 à la 100 eme ligne .
:10,20y	met dans le buffer les 10 lignes de 10 à 20.
:100,$m 50	déplace les lignes 100 jusqu'a la fin du fichier vers la 50ême ligne.
yy	copie la ligne courante dans le buffer .
p	colle le buffer dans le texte.

——————————————————————

Remarque

le couper-coller marche par la commande 'dd' aprés on déplace le curseur puis 'p' (comme paste) coller la ligne coupée; de même avec "10x" pour les caractêres.

6 Substitution, ou VI puissance 10

:%s/window/linux/g	change windows par linux dans tout le fichier.
:%s/windows/linux	change windows par linux dans tout le fichier, mais une seule fois par ligne.
:%s/windows/linux/gc	change windows par linux dans tout le fichier, mais avec confirmation.
:10,20s/windows/linux/g	change windows par linux dans les lignes de 10 à 20.
:s/.*/\U&/	met en majuscule la ligne courante. U majuscule L minuscule
:%s/\^m//	équivalent de la commande dos2unix pour avoir ^m appuyer sur CTRL-V puis m enter .
:s/[a-zA-Z]/1&/g	ajoute après tout caractere dans [a-zA-Z] un 1 devant.
:%s\^windows//	élimine windows en début de ligne. ^veux dire début de ligne $ veux dire fin de ligne
:/windows/-2,+4d	élimine 5 lignes autour de la première occurence de "windows"
:10,20s/$/\\	ajoute à la fin des lignes 10 à 20 un \
:s/[ABC]/[abc]/g	A devient a, B devient b, C devient c.

7 Action contexte sensitif, ou Substitution^{10}

On peut demander à VI de faire une action sur une phrase uniquement si cette dernière contient un motif textuel.
Syntaxe: ":g/chaine/commande"
Exemples :

:g/micrisoft/s/windows/linux/g	changes windows par linux uniquement dans les lignes ou "micrisoft" se trouve .
:g/^[ ]*$/d	supprime toutes les lignes vides.
:g/linux/t 0	copie en haut du texte toutes les lignes contenants linux
:g/linux/cd 80	centre tous les lignes contenant linux
:g!/linux/d	efface toutes les lignes ne contenant pas linux
:v/linux/d	idem

8 Visualisation et recherche

:/linux/-2,+4p	affiche 5 lignes autour de la première occurence de "linux".
\^linux	recherche linux en debut de ligne
:10,20nu	visualisation des lignes 10 à 20 avec affichage des numéros de lignes.
/lin[uU]\?x/	recherche linux ou linUx
n	Recherche la nouvelle occurence.

9 Changement du Mappage du clavier

Les touches des claviers peuvent acquérir une fonctionalité, en mode **COMMAND**.
'v' qui n'en n'a pas, on peut lui en affecter une:
:map v 10dd
Apres la touche 'v' éfface 10 lignes à partir de la position courante, en mode **COMMAND**.
:map! v 10dd
v aura la fonctionnalitée dans les 2 modes.

Exemple:

:map! ^v<DELETE> ^v<ESC>lxi
Maintenant, la touche DELETE supprime dans les 2 modes
de même:
:map! ^v<BACKSPACE> ^v<ESC>hxi
Maintenant, la touche BACKSPACE marche aussi.

10 Entrées-sorties sous VI

:w	sauve le fichier édité.
:w!	force la sauvegarde d'un fichier READ ONLY, si l'on a les droits si on est root.
:e	ouvre le même fichier que celui édité, si on n'as pas sauvé, c'est un puissant undo.
:e#	ouvre l'ancien fichier édité après un :e fichier
:r toto.c	insère le fichier toto.c à la position courante.
:10,20w toto	sauve les lignes de 10 à 20 dans le fichier toto
:10,20w >> toto	ajoute au fichier toto les lignes 10 à 20 du fichier édité.
:r!ls -1	insere le résultat de la commande ls -1 dans l'éditeur à la position courante.

AU SECOURS

:vi	permet de rétablir l'affichage
:q!	permet de sortir sans rien changer.
:e!	permet de rééditer le fichier sans sauver une connerie (ca arrive à tous le monde).
:u	undo annule la dernière insertion.
:se all	affiche toutes les options.

