usr_22.txt Per **Vim version 8.2.** Ultima modifica: 2019 Mar 28

VIM USER MANUAL - di Bram Moolenaar
Traduzione di questo capitolo: Stefano Palmeri

Trovare il file da aprire

I file possono venir trovati ovunque. Così, come fare a trovarli? Vim offre vari modi per esplorare l'albero delle directory. Ci sono comandi per saltare ad un file che è menzionato in un altro. E Vim ricorda quali file siano stati modificati in precedenza.

Capitolo seguente: |**usr_23.txt**| Modifica di altri file
Capitolo precedente: |**usr_21.txt**| Andarsene e ritornare
Indice: |**usr_toc.txt**|

==========

22.1 Il file browser

Vim ha un plugin che rende possibile visualizzare una directory. Provate questo: >

```
:edit .
```

Tramite la magia dei comandi automatici e degli script di Vim, la finestra verrà riempita con i contenuti della directory. Apparirà come questa:

```
" ============================================================================ ~
" Netrw Directory Listing                                        (netrw v109) ~
"   Sorted by      name ~
"   Sort sequence: [\/]$,\.h$,\.c$,\.cpp$,*,\.info$,\.swp$,\.o$\.obj$,\.bak$ ~
"   Quick Help: <F1>:help  -:go up dir  D:delete  R:rename  s:sort-by  x:exec ~
" ============================================================================ ~
../ ~
./ ~
check/ ~
Makefile ~
autocmd.txt ~
change.txt ~
eval.txt~ ~
filetype.txt~ ~
help.txt.info ~
```

Potrete vedere queste voci:

1. Il nome dello strumento di browsing tool e il suo numero di versione
2. Il nome della directory in browsing
3. Il tipo di ordinamento della lista (può essere per nome, data, o dimensione, rispettivamente name, time, size)
4. Come vanno ordinati i nomi (prima le directory poi i file *.h, i file *.c, etc.)
5. Come ottenere aiuto (usare il tasto <F1>), e una lista abbreviata dei comandi disponibili
6. Una lista di file, compreso il file "../", che permette di passare alla directory superiore (parent directory).

Se l'evidenziazione della sintassi è abilitata, le diverse parti sono messe in evidenza in modo da poterle individuare più facilmente.

Potete usare i comandi di Vim in modo Normal per muovervi nel testo. Ad esempio, spostate il cursore su un nome file e premete <Invio>; vi troverete in modifica su quel file. Per tornare di nuovo nel browser, usate nuovamente ":edit .", oppure usate ":Explore". Si può usare anche CTRL-O.
Provate a usare <Invio> mentre il cursore è sul nome di una directory. Il risultato è che il file browser si sposta in quella directory e mostra quel che si trova in essa. Premendo <Invio> sulla prima directory "../" vi sposta al livello superiore. Premere "-" fa la stessa cosa, senza bisogno di portare

prima il cursore su "../".

Potete premere <F1> per avere aiuto sulle cose che potete fare nel netrw
browser. Questo è ciò che otterrete: >

```
    9. Directory Browsing        netrw-browse   netrw-dir   netrw-list   netrw-help

    MAPS                                                                  netrw-maps
         <F1>.............Help.....................................|netrw-help|
         <cr>.............Browsing.................................|netrw-cr|
         <del>............Deleting Files or Directories............|netrw-delete|
         -................Going Up.................................|netrw--|
         a................Hiding Files or Directories..............|netrw-a|
         mb...............Bookmarking a Directory...................|netrw-mb|
         gb...............Changing to a Bookmarked Directory.........|netrw-gb|
         cd...............Make Browsing Directory The Current Dir....|netrw-c|
         d................Make A New Directory.....................|netrw-d|
         D................Deleting Files or Directories............|netrw-D|
         <c-h>............Edit File/Directory Hiding List...........|netrw-ctrl-h|
         i................Change Listing Style.....................|netrw-i|
         <c-l>............Refreshing the Listing....................|netrw-ctrl-l|
         o................Browsing with a Horizontal Split..........|netrw-o|
         p................Use Preview Window.......................|netrw-p|
         P................Edit in Previous Window...................|netrw-p|
         q................Listing Bookmarks and History.............|netrw-qb|
         r................Reversing Sorting Order...................|netrw-r|
<        (etc)
```

Il tasto <F1> vi porta poi a una pagina di indice dell'aiuto sui contenuti
del netrw browser.
È una normale pagina di aiuto; usate il solito |**CTRL-]**| per saltare a elementi
marcati (tag) e |**CTRL-O**| per tornare indietro.

