usr_31.txt Per Vim version 7.1. Ultima modifica: 2007 Mag 08

VIM USER MANUAL - di Bram Moolenaar
Traduzione di questo capitolo: Roberto Franceschini

Sfruttare la GUI

Vim funziona bene in un terminale, ma la sua GUI ha qualche componente in più. Un navigatore di file può venir impiegato per i comandi che usano un file. Finestre di dialogo per scegliere tra più alternative. Tasti di scelta rapida per accedere a voci di menù rapidamente.

Capitolo seguente: |usr_32.txt| La storia degli undo
Capitolo precedente: |usr_30.txt| Editare programmi
Indice: |usr_toc.txt|

==========

31.1 Il Navigatore

Quando utilizzate la voce di menù File/Open... ottenete un navigatore di file. Questo rende semplice trovare il file che volete modificare. Ma cosa fare se voleste dividere la finestra per aprire un altro file? Non c'è una voce di menù per questo. Potreste usare prima Window/Split e poi File/Open..., ma è lavoro in più.

Poiché digitate la maggior parte dei comandi in Vim, è possibile anche aprire il navigatore digitando un comando. Per fare in modo che il comando split utilizzi il navigatore, anteponete "browse": >

```
:browse split
```

Selezionate un file ed il comando ":split" verrà eseguito su di esso. Se chiudete la finestra di dialogo non succederà niente, la finestra non verrà divisa.

Potete anche specificare un argomento per il nome del file. Questo verrà utilizzato per indicare al navigatore dove iniziare la ricerca. Ad esempio: >

```
:browse split /etc
```

Il navigatore si aprirà iniziando la ricerca nella directory "/etc".

Potete anteporre il comando ":browse" a qualunque comando che apra un file.

Se non viene specificata alcuna directory, Vim deciderà dove iniziare la ricerca. Per default utilizza l'ultima directory utilizzata la sessione precedente. Quindi se voi utilizzate ":browse split" e selezionate un file in "/usr/local/share", la prossima volta che utilizzerete ":browse" la ricerca inizierà nuovamente in "/usr/local/share".

Ciò può essere modificato con l'opzione **'browsedir'**. Può assumere uno di questi tre valori:

last	Utilizza l'ultima directory selezionata (default)
buffer	Utilizza la stessa directory del buffer corrente
current	Utilizza la directory corrente

Ad esempio, se voi siete nella directory "/usr", e state editando il file "/usr/local/share/readme", allora il comando: >

```
:set browsedir=buffer
:browse edit
```

Avvierà il navigatore in "/usr/local/share". In alternativa: >

```
:set browsedir=current
:browse edit
```

Avvierà il navigatore in "/usr".

> Note:
> Per evitare l'utilizzo del mouse, molti navigatori permettono di muoversi usando combinazioni di tasti. Dato che queste sono differenti per ogni sistema, non verranno spiegate qui. Vim utilizza un navigatore standard quando possibile, e la documentazione del vostro sistema dovrebbe contenere una spiegazione delle scorciatoie da tastiera.

Se non utilizzate l'interfaccia grafica, potete utilizzare la finestra del "file explorer" per selezionare i file come in un navigatore. Questo comunque non funziona per il comando ":browse". Vedere |**netrw-browser**|.

==========

31.2 Conferme

Vim vi protegge dal sovrascrivere accidentalmente un file o da altri modi di perdere delle modifiche. Se fate qualcosa che potrebbe essere sbagliato, Vim produce un messaggio di errore e vi suggerisce di aggiungere ! se veramente volete proseguire.

Per evitare di digitare nuovamente il comando con !, potete fare in modo che Vim vi presenti una finestra di dialogo. Potete quindi scegliere "OK" o "Cancel" per dire a Vim cosa volete.

