usr_28.txt Per Vim version 7.3. Ultima modifica: 2008 Jun 14

VIM USER MANUAL - di Bram Moolenaar
Traduzione di questo capitolo: Giuliano Bordonaro

La piegatura

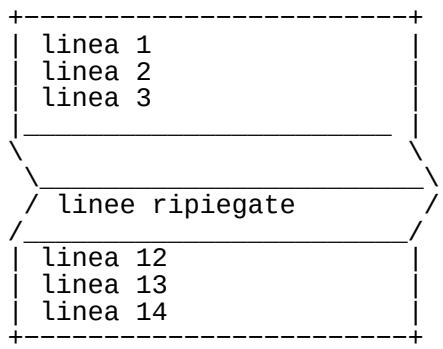
Un testo strutturato può essere diviso in sezioni. E le sezioni in sottosezioni. La piegatura vi consente di mostrare una sezione come una sola linea, consentendo una visione d'assieme. Questo capitolo spiega i diversi modi per farlo.

  Capitolo seguente: |usr_29.txt| Spostarsi attraverso i programmi
Capitolo precedente: |usr_27.txt| Comandi di ricerca e modelli
             Indice: |usr_toc.txt|

==========

28.1 Cosa vuol dire piegatura?

La piegatura serve per mostrare un gruppo di linee del buffer come se fossero una sola riga sullo schermo. Allo stesso modo con cui piegate un foglio di carta per renderlo più corto:



Il testo esiste ancora nel buffer, intatto. Soltanto che le linee visibili hanno una piegatura.

Un vantaggio conseguente alla piegatura è che potete godere di una migliore vista d'assieme della struttura del testo, piegando delle linee di una sezione e sostituendole con una sola linea che indica che lì c'è una sezione.

==========

28.2 Piegatura manuale

Provate: mettete il cursore entro un paragrafo e digitate: >

```
zfap
```

Vedrete che il paragrafo viene sostituito da una linea evidenziata. Avete effettuato una piegatura. |zf| è un operatore ed |ap| la selezione di un oggetto di testo. Potrete impiegare l'operatore |zf| con tutti i comandi di movimento per creare una piegatura del testo su cui vi siete spostati. |zf| lavora anche nel modo visuale.

Per rivedere il testo, dispiegatelo digitando: >

```
zo
```

E potrete richiudere la piegatura con: >

```
zc
```

Tutti i comandi di piegatura cominciano con "z". Con un po' di fantasia "z" ricorda un foglio di carta piegato visto da un lato. La lettera che segue la "z" ha un significato mnemonico per aiutarvi a rammentare i comandi:

```
zf      F-old creation  (Creare una piegatura)
zo      O-pen a fold    (Riaprire una piegatura)
zc      C-lose a fold   (Chiudere una piegatura)
```

La piegatura può venire annidata: una regione di testo che contenga delle piegature può essere nuovamente piegata. Ad esempio, potete piegare tutti i paragrafi di questa sezione e poi farlo con le sezioni costituenti questo capitolo. Provate. Vedrete che aprendo la piegatura che contiene l'intero capitolo verranno conservate le pieghe nidificate così come erano, alcune potrebbero essere aperte ed altre invece chiuse.

Supponete di aver fatto diverse piegature e di voler invece vedere tutto il testo. Potreste portarvi su ogni piega e digitare "zo". Per fare prima usate questo comando: >

```
zr
```

Questo R-idurrà la piegatura. Il contrario è: >

```
zm
```

Questo piega M-ore (nuovamente). Potete ripetere "zr" e "zm" per aprire e chiudere le pieghe annidate di molti livelli.

Se aveste annidato attraverso molti livelli, potreste aprirli tutti con: >

```
zR
```

Questo R-iduce le piegature non lasciandone alcuna. E potete chiudere tutte le pieghe con: >

```
zM
```

Questo piegherà M-olto e M-olto di più.

Potete togliere rapidamente le piegature con il comando |**zn**|. E |**zN**| riporta la piegatura come era. |**zi**| cambia tra le due. Questo è un modo proficuo di lavorare:
- creare le piegature per avere la vista d'assieme del vostro file
- spostarvi dove vi pare per fare il vostro lavoro
- fare |**zi**| per cercare del testo e lavorarci
- fare |**zi**| nuovamente per tornare indietro e ricominciare lo spostamento

Maggiori informazioni sulla piegatura manuale sono nel manuale di riferimento: |**fold-manual**|

==========

28.3 Lavorare con le piegature

Quando una piegatura è chiusa, i comandi di movimento come "j" e "k" la attraversano come se fosse una sola linea vuota. Ciò vi consente di spostarvi rapidamente sopra il testo ripiegato.

