Vi Vademecum

Francesco Corsentino (Kiko)
kikocorsentino@gmail.com
http://www.kikoweb.org/

April 3, 2007

Abstract

Questo documento non vuole essere una guida esauriente all'uso di VI (ne esistono di migliori e piú complete in rete e nelle librerie) ma una sorta di vademecum, un prontuario, una lista dei comandi piú comuni, per la quale la memoria riserva solo brevi intervalli di tempo, e in fretta si dimenticano. L'idea di scrivere questo documento nasce proprio dalle innumerevoli volte che sono stato costretto a riprendere in mano guide e manuali alla ricerca della giusta combinazione per ottenere un risultato. Quanto trattato é solo ció che l'autore usa piú di frequente, chiunque voglia arricchire il presente documento non ha che contattarmi e saró felice di passare i sorgenti. Spero, in ultimo, possa essere di ausilio a chiunque ne faccia uso.

Contents

1 Introduzione

Vi é il classico strumento di editing e screening per Unix. All'origine era *ex*. Lavorando su una riscrittura di questo e apportando migliorie e creando l'ambiente per sviluppi futuri nacque **vi**. Chi si avvicina a Linux sente anche parlare di *vim*, *vile*, *elvis*. Non sono altro che versioni di vi. In molte distribuzioni, lanciare il comando vi significa anzi richiamare una di queste versioni (quindi un vero e proprio collegamento).

Vi consiste di due modalitá:

- ⇝ modalitá di comando (*command mode*), dalla quale é appunto possibile impartire comandi di editing, invocare ed usare la shell;
- ⇝ modalitá di inserimento (*insert mode*), e questa ci permette di inserire il testo.

In modalitá di inserimento é possibile passare al command-mode premendo il tasto ESC oppure premendo CTRL+[. Da qui, semplicemente premendo il tasto `a` ritorniamo alla modalitá di inserimento.

Piú specificatamente, la sintassi dei comandi *vi* prevede la seguente forma:

```
[x] operator [y] object
```

,dove `x` e `y` indicano il numero di volte per le quali deve essere effettuata l'operazione `operator`, ovvero il numero di oggetti (*object*) sui quali l'operazione viene effettuata. I numeri x e y sono chiaramente facoltativi e indipendenti l'un l'altro (possiamo specificarne uno invece che l'altro, nessuno dei due, o tutti e due nel qual caso l'effetto é un numero di esecuzioni pari a `x` × `y`). Gli object possono rappresentare una parola (*word*, che include i caratteri fino a uno spazio oppure a un carattere di punteggiatura), una frase (*sentence*, e ció significa estendere l'azione fino a un carattere di punteggiatura quale `.` `!` `?` seguito da due spazi), un paragrafo (*paragraph*, quindi fino alla successiva riga vuota) o una sezione (*section*, fino alla successiva sezione).

Dalla riga di comando é possibile lanciare *vi* secondo tre metodi:

`vi file` apre un file per l'editing;

`vi +n file` apre il file alla riga n;

`vi +/pattern file` apre il file alla prima corrispondenza del pattern specificato (nota: i pattern non vengono trattati in questo articolo).

Da notare che se non viene specificato nessun file allora viene avviato un buffer vuoto.

Altre opzioni in fase di avvio sono:

-R i file vengono aperti in sola lettura;

-c *comando* una volta caricato il primo dei file specificati viene eseguito il comando;

-s *file_ comandi* una volta caricato il primo dei file vengono eseguiti i comandi contenuti nel file specificato.

Nella terminologia **vi** un concetto molto importante é l'*oggetto attivo*, dove oggetto sta per carattere/parola/riga/colonna. L'oggetto attivo non é altro che l'oggetto in cui si trova il cursore.

2 Modalitá di inserimento

In tale modalitá la tastiera e tutti i suoi simboli ci permettono di inserire il contenuto del documento. Qualora fossimo in modalitá di comando é sufficiente premere il tasto `a` o `i` per passare a *insert-mode*.

Facendo riferimento ai **vi** piú sofisticati, i tasti della nostra tastiera assumeranno il loro consueto significato:

- ⇝ tasti freccia per spostarci lungo il documento;
- ⇝ tasto enter per terminare una riga;
- ⇝ tasto backspace per cancellare il carattere e tornare indietro;
- ⇝ tasto canc per cancellare il carattere successivo al punto in cui si trova il cursore;
- ⇝ tasto ESC per passare in *command-mode*.

