usr_08.txt Per **Vim version 8.2.** Ultima modifica: 2017 Aug 11

VIM USER MANUAL - di Bram Moolenaar
Traduzione di questo capitolo: Valentino Squilloni

Dividere le finestre

Poter osservare due diversi file, uno sopra l'altro. Oppure vedere contemporaneamente due diversi punti dello stesso file. Confrontare due file diversi mettendoli l'uno affianco all'altro. Tutto ciò è possibile dividendo in più parti la finestra.

Capitolo seguente: |**usr_09.txt**| Usare la GUI
Capitolo precedente: |**usr_07.txt**| Elaborare più di un file
Indici: |**usr_toc.txt**|

==========

08.1 Dividere una finestra

Il modo più semplice per aprire una nuova finestra è quello di usare il seguente comando: >

```
:split
```

Questo comando divide lo schermo in due finestre e posiziona il cursore nella finestra più in alto:

```
+----------------------------------+
|/* file one.c */                  |
|~                                 |
|~                                 |
|one.c=============================|
|/* file one.c */                  |
|~                                 |
|one.c=============================|
|                                  |
+----------------------------------+
```

Quello che vedrete qui sono due finestre aperte sullo stesso file. La linea con "====" è la linea di status. Essa ci dà informazioni sulla finestra sovrastante. (In pratica la linea di status si presenterà con colori opposti a quelli dello schermo.)
Le due finestre permettono di vedere due parti dello stesso file. Ad esempio, sarà possibile far mostrare alla finestra in alto la dichiarazione delle variabili di un programma, ed a quella in basso il codice che utilizza quelle variabili.

Il comando CTRL-W w viene usato per saltare fra le finestre. Se siete nella finestra in alto, CTRL-W w salterà alla finestra sottostante. Se siete nell'ultima finestra (quella più in basso) CTRL-W w salterà alla prima. (CTRL-W CTRL-W fa la stessa cosa, nel caso il tasto CTRL venga rilasciato un pelo troppo tardi.)

CHIUDERE LA FINESTRA

Per chiudere una finestra, si usa il comando: >

```
:close
```

In realtà, ogni comando che chiude l'editing di un file funzionerà, come ":quit" e "ZZ". Ma ":close" previene l' uscita accidentale da Vim quando si

chiude l'ultima finestra.

CHIUDERE TUTTE LE ALTRE FINESTRE

Se avete aperto un mucchio di finestre, ma ora volete concentrarvi solo su una di esse, questo comando vi sarà utile: >

```
:only
```

Questo chiuderà tutte le finestre, a parte quella corrente (quella dove è presente il cursore). Se qualcuna delle altre finestre ha avuto dei cambiamenti, avrete un errore e quelle finestre non verranno chiuse.

==========

08.2 Dividere una finestra aprendo un altro file

Il comando seguente apre una seconda finestra e inizia ad editare in essa il file dato: >

```
:split two.c
```

Se state editando one.c, allora il risultato del comando sarà circa così:

```
+----------------------------------+
|/* file two.c */                  |
|~                                 |
|~                                 |
|two.c=============================|
|/* file one.c */                  |
|~                                 |
|one.c=============================|
|                                  |
+----------------------------------+
```

Per aprire una finestra su un nuovo file vuoto, usate questo: >

```
:new
```

Potete ripetere i comandi ":split" e ":new" per creare quante finestre volete.

==========

08.3 Dimensioni della finestra

Il comando ":split" può avere un argomento numerico. Se viene specificato, questo sarà l'altezza della nuova finestra. Per esempio, il comando seguente apre una nuova finestra alta tre linee e inizia a editarvi il file alpha.c: >

```
:3split alpha.c
```

Per le finestre già esistenti potete cambiarne le dimensioni in diversi modi. Se avete un mouse funzionante, è facile: basta muovere il cursore sulla linea di status che separa le due finestre, e trascinarla in alto o in basso.