Ad esempio, voi state aprendo un file e vi apportate delle modifiche. Iniziate la modifica di un altro file con: >

```
:confirm edit foo.txt
```

Vim farà apparire una finestra di dialogo che appare simile a questa:

```
+----------------------------------------+
|                                        |
|   ?   Salvare modifiche a "bar.txt"?   |
|                                        |
|   YES   NO             CANCEL          |
+----------------------------------------+
```

Adesso fate la vostra scelta. Se volete salvare le modifiche, selezionate "YES". Se volete abbandonare le modifiche per sempre: "NO". Se avete dimenticato cosa stavate facendo e volete verificare le modifiche effettuate usate "CANCEL". Tornerete allo stesso file, con le modifiche ancora lì.

Come ":browse", il comando ":confirm" può essere anteposto alla maggior parte dei comandi che aprano un altro file. E possono essere anche combinati: >

```
:confirm browse edit
```

Se il buffer corrente fosse stato modificato, ciò genererebbe una finestra di dialogo. Quindi apparirebbe un navigatore per scegliere il file da aprire.

> Note:
> Nelle finestre di dialogo potete usare la tastiera per selezionare la scelta. Solitamente il tasto **<Tab>** e le frecce cambiano la scelta. Premendo **<Enter>** selezionate la scelta. Questo, comunque, dipende dal sistema che utilizzate.

Il comando ":confirm" funziona anche quando non state usando la GUI. Invece di aprire una finestra di dialogo, Vim stamperà il messaggio in fondo alla finestra e vi chiederà di premere un tasto per effettuare la scelta. >

```
:confirm edit main.c
Salvare modifiche a "Untitled"? ~
[Y]es, (N)o, (C)ancel:  ~
```

Ora potete premere il tasto per la scelta. Non dovrete premere **<Enter>**, a differenza delle altre battute dalla riga di comando.

==========

31.3 Scelte rapide

La tastiera si usa per tutti i comandi di Vim. I menù forniscono un modo più semplice per selezionare i comandi, senza sapere come vengono chiamati. Ma dovete spostare la vostra mano dalla tastiera ed afferrare il mouse.

I menù possono spesso essere selezionati anche da tastiera. Dipende dal

vostro sistema, ma quasi sempre funziona così. Usate il tasto **<Alt>** assieme alla lettera sottolineata del menù. Per esempio **<A-w>** (**<Alt>** e w) aprirà il menù Finestra.
Nel menù Finestra , "split" ha la **<p>** sottolineata. Per selezionarlo premete **<Alt>** e quindi **<p>** assieme.

Dopo la prima selezione di un menù con il tasto **<Alt>**, potete usare le frecce per muovervi attraverso i menù. La freccia a destra seleziona un sottomenù e la freccia a sinistra lo chiude. Anche **<Esc>** chiude un menù. **<Enter>** seleziona una scelta.

Esiste un conflitto utilizzando il tasto **<Alt>** per scegliere le componenti dei menù ed impiegando le combinazioni del tasto **<Alt>** per le mappature. L'opzione **'winaltkeys'** dice a Vim cosa può fare con il tasto **<Alt>**.
Il valore di default "menu" è la scelta migliore: se la combinazione di tasti è una scelta rapida per il menù non può essere mappata. Tutti gli altri tasti sono disponibili per mappature.
Il valore "no" non consentirà l'uso di alcun tasto **<Alt>** per i menù. Così dovrete usare il mouse per i menù, e potranno essere mappate tutte le combinazioni con **<Alt>**.
Se impostata a "yes" significa che potrà utilizzare le combinazioni con **<Alt>** per i menù. Qualche combinazione del tasto **<Alt>** potrà eseguire anche azioni diverse dal selezionare i menù.

==========

31.4 Posizione e dimensione della finestra

Per leggere la posizione corrente della finestra di Vim usate: >

```
:winpos
```

Questo funzionerà solo nella GUI. L'output può essere simile a questo:

```
Posizione finestra: X 272, Y 103 ~
```

La posizione è data in pixel dello schermo. Ora potete usare i numeri per muovere la finestra di Vim. Per esempio, per muoverla cento pixel a sinistra: >

```
:winpos 172 103
```
<
Note:
Può esserci una leggera differenza tra la posizione riportata e dove la finestra si posiziona. Questo a causa del bordo della finestra. Quest'ultimo è aggiunto dal programma gestore delle finestre.