Potete copiare, cancellare ed incollare le piegature come se si trattasse di una sola linea. Ciò è utilissimo per riordinare delle funzioni entro un programma. Prima assicuratevi che ciascuna piega contenga tutta una funzione (o poco meno) selezionando il corretto **'foldmethod'**. Poi cancellate la funzione con "dd", spostate il cursore ed incollatela con "p". Se alcune linee della funzione fossero prima o dopo la piegatura, potreste usare la selezione Visual:
- ponete il cursore sulla prima linea da spostare
- premete "V" per entrare nel modo Visual
- portate il cursore sull'ultima linea che volete spostare
- premete "d" per cancellare le linee selezionate.
- spostate il cursore sulla nuova posizione e "p"ut (incollate) qui le linee.

È talvolta difficile vedere o ricordare dove si trovi una piegatura, ossia dove un comando |zo| potrebbe davvero operare. Per vedere le piegature definite: >

```
:set foldcolumn=4
```

Ciò mostrerà una piccola colonna alla sinistra della finestra per indicare le piegature. Un "+" viene mostrato per una piegatura chiusa. Un "−" è posto all'inizio di ciascuna piega aperta e "|" alle linee seguenti della piega.

Potete usare il mouse per aprire una piega facendo clic sul "+" nella colonna della piegatura. Fare clic sul "−" o sul "|" chiuderà una piegatura aperta.

Per aprire tutte le pieghe sulla linea del cursore usate |zO|.
Per chiudere tutte le pieghe sulla linea del cursore usate |zC|.
Per eliminare una piega sulla linea del cursore usate |zd|.
Per eliminare tutte le pieghe sulla linea del cursore usate |zD|.

Se siete nell'Insert mode, la piega alla linea del cursore non è mai chiusa. Ciò vi consente di vedere ciò che state scrivendo!

Le piegature vengono aperte automaticamente quando si salta in giro o si sposta il cursore a sinistra od a destra. Ad esempio, il comando "0" apre la piega sotto il cursore (se 'foldopen' contiene "hor", che è il default). L'opzione 'foldopen' può essere cambiata per aprire pieghe per comandi specifici. Se volete che venga sempre aperta la linea sotto il cursore, fate così: >

```
:set foldopen=all
```

Attenzione: non potrete spostarvi entro una piega chiusa allora. Potreste voler usare ciò solo temporaneamente e poi tornare al settaggio di default: >

```
:set foldopen&
```

Potete far sì che le piegature si chiudano automaticamente quando uscite da esse: >

```
:set foldclose=all
```

Ciò riapplicherà 'foldlevel' a tutte le piegature che non contengono il cursore. Dovrete provarlo per sentire se vi piace. Usate |zm| per piegare ancora e |zr| per piegare di meno (ridurre le pieghe).

La piega è localizzata nella finestra. Ciò vi permette di aprire due finestre sullo stesso buffer, una con le pieghe e l'altra senza. Od una con tutte le pieghe chiuse ed una con tutte aperte.

==========

28.4 Salvataggio e ripristino delle piegature

Quando lasciate un file (iniziando ad editarne un altro), lo stato delle piegature viene perduto. Ritornando allo stesso file successivamente tutte le piegature aperte o chiuse manualmente saranno tornate al proprio default. Quando le pieghe sono state create manualmente, tutte le pieghe vanno perdute! Per salvare le piegature usate il comando |:mkview|: >

```
:mkview
```

Ciò scriverà nel file le impostazioni e le altre cose che influenzano la vista. Potete modificare quanto salvato con l'opzione 'viewoptions'. Riaprendo successivamente lo stesso file, potrete caricare nuovamente le viste: >

```
:loadview
```

Potete salvare fino a dieci viste di un file. Ad esempio, salvare le impostazioni attuali come terza vista e caricare la seconda: >

```
:mkview 3
:loadview 2
```

Notare che quando aggiungete o cancellate delle linee la vista potrebbe diventare non valida.
Così osservate l'opzione 'viewdir', che indica dove le viste sono state salvate. Potreste voler cancellare delle vecchie viste.

==
28.5 Piegature secondo i rientri

Definire le pieghe con |zf| costa un sacco di lavoro. Se avete strutturato il vostro testo dando un rientro più ampio alle componenti di livello più basso, potrete usare il metodo di piegatura dei rientri. Ciò creerà piegature per tutti gli insiemi di linee aventi lo stesso rientro.
Linee con un rientro più ampio diverranno piegature annidate. Ciò funziona assai bene con molti linguaggi di programmazione.