Per aumentare la dimensione di una finestra: >

```
CTRL-W +
```

Per diminuirla: >

```
CTRL-W -
```

Entrambi i comandi prendono un numero e aumentano o diminuiscono l'altezza della colonna di quel numero di linee. In questo modo "4 CTRL-W +" da sì che la finestra diventi 4 linee più alta.

Per impostare l'altezza della finestra ad un numero specificato di linee: >

```
{altezza}CTRL-W _
```

Cioè: un numero {altezza}, CTRL-W e un underscore (il tasto - con Shift premuto sulle tastiere English-US).

Per rendere una finestra più alta possibile, si usa il comando CTRL-W _ senza un numero prima.

USARE IL MOUSE

In Vim potete fare un sacco di cose molto velocemente direttamente con la tastiera. Sfortunatamente, i comandi per ridimensionare la finestra richiedono numerose pressioni dei tasti. In questo caso, usare il mouse è più veloce. Posizionate il mouse sulla linea di status. Ora premete il tasto sinistro del mouse e trascinate. La linea di status si muoverà, facendo diventare la finestra da una parte più alta e quella dall'altra più bassa.

OPZIONI

L'opzione 'winheight' è usata per impostare l'altezza minima desiderata di una finestra e 'winminheight' per l'altezza minima consentita di una finestra.
Allo stesso modo c'è 'winwidth' per la larghezza minima desiderata e 'winminwidth' per la minima larghezza consentita.
L'opzione 'equalalways', quando è impostata, fa sì che Vim aggiusti le altezze delle finestre quando viene chiusa o aperta una finestra per uniformarle.

==========

08.4 Tagli verticali

Il comando ":split" crea la finestra nuova sopra a quella corrente. Invece per far comparire la nuova finestra alla sinistra di quella già presente, si usa: >

```
:vsplit
```

oppure: >

```
:vsplit two.c
```

Il risultato sarà simile a questo:

```
+----------------------------------------+
|/* file two.c */    |/* file one.c */   |
|~                   |~                  |
|~                   |~                  |
|~                   |~                  |
|two.c===============one.c=============|
|                                        |
+----------------------------------------+
```

In pratica, la colonna di | al centro dello schermo sarà in colori opposti a quelli dello schermo. Viene chiamata separatore verticale. Serve a separare le due finestre alla destra e alla sinistra di essa.

C'è anche il comando ":vnew", che taglia verticalmente la finestra aprendo un nuovo file vuoto. Un altro modo di ottenere ciò: >

```
:vertical new
```

Il comando ":vertical" può esser inserito prima di un altro comando che divide una finestra. Questo farà sì che quel comando divida la finestra in verticale invece che in orizzontale. (Se il comando non divide finestre, non verrà influenzato da esso.)

MUOVERSI FRA LE FINESTRE

Poiché si può dividere più volte la finestra in orizzontale e in verticale, è possibile creare quasi ogni tipo di schema. Potete poi usare questi comandi per muovervi da una finestra all'altra:

```
CTRL-W h        muove verso la finestra a sinistra
CTRL-W j        muove verso la finestra di sotto
CTRL-W k        muove verso la finestra di sopra
```

```
        CTRL-W l        muove verso la finestra a destra

        CTRL-W t        muove verso la finestra più in alto
        CTRL-W b        muove verso la finestra più in basso
```

Noterete che le stesse lettere sono usate per muovere il cursore. E anche le frecce possono essere usate, se preferite.
Ulteriori comandi per muoversi verso altre finestre: |**Q_wi**|..