Per scegliere file da vedere e modificare: (col cursore sopra un nome file)

<enter>	Apre il file nella finestra corrente.	|**netrw-cr**|
o	Divide in orizz. finestra e mostra file	|**netrw-o**|
v	Divide in vert. finestra e mostra file	|**netrw-v**|
p	Usa la |**preview-window**|	|**netrw-p**|
P	Manda in edit nella finestra precedente	|**netrw-P**|
t	Apre il file in una nuova linguetta	|**netrw-t**|

I seguenti comandi in modo Normal si possono usare per controllare quel che
il browser mostra:

i	Controlla stile lista (thin, long, wide, and tree). (smilzo, lungo, largo e ad albero). La lista lunga include le informazioni dimensione e data.
s	Premendo s ripetutamente alterna fra i modi in cui i file sono ordinati; si può ordinare per nome, data di modifica, o dimensione.
r	Inverte l'ordine della lista ["gli ultimi saranno i primi"].

Come esempio di comandi differenti da quelli in modo Normal:

cd	Cambia l'idea di directory corrente per Vim, questa diventa la stessa della directory del browser. (Vederer |**g:netrw_keepdir**| per controllare anche questo)
R	Rinomina il file o directory sotto il cursore; una domanda verrà fatta per specificare il nuovo nome.
D	Elimina il file o directory sotto il cursore; verrà fatta una domanda di conferma.
mb gb	Crea un segnalibro [bookmark] /vai a un segnalibro

Si può anche usare il modo Command; ancora, solo qualche esempio:

:Explore [directory]	Esplora la directory specificata/corrente
:NetrwSettings	Una lista esauriente delle impostazioni netrw correnti, con possibile accesso all'aiuto.

Il browser netrw non è solo limitato alla vostra macchina locale; si possono usare URL, come: (la "/" finale è importante)

```
:Explore ftp://somehost/path/to/dir/
:e scp://somehost/path/to/dir/
```

Si veda |**netrw-browse**| per ulteriori informazioni.

==========

22.2 La directory corrente

Proprio come la shell, Vim ha il concetto di directory corrente. Supponete che voi siate nella vostra home directory e vogliate aprire alcuni file nella directory "VeryLongFileName". Potete fare: >

```
:edit VeryLongFileName/file1.txt
:edit VeryLongFileName/file2.txt
:edit VeryLongFileName/file3.txt
```

Per evitare una digitazione eccessiva, fate questo: >

```
:cd VeryLongFileName
:edit file1.txt
:edit file2.txt
:edit file3.txt
```

Il comando ":cd" cambia la directory corrente. Potete vedere quale sia la directory corrente con il comando ":pwd" : >

```
:pwd
/home/Bram/VeryLongFileName
```

Vim ricorda l'ultima directory che avete usato. Usate "cd -" per ritornarvi. Esempio: >

```
:pwd
/home/Bram/VeryLongFileName
:cd /etc
:pwd
/etc
:cd -
:pwd
/home/Bram/VeryLongFileName
:cd -
:pwd
/etc
```

LA FINESTRA DELLA DIRECTORY LOCALE

Quando dividete una finestra, entrambe le finestre useranno la stessa directory corrente. Quando volete modificare un certo numero di file da qualche altra parte nella nuova finestra, potete far sì che essa usi un'altra directory, senza cambiare la directory corrente nell'altra finestra. Questa si chiama directory locale. >

```
:pwd
/home/Bram/VeryLongFileName
:split
:lcd /etc
:pwd
/etc
CTRL-W w
:pwd
/home/Bram/VeryLongFileName
```

Se il comando `:lcd` non è ancora stato usato, tutte le finestre condivideranno la stessa directory corrente. Eseguire un comando `:cd` in una finestra cambierà la directory corrente anche nell'altra finestra.

Per una finestra dove sia stato usato il comando `:lcd` verrà ricordata una directory corrente differente. Usare `:cd` o `:lcd` in altre finestre non la cambierà.

Quando si usa il comando `:cd` in una finestra posta in una diversa directory corrente, si tornerà a usare la directory condivisa.