3 Comandi di spostamento

Esistono versioni di **vi** che non permettono di usare le frecce, il tasto enter, il tasto backspace e quindi per ottenere questi risultati bisogna passare in modalitá di comando e qui usare apposite funzioni (appositi comandi). Riportiamo nella tabella seguente i comandi per lo spostamento e altri utili comandi per l'inserimento del testo nel documento.

comando	**azione**
h	sinistra
l	destra
j	basso
k	alto
I	inserisce all'inizio della riga attiva
A	inserisce alla fine della riga attiva
i	inserisce prima della posizione attiva
a	inserisce dopo la posizione attiva
O	aggiunge prima della riga attiva inserendo una riga
o	aggiunge dopo la riga attiva inserendo una riga

Possiamo pure usare i moltiplicatori per ripetere lo spostamento, ad esempio scrivere

`5j` : ci spostiamo di 5 posizioni in basso

In tal senso é anche utile (specie in fase di programmazione) utilizzare i comandi che ci permettono di individuare i numeri di riga:

comando	**azione**
nG	posiziona il cursore alla riga n del file
:n	posiziona il cursore alla riga n del file
CTRL+G	visualizza il numero di riga attiva

La tabella che segue mostra tutti gli altri utili comandi per la navigazione.

comando	**azione**
-	inizio riga precedente
+	inizio riga successiva
b	parola precedente
w	parola successiva
0	inizio riga attiva
$	fine riga attiva
H	prima riga della schermata
M	riga centrale della schermata
L	ultima riga della schermata
G	ultima riga del file
n|	colonna n
CTRL+B	indietro di una schermata
CTRL+F	avanti di una schermata

4 Comandi di cancellazione

Facciamo ancora riferimento a quelle versioni di *vi* che non permettono l'uso di backspace e del tasto canc. Di seguito, allora, i comandi principali:

comando	**azione**
x	cancella il carattere attivo
j	unisce la riga attiva con quella successiva
d	cancella la riga attiva
d*mod*	cancella dalla posizione attiva fino all'estensione individuata dal modificatore

Cosa significa l'ultimo comando? Il modificatore permette di definire gli oggetti cui é applicato il comando. In particolare, associato al comando *d*, il modificatore specifica la zona da cancellare a partire dalla posizione attiva. Qualche esempio:

`dw`: cancella dalla posizione attiva fino all'inizio della prossima parola
`d$`: cancella da posizione attiva fino alla fine della riga

5 Comandi di sostituzione

Sono comandi che permettono modifiche del testo piú complesse e ragionate. Eseguito il comando, si ritorna in modalitá di inserimento:

comando	azione
C	sostituisce dalla posizione attiva fino alla fine della riga
c*mod*	sostituisce dalla posizione attiva fino all'estensione indicata
cc	sostituisce la riga attiva
~	inverte maiuscole e minuscole
r*char*	rimpiazza il carattere attivo con *char*

6 Annullamento comandi

Con **u** annulliamo l'effetto dell'ultimo comando; con **U** annulliamo le modifiche sulla riga attiva.
Pressioni consecutive di **u** o **U** sono realizzate in maniera diversa dalle varie versioni di *vi*: si annulla a ritroso o si ripete.

7 Ricerche

Questa attivitá é piú proficua e interessante se usata con nozioni di espressioni regolari.

Vediamo le cose piú basilari, per adesso.

Il comando */modello* permette le ricerche in avanti, il comando **?***modello* permette le ricerche indietro dalla posizione attiva. Il comando **n** ripete l'ultima ricerca, il comando **N** ripete l'ultima ricerca in senso inverso.

Le espressioni regolari usabili sono quelle elementari e quindi facenti capo alle seguenti regole:

. carattere qualsiasi;

\ fa perdere di significato al carattere seguente (lo cerca letteralmente cosí come scritto);

^ inizio riga;

$ fine riga;

[abc] qualsiasi carattere all'interno dell'insieme specificato;

[^abc] qualsiasi carattere che non appartengo all'insieme specificato;

[a-z] intervallo di caratteri

Un altro utile comando é **%** che permette la ricerca della parentesi tonda/quadra/graffa corrispondente a quella attiva (sotto il cursore).

8 Operazioni sui file

Brevemente nella tabella seguente, i comandi piú comuni per la manipolazione dei file.

comando	**azione**
:f	visualizza nome e caratteristiche del file corrente
:w	salva il file corrente
:w *file*	salva una copia del file corrente in *file*
:*n1*,*n2*w *file*	scrive le righe da *n1* a *n2* in *file*
:*n1*,*n2*w >> *file*	aggiunge le righe da *n1* a *n2* a *file*
:w!	scrive ignorando la protezione
:q	esce dal file
:q!	esce dal file scartando le modifiche
:ZZ	esce e scrive il file solo se sono state apportate modifiche

E' possibile avviare l'editing di piú file da una singola istanza di *vi* e gestirli tramite i seguenti comandi:

comando	**azione**
:e *file*	carica *file* per l'editing
:e!	ricarica il file corrente annullando possibili modifiche non salvate
:n	avvia l'editing del prossimo file
:args	elenca i file aperti
%	nome file corrente
#	nome di file alternativo

Una funzionalitá molto utile é la possibilitá di sfruttare le potenzionalitá e i risultati della shell:

comando	**azione**
:r *file*	inserisce il contenuto di *file* dopo la posizione attiva
:r !*comando*	inserisce l'output di *comando* dopo la riga corrente
:!*comando*	esegue *comando* e poi restituisce il controllo
!!*comando*	sostituisce la riga corrente con l'output del comando specificato
CTRL+7	sospende l'editor; per ripristinarlo premere `fg`