==========

08.5 Muovere le finestre

Avete tagliato alcune finestre, ma ora sono nel posto sbagliato. Quindi avrete bisogno di un comando per muovere la finestra in un altro posto. Ad esempio, avete tre finestre messe in questo modo:

```
+----------------------------------+
|/* file two.c */                  |
|~                                 |
|~                                 |
|two.c=============================|
|/* file three.c */                |
|~                                 |
|~                                 |
|three.c===========================|
|/* file one.c */                  |
|~                                 |
|one.c=============================|
|                                  |
+----------------------------------+
```

Chiaramente l'ultima in basso dovrebbe stare nel posto più in alto. Allora muovetevi verso quella finestra (usando CTRL-W w) e usate questo comando: >

```
        CTRL-W K
```

Questo comando usa la lettera maiuscola K. Quello che succede è che la finestra in questione viene spostata nel punto più alto. Noterete che K è usata ancora per muoversi verso l'alto.
Se ci sono dei tagli verticali, CTRL-W K muoverà la finestra corrente nel punto più in alto e le farà occupare l'intera larghezza della finestra di Vim. Se questo è il vostro layout:

```
+-----------------------------------------+
|/* two.c */  |/* three.c */  |/* one.c */  |
|~             |~              |~           |
|~             |~              |~           |
|~             |~              |~           |
|~             |~              |~           |
|~             |~              |~           |
|two.c=========three.c=========one.c========|
|                                         |
+-----------------------------------------+
```

Allora usando CTRL-W K sulla finestra al centro (three.c) avrà come risultato:

```
+-----------------------------------------+
|/* three.c */                           |
|~                                        |
|~                                        |
|three.c==================================|
|/* two.c */           |/* one.c */       |
|~                     |~                 |
|two.c=================one.c==============|
|                                         |
+-----------------------------------------+
```

Gli altri tre comandi simili (e probabilmente ora già ve li immaginate) sono:

```
        CTRL-W H        muove la finestra nel punto più a sinistra
        CTRL-W J        muove la finestra nel punto più in basso
        CTRL-W L        muove la finestra nel punto più a destra
```

==========

08.6 Comandi per tutte le finestre

Quando avete molte finestre aperte e volete uscire da Vim, potete chiudere separatamente ogni finestra. Un modo più veloce per fare la stessa cosa è il comando: >

```
:qall
```

Che sta per "quit all". Se qualche finestra contiene delle modifiche, Vim non uscirà. Il cursore sarà automaticamente posizionato in una di queste finestre che hanno subito modifiche. Potete usare ":write" per salvare i cambiamenti fatti, oppure ":quit!" per buttarli via.

Se sapete che ci sono più finestre da salvare, e le volete salvare tutte, usate questo comando: >

```
:wall
```

Che sta per "write all". In realtà va a scrivere solo quei file che hanno subito delle modifiche. Vim sa che non ha senso scrivere un file che non è stato modificato.
   C'è anche la combinazione di ":qall" e ":wall": il comando "scrivi tutto ed esci": >

```
:wqall
```

Che salva tutti i file modificati e chiude Vim.
   Infine, c'è un comando per chiudere Vim e ignorare tutti i cambiamenti fatti: >

```
:qall!
```

Attenzione, non c'è il modo di annullare questo comando!

APRIRE UNA FINESTRA PER OGNI ARGOMENTO

Per far sì che Vim apra una finestra per ogni file, lo si fa partire con l'argomento "-o": >

```
vim -o one.txt two.txt three.txt
```

Il risultato sarà:

```
+-------------------------------+
|file one.txt                   |
|~                              |
|one.txt========================|
|file two.txt                   |
|~                              |
|two.txt========================|
|file three.txt                 |
|~                              |
|three.txt======================|
|                               |
+-------------------------------+
```

L'argomento "-O" è usato per avere finestre separate verticalmente.
   Quando Vim è già in esecuzione, il comando ":all" apre una finestra per ogni file nella lista degli argomenti. ":vertical all" fa la stessa cosa ma con tagli verticali.