Provatelo impostando l'opzione 'foldmethod': >

```
:set foldmethod=indent
```

Così potrete usare i comandi |zm| e |zr| per piegare maggiormente o ridurre le piegature. È facile da vedere in questo testo di esempio:

```
This line is not indented
        This line is indented once
                This line is indented twice
                This line is indented twice
        This line is indented once
This line is not indented
        This line is indented once
        This line is indented once
```

Notare che la relazione tra l'estensione del rientro e la profondità della piegatura dipendono dall'opzione 'shiftwidth'. Ciascun valore 'shiftwidth' del rientro aggiunge un'unità alla profondità della piegatura. Ciò viene chiamato livello di piegatura.

Usando i comandi |zr| e |zm| adesso aumentate o diminuite il valore dell'opzione 'foldlevel'. Potete anche impostarlo direttamente: >

```
:set foldlevel=3
```

Ciò significa che tutte le piegature con tre volte il rientro 'shiftwidth' o più verranno chiuse. Più basso sarà il livello di piegatura, tante più piegature verranno chiuse. Quando 'foldlevel' è zero tutte le piegature vengono chiuse. |zM| pone 'foldlevel' a zero. Il comando opposto |zR| pone 'foldlevel' al livello più profondo presente entro il file.

Così ci sono due modi per aprire e chiudere le piegature:
(A) Impostando il livello di piegatura.
    Fornisce un velocissimo modo di "portarsi in alto" per vedere la struttura del testo, spostare il cursore, e "ritornare in basso" nel testo.

(B) Usando i comandi |zo| e |zc| per aprire o chiudere specifiche piegature.
    Consente di aprire soltanto le piegature volute, lasciando chiuse le altre.

Ciò può venire combinato: potete prima chiudere molte piegature usando |zm| poche volte e poi aprire una piega specifica con |zo|. Od aprire tutte le piegature con |zR| e poi chiuderne alcune con |zc|.

Ma non potete definire delle piegature manualmente se il 'foldmethod' è "indent", poiché ciò sarebbe in conflitto con la relazione tra il rientro ed il livello di piegatura.

Di più circa le piegature basate sul rientro nel manuale di riferimento:
|fold-indent|

==
28.6 Piegature mediante marker

Dei marker entro il testo vengono usati per specificare l'inizio e la fine di una regione di piegatura. Ciò dà un esatto controllo su quali linee siano incluse entro una piegatura. Lo svantaggio è che il testo necessita di essere

modificato.

Provatelo: >

```
:set foldmethod=marker
```

Testo di esempio, come apparirebbe in un programma C:

```
/* foobar () {{{ */
int foobar()
{
        /* return a value {{{ */
        return 42;
        /* }}} */
}
/* }}} */
```

Notate che la linea piegata mostrerà il testo prima del marker. Ciò è molto utile per capire cosa contenga la piegatura.

È fastidioso quando i marker non siano più accoppiati correttamente dopo aver spostato qualche linea. Ciò può essere evitato usando marker numerati. Esempio:

```
/* global variables {{{1 */
int varA, varB;

/* functions {{{1 */
/* funcA() {{{2 */
void funcA() {}

/* funcB() {{{2 */
void funcB() {}
/* }}}1 */
```

Ad ogni marker numerato comincia una piegatura del livello specificato. Ciò farà che qualsiasi piegatura di livello più alto finisca qui. Potete usare proprio dei marker numerati di partenza per definire tutte le piegature. Solo quando voleste esplicitamente terminare una piegatura prima che ne inizi un'altra dovrete aggiungere un marker di fine.

Di più circa le piegature con i marker nel manuale di riferimento:
|**fold-marker**|

28.7 Piegature secondo la sintassi

Per ogni linguaggio Vim usa un file di sintassi diverso. Ciò definisce i colori per le diverse componenti entro il file. Se state leggendo questo testo su di un terminale che supporti i colori, i colori che vedete vengono fatti con il file di sintassi "help".

Nei file di sintassi è possibile aggiungere oggetti di sintassi che abbiano l'argomento "fold". Questi definiscono una regione di piegatura. Ciò comporta di scrivere un file di sintassi ed aggiungergli questi oggetti. Non è semplice da fare. Ma una volta fatto, tutte le piegature avvengono automaticamente.