LOCAL DIRECTORY PER LINGUETTE

Se si apre una nuova linguetta, questa usa la directory della finestra della precedente linguetta, da cui è stata aperta la nuova linguetta. Si può modificare la directory della linguetta corrente usando il comando `:tcd`. Tutte le finestre in una linguetta condividono questa directory, tranne le finestre che hanno una directory locale alla finestra. Ogni nuova finestra aperta in questa linguetta userò questa directory come directory corrente di lavoro. Se si usa un comando `:cd` in una linguetta non cambierà la directory di lavoro delle linguette che hanno una directory di lavoro a livello di linguetta.
Quando si cambia la directory di lavoro globale usando il comando `:cd` in una linguetta, verrà cambiata anche la directory di lavoro corrente relativa alla linguetta corrente.

==========

22.3 Trovare un file

State scrivendo un programma in linguaggio C che contiene questa linea:

```
#include "inits.h" ~
```

Volete vedere cosa c'è in quel file "inits.h". Muovete il cursore sul nome del file e digitate: >

```
gf
```

Vim troverà il file e ne mostrerà il contenuto.
Cosa succede se il file non è nella directory corrente? Vim userà l'opzione 'path' per trovare il file. Questa opzione è un elenco di nomi di directory nelle quali cercare il vostro file.
Supponete che i vostri file include siano in "c:/prog/include". Questo comando la aggiungerà alla opzione 'path': >

```
:set path+=c:/prog/include
```

Questa directory si trova in un percorso assoluto. Non importa dove voi siate, sarà lo stesso posto. Cosa fare se avete collocato dei file in una subdirectory, al di sotto di dov'è il file? Potete specificare un percorso relativo. Questo inizia con un punto: >

```
:set path+=./proto
```

Questo dice a Vim di cercare nella directory "proto", sotto la directory dove si trova il file nel quale avete usato "gf". Così, usare "gf" su "inits.h" farà sì che Vim cerchi "proto/inits.h", iniziando nella directory del file.
Senza "./", quindi "proto", Vim dovrebbe cercare nella directory "proto" sotto la directory corrente. La directory corrente, però, potrebbe non essere quella dove il file che state modificando è collocato.

L'opzione 'path' permette di specificare le directory dove cercare i file in molti più modi. Leggete l'aiuto per l'opzione 'path'.
L'opzione 'isfnamè è usata per decidere quali caratteri sono inclusi nel nome del file e quali non lo sono (es., il carattere " nell'esempio in alto).

Quando conoscete il nome del file, ma non deve essere trovato nel file, potete digitare questo: >

```
:find inits.h
```

Vim userà quindi l'opzione 'path' per trovare il file. Questa è la stessa cosa del comando ":edit", eccetto che per l'uso di 'path'.

Per aprire il file trovato in una nuova finestra usate CTRL-W f anziché "gf", oppure usate ":sfind" al posto di ":find".

Un bel modo per avviare direttamente Vim per aprire un file che sia ovunque

nel 'path' è: >

```
vim "+find stdio.h"
```

Ciò trova "stdio.h" nel vostro valore di 'path'. I doppi apici sono necessari per avere un solo argomento |**-+c**|.

==========

22.4 La lista dei buffer

L'editor Vim usa il buffer del terminale per descrivere un file che si sta aprendo. In realtà, il buffer è una copia del file sul quale voi state lavorando. Quando avrete finito di modificare il buffer, voi scriverete i contenuti del buffer nel file. I buffer non contengono solo i contenuti del file, ma anche tutti i segnalibri, le impostazioni e le altre cose che lo accompagnano.

NASCONDERE I BUFFER

Supponete che voi stiate lavorando sul file one.txt e abbiate bisogno di passare al file two.txt.
Potreste usare semplicemente ":edit two.txt", ma poiché avete fatto delle modifiche in one.txt, quel comando non funzionerà. Inoltre non volete ancora salvare one.txt. Vim ha una soluzione per voi: >

```
:hide edit two.txt
```

Il buffer "one.txt" scompare dallo schermo, ma Vim sa ancora che voi state lavorando su questo buffer, così conserva il testo modificato. Questo viene chiamato buffer nascosto: il buffer contiene il testo, ma voi non potete vederlo.
L'argomento del comando ":hide" è un altro comando. ":hide" fa sì che quel comando si comporti come se l'opzione 'hidden' sia stata impostata. Potete anche impostare questa opzione esplicitamente. L'effetto è che quando un buffer viene lasciato, esso diventa nascosto.
State attenti! Quando avete nascosto buffer con cambiamenti, non chiudete Vim senza essere sicuri di avere salvato tutti i buffer.