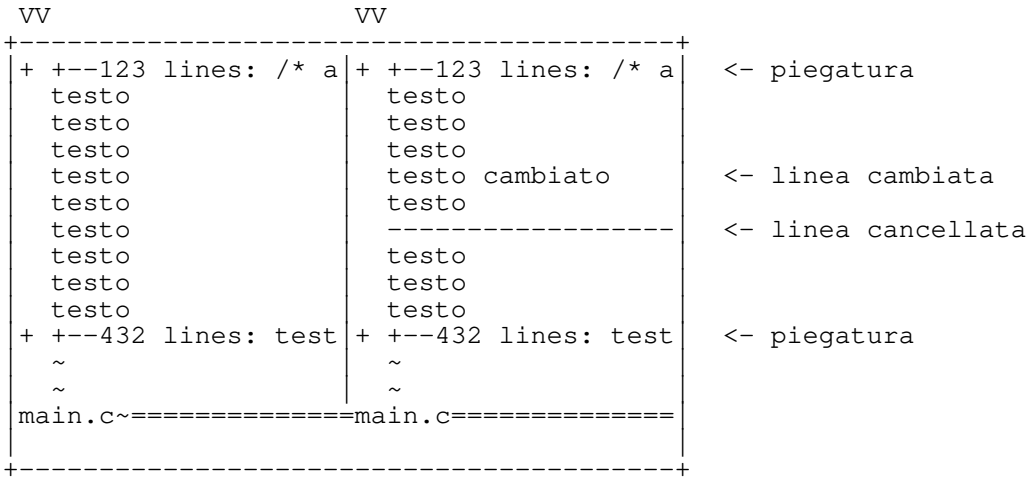
==========

08.7 Evidenziare le differenze con vimdiff

C'è un modo speciale di far partire Vim, che mostra le differenze fra due file. Prendiamo ad esempio il file "main.c" e inseriamo qualche carattere in una linea. Salviamo il file con l'opzione 'backup' attiva, così che il file di backup "main.c~" conterrà la versione precedente del file.
   Scrivete questo comando in una shell (non in Vim): >

```
vimdiff main.c~ main.c
```

Vim partirà con due finestre una al fianco dell'altra. Voi vedrete solo la linea in cui avete aggiunto i caratteri, e qualche linea sopra e sotto di essa.



(Questa figura non mostra le parti evidenziate, usate direttamente vimdiff per una resa migliore.)

Le linee che non erano state modificate sono collassate in una riga. Questo è chiamato una piega chiusa (closed fold). Nella figura sono indicate con "<- piegatura". Sebbene la prima linea piegata rappresenti 123 linee di testo, queste linee sono uguali in entrambi i file.

La linea contrassegnata con "<- linea cambiata" è evidenziate, e il testo inserito viene visualizzato in un altro colore. Questo mostra in modo chiaro le differenze fra i due file.

La linea che è stata cancellata è mostrata con "---" nella finestra di main.c. Notate il contrassegno "<- linea cancellata" nella figura. Questi caratteri non sono veramente lì. Servono solo a riempire main.c, in modo che mostri lo stesso numero di linee dell'altra finestra.

LA COLONNA PIEGA

Ogni finestra ha una colonna sulla sinistra con lo sfondo leggermente differente. Nella figura sopra queste colonne sono indicate con "VV". Noterete un carattere "+" all'inizio di ogni piega chiusa. Muovete il mouse su quel "+" e fate clic col pulsante sinistro. La piega si distenderà, e potrete vedere il testo che contiene.

La colonna piega contiene un segno di "-" per ogni piegatura aperta. Se fate clic su quel simbolo meno, la piega si chiuderà.

Ovviamente questo sarà possibile solo se avete un mouse funzionante. Altrimenti, potete comunque usare "zo" per aprire una piegatura, e "zc" per chiuderla.

EVIDENZIARE LE DIFFERENZE IN VIM

Il modo Diff può essere attivato in un'altra maniera, direttamente da dentro a Vim. Editate il file "main.c", poi dividete la finestra e mostrate le differenze: >

```
:edit main.c
:vertical diffsplit main.c~
```

Il comando ":vertical" viene usato per far sì che la finestra venga tagliata verticalmente. Se lo omettete, avrete un taglio orizzontale.