11 Divers

L'appui de la touche '.' en mode **COMMAND** répète soit la derniere insertion ou la derniere commande, ceci est hyper pratique.

la touche $	envoie à la fin de ligne.
la touche ^	envoie au début de la ligne.
la touche w	envoie au début du mot suivant.

r	remplace la lettre courante
cw	change le mot courant.
2cc	change 2 lignes à partir de la ligne courante. pour ces dernières commandes il y a un passage en mode **INSERT**.
:10ce 80	centre la ligne 10 sur 80 colonnes.
:10,20ce 80	centre les lignes de 10 à 20 sur 80 colonnes.
:right	aligne à droite la ligne courante.
:left	aligne à gauche la ligne courante.
:10,20 > 1	insertion d'une tabulation pour les lignes de 10 à 20 < 1 en enlève une.
:10,20j	joint toutes les lignes de 10 à 20 en une seule
:map #7 dd	associe la touche F7 à la commande dd
:color light cyan on blue	cyan sur fond bleu
:10,20w tmp	sauve les lignes de 10 à 20 dans le fichier tmp.
:ab GNU gnu is not unix	abréviation GNU devient gnu is not unix attention à la récursivité (GNU - gnu et non GNU - GNU) plus pratique.
:ab email vidalc@mlv-univ.fr	pratique pour son mail ....
:ab	affiche toutes les abréviations
:unab email	élimine l'abréviation email.
:10,20!sort	remplace le texte (10 -¿20) par la sortie du filtre unix sort dont l'entrée a été ce texte
:so .exrc	exécute toute les commandes VI du fichier .exrc
ma	déplace le curseur (puis marque la position courante par a)
mb	(puis marque la position courante par b)
'a	(puis va à la marque a)
'b	(puis va à la marque b)
:jumps	(puis affiche les sauts)
:marks	affiche toutes les marques du texte

12 Trucs pour les codeurs

Il faut savoir que VI a été fait pour les programmeurs UNIX et il est donc normal de retrouver des fonctionnalitées simplifiant la programmation.
Un langage du premier âge, LISP, (LANGUAGE OF IDIOT AND STUPID PARENTHESES) est un enfer de parentheses.
Heureuseusement il y a VI.
Si dans un programme LISP, vous tapez '%' en mode **COMMAND** (c'est la derniere fois que je le précise), VI vous montrera la première parenthèse ouvrante trouvée, puis de nouveau '%' et hop VI montre la parenthèse fermante correspondante.

Aucun éditeur sous DOS commercial, fait ça!!!
Même pas les éditeurs BORLAND.
C'est pas dur pourtant..., et tellement pratique.

la commande '%' fonctionne avec '{','}', '(',')', '[',']', enfin grossomodo tous les caracteres qui s'ouvrent et se ferment.
Ca marche aussi, avec le C ou C++, ou autre.
Donc pas de problème pour vérifier les accollades, sauf quand elles se trouvent dans une chaine de caractères (dommage, probleme résolu sous Emacs).

Autres trucs

la commande)	avance sur la prochaine phrase.
la commande }	avance sur le prochaine paragraphe.
la commande]]	avance sur la prochaine section.

Ceci permet de sauter de fonctions en fonctions en C ou C++, entre autres.
Mêmes commandes avec les caracteres ouverts, mais le déplacement se fait dans l'autre sens.

Macros pour le C

Pour obtenir le ^[appuyer sur CTRL-V puis CTRL-ESC.
Ce caractère ^[est la simulation ESC, donc passage en mode **COMMAND**.

:map ^A o{^[o}^[O	CTRL-A insère deux acollades et vous positionne entre les acollades en mode **INSERT**.
:ab defoo ^[:0^MO#define	abreviation defoo insert #define en haut du fichier
:ab foo for (){^[o}^[k$5hi	abreviation foo insert for avec parenthèse, et tout et tout..

Utilisation de ctags

Allez dans le répertoire ou il y a vos sources sous un shell quelconque, tapez "ctags *.c" (création d'un fichier tags).
Puis sous VI, dans un programme source tapez en mode COMMAND:
":ta nom d'une fonction"
Et hop vous voila amené dans le fichier contenant ladite fonction, à l'endroit où elle se trouve.