Potete utilizzare questo comando nello script di avvio per posizionare la finestra in una posizione specifica.

La dimensione della finestra di Vim è misurata in caratteri. Quindi dipende dalla dimensione del font in uso. Potete visualizzare la dimensione corrente con questo comando: >

```
:set lines columns
```

Per cambiare la dimensione impostate le opzioni **'lines'** e/o **'columns'** ad un nuovo valore: >

```
:set lines=50
:set columns=80
```

La visualizzazione della dimensione in un terminale funziona proprio come nella GUI. Nella maggior parte dei terminali non è però possibile impostare la dimensione.

Puoi avviare la versione X-Windows di gvim con un argomento per specificare la dimensione e la posizione della finestra: >

```
gvim -geometry {larghezza}x{altezza}+{x_offset}+{y_offset}
```

{larghezza} e **{altezza}** sono in caratteri, {x_offset} e {y_offset} sono in pixel. Esempio: >

```
gvim -geometry 80x25+100+300
```

==
31.5 Varie

Potete utilizzare gvim per scrivere un messaggio di posta elettronica. Nel vostro programma di posta elettronica dovete selezionare gvim come editor per i messaggi. Quando proverete, probabilmente non funzionerà: Il programma di posta pensa che la scrittura del messaggio sia finita, mentre gvim sta girando ancora!
Succede che gvim si scollega dalla shell nella quale è stato avviato. Ciò va bene quando avviate gvim in un terminale, per poter fare altro in quel terminale. Ma quando volete davvero aspettare che gvim abbia finito, dovete fare in modo che non si scolleghi. L'argomento "-f" lo fa: >

```
gvim -f file.txt
```

La "-f" sta per "foreground" (primo piano). Ora Vim bloccherà la shell da cui è stato avviato fino a che avrete finito di scrivere e siete usciti.

AVVIO RITARDATO DELLA GUI

In Unix è possibile avviare Vim in un terminale. Questo serve se volete avviare diversi programmi nella stessa shell. Se voi state lavorando su di un file e, ad un certo punto, volete passare alla GUI, potete avviarla con: >

```
:gui
```

Vim aprirà la finestra della GUI e non userà più il terminale. Potete quindi utilizzare il terminale per altri scopi. L'argomento "-f" viene qui usato per avviare la GUI in primo piano. Potete anche usare ":gui -f".

IL FILE DI AVVIO DI GVIM

Quando gvim viene avviato, legge il file gvimrc. Questo è simile al file vimrc usato quando lanciate Vim. Il file gvimrc può essere utilizzato per impostazioni e comandi che vengono usati solo quando la GUI deve ancora venire avviata. Per esempio, potete impostare l'opzione 'lines' per avere una diversa dimensione della finestra: >

```
:set lines=55
```

Non dovete fare ciò entro un terminale, in quanto la sua dimensione è fissa (ad eccezione di xterm che supporta il ridimensionamento).
Il file gvimrc viene cercato nella stessa posizione di vimrc. Normalmente il suo nome è "~/.gvimrc" in Unix e "$VIM/_gvimrc" in MS-Windows. La variabile d'ambiente $MYGVIMRC è impostata col suo nome, quindi potete usare questo comando per modificare il file, se ne avete uno: >

```
:edit $MYGVIMRC
```
<
Se per qualche ragione non voleste utilizzare il file gvimrc normale, potreste specificarne un altro con l'argomento "-U": >

```
gvim -U thisrc ...
```

Questo permette di avviare gvim per differenti tipi di lavori. Potete, per esempio, impostare un'altra dimensione di font.
Per evitare definitivamente la lettura del file gvimrc: >

```
gvim -U NONE ...
```

==

Capitolo seguente: |usr_32.txt| L'albero degli undo

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