Se avete un file patch o diff, potete usare un terzo modo per far partire il modo Diff. Innanzitutto editate il file a cui le patch andranno applicate. Poi dite a Vim il nome del file di patch: >

```
:edit main.c
:vertical diffpatch main.c.diff
```

ATTENZIONE: Il file patch deve contenere solo una patch per il file che state

editando. Altrimenti avrete molti messaggi di errore, e alcuni file potrebbero essere aggiornati dalla patch in modo inaspettato.
La patch verrà applicata solo al file all'interno di Vim. Il file sul vostro hard disk resterà immutato (fino a che deciderete di salvare il file).

TENERE UNITO LO SCORRIMENTO

Quando i file hanno più di una variazione, potete scrollare nel solito modo. Vim cercherà di tenere allineato l'inizio di entrambe le finestre, per mostrare semplicemente le differenze, lato a lato.
Se volete disabilitare temporaneamente questa funzionalità, usate questo comando: >

```
:set noscrollbind
```

SALTARE ALLE DIFFERENZE

Se in qualche maniera avete disabilitato la funzione di piegatura, potrebbe essere difficile trovare le differenze. Usate questo comando per saltare in avanti alla prossima differenza: >

```
]c
```

Per andare nell'altra direzione (verso l'alto) usare : >

```
[c
```

Mettere come prefisso un numero per saltare più volte.

ELIMINARE LE DIFFERENZE

Potete spostare delle parti di testo da una finestra all'altra. Questo fa sì che siano rimosse le differenze, oppure che siano aggiunte le parti nuove. Vim non tiene aggiornate le differenze in ogni occasione. Per aggiornarle, usate questo comando: >

```
:diffupdate
```

Per eliminare una differenza, potete spostare il testo all'interno di un blocco evidenziato da una finestra all'altra. Prendete ad esempio i file "main.c" e "main.c~" citati precedentemente. Spostate il cursore sulla finestra di sinistra, sopra alla linea che nell'altra finestra era stata cancellata. Ora usate questo comando: >

```
dp
```

Le differenze verranno rimosse mettendo il testo della finestra corrente nell'altra finestra. "dp" sta per "diff put" (inserisci la differenza).
Potete anche fare la stessa cosa in un'altra maniera. Spostate il cursore nella finestra di destra, sulla linea dove era stati inseriti dei "cambiamenti". Ora usate questo comando: >

```
do
```

Le differenze in questo caso saranno rimosse prendendo la parte di testo dall'altra finestra. Poiché a questo punto non ci saranno più differenze, Vim racchiuderà tutto il testo in una piega chiusa. "do" sta per "diff obtain" (prendi le differenze). "dg" sarebbe stato meglio, ma aveva già un significato diverso ("dgg" cancella tutto dal cursore fino alla prima linea compresa).

Per maggiori dettagli sul modo Diff, vedere |**vimdiff**|.

==========

08.8 Varie ed eventuali

L'opzione 'laststatus' può essere usata per specificare se e quando l'ultima finestra debba avere la linea di status:

```
0       mai
```

```
        1       solo quando ci sono finestre divise (default)
        2       sempre
```

Molti comandi che editano un altro file hanno una variante che divide la finestra. Per i comandi nella linea dei comandi questo viene fatto preponendo una "s". Ad esempio: ":tag" salta ad un certo punto contrassegnato, ":stag" divide in due la finestra e salta a quel punto.
   Per i comandi in modo Normal viene preposta la combinazione CTRL-W. CTRL-^ salta al file alternato, CTRL-W CTRL-^ divide in due la finestra ed edita il file alternato.

L'opzione 'splitbelow' può essere impostata per far sì che la finestra nuova appaia sotto alla finestra corrente. L'opzione 'splitright' analogamente fa sì che appaia alla destra della finestra corrente.

Quando si divide in due la finestra si può premettere al comando un modificatore per decidere dove apparirà la finestra:

```
        :leftabove {comando}  a sinistra o sopra la finestra corrente
        :aboveleft {comando}  idem
        :rightbelow {comando} a destra o sotto la finestra corrente
        :belowright {comando} idem
        :topleft {comando}    nel punto più in alto o a sinistra della
                              finestra di Vim
        :botright {comando}   nel punto più in alto o alla destra della
                              finestra di Vim
```

==========

08.9 Linguette

Avrete notato che le finestre non si sovrappongono mai. Questo implica che potete velocemente esaurire lo spazio sullo schermo. La soluzione a questo si chiama Linguette.