Poniamo di usare un file di sintassi esistente. Così non vi è più nulla da spiegare. Potete aprire e chiudere piegature come spiegato sopra. Le piegature verranno create e distrutte automaticamente scrivendo il file.

Di più circa le piegature secondo sintassi nel manuale di riferimento:
|**fold-syntax**|

28.8 Piegature secondo espressione

È come la piegatura basata sul rientro ma, invece di impiegare il rientro di una linea, una funzione utente verrà chiamata a calcolare il livello di piegatura di una linea. Potete usarlo per del testo ove qualcosa indichi quali linee debbano stare insieme. Un esempio è costituito da un messaggio e-mail dove il testo quotato viene indicato da un ">" prima della linea. Per piegare queste quote usate questo: >

```
        :set foldmethod=expr
        :set foldexpr=strlen(substitute(substitute(getline(v:lnum),'\\s','',\"g\"),'
[^>].*','',''))
```

Potete provarlo con il testo seguente:

```
> quoted text he wrote
> quoted text he wrote
> > double quoted text I wrote
> > double quoted text I wrote
```

Spiegazione per la 'foldexpr' usata nell'esempio (a partire dal centro):

```
   getline(v:lnum)                    prende la linea corrente
   substitute(...,'\\s','','g')       toglie tutti gli spazi bianchi dalla
                                      linea
   substitute(...,'[^>].*','','')     toglie tutto ciò che segue '>'
   strlen(...)                        misura la lunghezza della stringa,
                                      pari al numero di '>'s trovati
```

Notare che una barra rovesciata deve essere inserita davanti ad ogni spazio, virgolette doppie e barra rovesciata per il comando ":set". Se ciò vi confonde usate >

```
        :set foldexpr
```

per vedere l'attuale valore risultante. Per correggere un'espressione complicata usate il completamento automatico da linea di comando: >

```
        :set foldexpr=<Tab>
```

Dove **<Tab>** è il vero Tab. Vim completerà il valore precedente che potrete allora modificare.

Di più circa la piegatura secondo espressione nel manuale di riferimento: |fold-expr|

==========

28.9 Piegatura delle linee non modificate

Risulta utile se impostate l'opzione 'diff' nella stessa finestra. Il comando |vimdiff| lo farà per voi. Esempio: >

```
        :setlocal diff foldmethod=diff scrollbind nowrap foldlevel=1
```

Fatelo in ogni finestra che mostri una versione diversa dello stesso file. Vedrete chiaramente le differenze tra i file, fin quando il testo non modificato rimane piegato.

Per maggiori dettagli vedere |fold-diff|.

==========

28.10 Quale metodo di piegatura usare?

Tutte queste possibilità inducono a domandarsi quale metodo scegliere. Purtroppo non esiste una regola d'oro. Ecco alcuni suggerimenti.

Se c'è un file di sintassi con le piegature per il linguaggio che state utilizzando, questa è probabilmente la scelta migliore. Se non ce n'è uno potreste provare a scriverlo. Ciò richiede una buona conoscenza delle stringhe di ricerca. Non è facile, ma quando funziona non dovete definire le piegature manualmente.

Piegare manualmente delle regioni scrivendo comandi può essere usato per il testo non strutturato. Allora usate il comando |:mkview| per salvare e ripristinare le vostre piegature.

Il metodo dei marker richiede che voi modifichiate il file. Se condividete i file con altri o dovete rispettare gli standard della ditta, potreste non essere in grado di aggiungerne.

Il vantaggio principale dei marker è che li potete mettere esattamente dove li volete. Ciò evita che qualche linea vada persa quando tagliate ed incollate delle piegature. E potete aggiungere un commento su cosa sia contenuto della piegatura.

La piegatura badata sul rientro è qualcosa che funziona in molti file, ma non sempre benissimo. Usatela quando non potete usare uno degli altri metodi. Comunque è molto utile per delineare. A questo fine, usate una sola **'shiftwidth'** per tutti i livelli di annidamento.

Piegare mediante espressioni può fare le piegature in quasi tutti i testi strutturati. È anche semplice da specificare, particolarmente se l'inizio e la fine di una piegatura può essere trovato facilmente.
Se usate il metodo "expr" per definire le piegature, ma non vi vengono come vorreste potete passare al metodo "manual". Ciò non rimuoverà le piegature già definite. Così potrete cancellare od aggiungere piegature manualmente.

==========

Capitolo seguente: |**usr_29.txt**| Spostarsi attraverso i programmi

Copyright: vedere |**manual-copyright**| vim:tw=78:ts=8:ft=help:norl:

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