INATTIVARE I BUFFER

Quando un buffer è stato usato una volta, Vim ricorda alcune informazioni su di esso. Quando non appare in una finestra e non è nascosto, esso è ancora nella lista dei buffer. Questo viene chiamato buffer inattivo. Panoramica:

Attivo	Appare in una finestra, testo caricato.
Nascosto	Non è in una finestra, testo caricato.
Inattivo	Non è in una finestra, testo non caricato.

I buffer inattivi vengono memorizzati, poiché Vim conserva le informazioni che li riguardano, come i segnalibri. Ed anche il ricordare il nome del file è utile, affinché voi possiate vedere su quali file avete lavorato. Ed aprirli ancora.

ELENCARE I BUFFERS

Esaminate la lista dei buffer con questo comando: >

```
:buffers
```

Un comando che fa la stessa cosa, non così ovvio per elencare i buffer, ma molto più corto da digitare, è: >

```
:ls
```

L'output potrebbe apparire come questo:

```
  1 #h   "help.txt"                     line 62 ~
  2 %a + "usr_21.txt"                   line 1 ~
  3      "usr_toc.txt"                  line 1 ~
```

La prima colonna contiene il numero del buffer. Potete usare questo per ritornare ad un buffer senza doverne digitare il nome, guardate sotto.
   Dopo il numero del buffer vengono i flag. Quindi segue il nome del file ed il numero della linea dove era il cursore l'ultima volta.
   I flag che possono apparire sono questi (da sinistra a destra):

```
      u       Il buffer è unlisted (non in lista) |unlisted-buffer|.
       %      Buffer corrente.
       #      Buffer alternativo.
        a     Il buffer è caricato e mostrato.
        h     Il buffer è caricato ma nascosto.
         =    Il buffer è di sola lettura (read-only).
         -    Il buffer non è modificabile, l'opzione 'modifiable' è
              inattiva.
          +   Il buffer è stato modificato.
```

RITORNARE AD UN BUFFER

Potete ritornare ad un buffer tramite il suo numero. Questo evita di dover scrivere il nome del file : >

```
:buffer 2
```

Ma il solo modo di sapere il numero è guardare nell'elenco dei buffer. Potete, invece, usare il nome, o parte di esso: >

```
:buffer help
```

Vim troverà la migliore corrispondenza per il nome che avete digitato. Se c'è un solo buffer che corrisponde al nome, esso verrà usato. In questo caso "help.txt".
   Per aprire un buffer in una nuova finestra: >

```
:sbuffer 3
```

Funziona anche con il nome ugualmente bene.

USARE LA LISTA DEI BUFFER

Potete muovervi nella lista dei buffer con questi comandi:

```
:bnext          va al buffer successivo
:bprevious      va al buffer precedente
:bfirst         va al primo buffer
:blast          va all'ultimo buffer
```

Per rimuovere un buffer dalla lista, usate questo comando: >

```
:bdelete 3
```

Ancora, ciò funziona anche con un nome.
   Se cancellate un buffer che era attivo (visibile in una finestra), quella finestra verrà chiusa. Se cancellate il buffer corrente, la finestra corrente verrà chiusa. Se fosse l'ultima finestra, Vim cercherebbe un altro buffer da modificare. Voi non potete non avere nulla di aperto!

```
Nota:
Anche dopo aver rimosso il buffer con ":bdelete", Vim lo ricorderà.
In realtà esso è reso "unlisted" e non compare più nella lista di
":buffers". Il comando ":buffers!" elencherà i buffer unlisted
(sì, Vim può fare l'impossibile). Perché Vim dimentichi veramente
un buffer, usate ":bwipe". Vedete anche l'opzione 'buflisted'.
```

==

Capitolo seguente: |**usr_23.txt**| Modifica di altri file

Per segnalazioni scrivere a vimdoc.it at gmail dot com
     oppure ad Antonio Colombo azc100 at gmail dot com