13 Les options

Il existe une commande sous VI, qui n'influe pas sur le texte édité, mais sur l'éditeur VI lui-même.
Cette commande est ":set" ou ":se"
comme ci-dessus, il y a plein de raccourcis clavier.
la syntaxe de cette commande est :

:se	affiche les options modifiées.
:se all	affiche toutes les options possibles
:se option	positionne l'option Booléenne à Vrai (True)
:se nooption	positionne l'option Booléenne à Faux (False)
:se option=*valeur*	positionne l'option à la valeur *valeur*.
:se option?	affiche la valeur de l'option.

voici quelques options très utiles ou loufoques.

Options Booléennes

:se autoindent	
:se ai	après ENTER la ligne suivante commencera à la même indentation.
:se binary	
:se bin	permet d'éditer des fichiers binaires (décidément VI peut tout faire).
:se ignorecase	A la recherche de chaine, VI ignore minuscule majuscule et à la substitution.
:se ic	A la recherche de chaine, VI ignore minuscule majuscule. et à la substitution.
:se number	
:se nu	affiche le numéro de ligne à gauche (très pratique)
:se visualbell	
:se vb	inverse le BEEP par un éffet visual.
:se readonly	
:se ro	écrit un fichier même si c'est protégé en écriture
:se ruler	
:se ru	affiche les coordonnées du curseur (pas sur HP-UX)
:se lisp	montre automatiquement la correspondance des caractères ouvrants et fermants. (marche pas sous elvis)
:se revins	
:se ri	inversion des caractères de droite à gauche (Il peut vraiment tout faire même les truc fous)
:se showmode	
:se smd	indique s'il on est en mode **INSERT** ou **COMMAND**.

Autres options

:se undolevels 100	
:se ul 100	positionne la possibilité de faire 100 fois undo.
:se history 20	
:se hi 20	positionne l'historique des commandes à 20.

14 Bidouille plus complexe

yyp	Duplique une ligne.
y$	Met dans le buffer, tout de la position courante à la fin de la ligne.
ddp	Bouge une ligne vers le bas .
xp	Déplace le caractère courant aprés le caractère suivant.
:map <ctrl-v><up-arrow>k<ctrl-v><ctrl-g>	essaye donc! Ca ralentit beaucoup VI.
:map <ctrl-v><down-arrow>j<ctrl-v><ctrl-g>	idem
:map v Gdd"Pj	inexplicable.
d%	efface jusqu'à la prochaine parenthèse fermante.

15 Le fichier de ressources

A chaque lancement de VI, le fichier .exrc est lu.
La structure du fichier .exrc: <ap|set|ab> options.
Mettez dans ce fichier:
se nu
se ruler

pour avoir les options nu et ruler automatiquement, quand on lance VI.
On peut aussi mettre des touches de commandes personnelles via map.
remarque : ruler ne marche pas partout (sous HP-UX il n'y a pas).

16 Références:

Voici quelques URLs :

`http://fillmore.univ-mlv.fr/~dittgen/Vi/vi.html`

L'excellente page francaise de l'ami Pierre Dittgen.
Tout ce que vous avez voullu savoir sur VI, sans jamais osez le demander.
Il y a sur cette page d'autre doc sur VI en francais.

`http://www.cs.vu.nl/~tmgil/vi.html`

Une page trés sympa mais en anglais.
Il y a aussi les FAQs, et le newsgroup comp.editors.

17 Clone de VI : vile

vile (Vi Like Emacs) est un clone de vi qui apporte des amélorations au VI de base.
vile permet le multifenêtrage, il y a une complétion de comandes, des fichiers à lire etc ...
Mais il a aussi quelques défaut, certaines commandes n'existent pas , comme :
:r! pwd
xvile est une amélioration de vile, car il permet de mettre des couleurs suivant les termes rencontrés (à la Emacs ou borland).

---

un grand merci à pierre DITTGEN pour son aide.

Toutes suggestions sont les bienvenues. Adressez les a:

```
         --
        /  \       charles vidal
       / ..|\
      (_\  |_)     vidalc@univ-mlv.fr
      /  \@'
     /     \       http://fillmore.univ-mlv.fr/~vidalc/
 _  /  `   |
\\/  \  | _\
 \   /_ || \\_
  \____)|_) \_)
```

copyright charles vidal 1996 :)