Supponete di stare modificando "file_uno". Per creare una nuova Linguetta, usate questo comando: >

```
        :tabedit file_due
```

Questo comincerà a modificare "file_due" in una finestra grande come l'intera finestra di Vim. Noterete una linguetta in alto coi nomi dei due file:

```
        +----------------------------------+
        | file_uno | /file_due/ _________X|     (file_due è in grassetto)
        |/* file_due */                    |
        |due                               |
        |due                               |
        |~                                 |
        |~                                 |
        |~                                 |
        |                                  |
        +----------------------------------+
```

Avete ora due Linguette. La prima contiene una finestra per "file_uno" e la seconda una finestra per "file_due". Sono come due pagine una sopra l'altra, con una linguetta che esce da ogni pagina, e mostra il nome del file.

Ora, fate clic col mouse su "file_uno" nella linea in alto. Otterrete:

```
        +----------------------------------+
        | /file_uno/ | file_due _________X|     (file_uno è in grassetto)
        |/* file_uno */                    |
        |uno                               |
        |uno                               |
        |~                                 |
        |~                                 |
        |~                                 |
        |                                  |
        +----------------------------------+
```

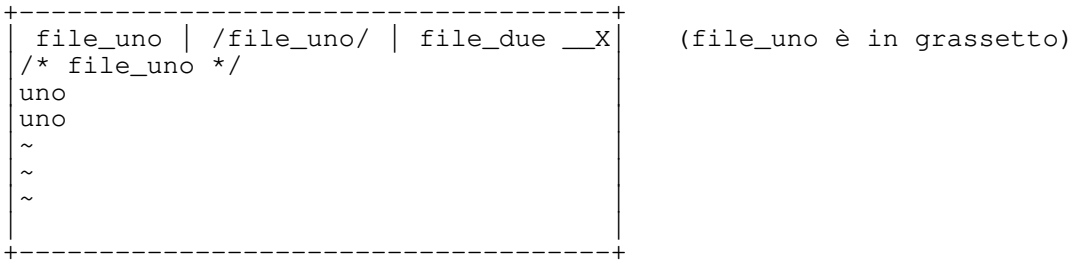
Quindi potete passare da una Linguetta all'altra facendo clic sull'etichetta corrispondente nella linea in alto. Se non avete un mouse o non volete

usarlo, potete usare il comando "gt". Per ricordarlo: Goto Tab.

Ora creiamo un'altra Linguetta col comando: >

```
:tab split
```

Questo crea una nuova Linguetta con una finestra che sta modificando lo stesso buffer della finestra in cui ci trovavamo:



(file_uno è in grassetto)

Potete mettere ":tab" prima di ogni comando Ex che apre una finestra. La finestra verrà aperta in una nuova linguetta. Altro esempio: >

```
:tab help gt
```

Mostrerà il testo di help per "gt" in una nuova linguetta.

Ci sono alcune altre cose possibili con le Linguette:

- fare clic col mouse nello spazio dopo l'ultima etichetta
  La successiva linguetta sarà scelta, come con "gt".

- fare clic col mouse sulla "X" nell'angolo in alto a destra
  La Linguetta corrente sarà chiusa. Tranne che ci siano modifiche non salvate nella Linguetta corrente.

- fare clic due volte col mouse sulla linea in alto
  Una nuova linguetta verrà creata.

- il comando "tabonly"
  Chiude tutte le Linguette, meno quella corrente. Tranne che ci siano modifiche non salvate nella Linguetta corrente.

Per maggiori informazioni sulle linguette si veda |**tab-page**|.

==========

Capitolo seguente: |**usr_09.txt**| Usare la GUI

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