Eine Einführung in Vim (Vi-IMprofed)

Urs Stotz

12. Januar 2004

Zusammenfassung

Dieser Vortrag wurde an der LUG Aargau am 28. November 2003 vorgetragen. Er lässt sich hier herunterladen und enthält LaTeX, Makefile und generierte Dokumente als dvi, ps, pdf, html und txt.
In diesem Vortrag zeige ich auf, wie man mit Vim arbeitet. Da Teilnehmer mit unterschiedlicher Kenntnislage von Vim anwesend sein werden, gehe ich von den Anfängern aus, werde aber schnell auf die erweiterten Funktionen von Vim übergehen. Dabei hoffe ich, dass es auch für Kenner einige neue und interessante Dinge zu sehen gibt, und die Anfänger durch die tollen Möglichkeit von Vim angespornt werden Vim besser kennen zu lernen.

Emacs-Jünger mögen mir dies verzeihen:

```
$ vim --noplugin -T vt100 -e -s ~/News/Score \
    -c /de.comp.editoren -c .,+2p -c q

[de.comp.editoren]
Score: =-9999
   Subject: [eE]macs
```

Inhaltsverzeichnis

1 Allgemeines über Vim

Geschichte: Vi baut auf ex auf. Ex ist eine Erweitertung von ed und arbeitet wie ed zeilenorientiert. Vi ist im Gegensatz zu ex bildschirmorientiert, darum der Name vi „visual editor“. Vi ist die visuelle Betriebsart des zugrundeliegenden Zeileneditors ex. Vim ist ein Clone von vi mit vielen Erweiterungen. Vim wurde und wird von Bram Moolenaar entwickelt. Moolenaar benutzte auf einem Amiga Computer den vi Clone stevie. Von dessen Quellcode ausgehend, entwickelte er Vim. Die erste Version 1.14 von Vim wurde auf einer Sammlung freier Software für den Amiga am 2. November 1991 veröffentlicht, das war auf der „Fred Fish Diskette 591“.

unterstützte Betriebssysteme: Aix, AmigaOS, Atari MiNT, BeOS, DOS, FreeBSD, HP–UX, Linux, NetBSD, MacOS, NextStep, OpenBSD, OS/2, OSF, RiscOS, SGI, Solaris, UNIX, VMS, WIn16 + Win32.

Copyright/Lizenz: Das Copyright liegt in den Händen des Hauptautors Bram Moolenaar `bram@vim.org`. Vim ist „charity-ware“, das heisst man wird dazu ermutigt eine Spende an Waisenkinder in Uganda zu machen (siehe „:help uganda“).

2 Die 6 Vim Modi

Vim kennt 6 Betriebsmodi. Die Statuszeile unten zeigt an, in welchem Modus man sich befindet.

„normal mode“ hier lässt sich der Text durch die Editor Kommands bearbeiten. Es ist die Betriebsart wenn Vim gestartet wird (insofern die Option „insert mode“ nicht gesetzt ist).

„insert mode“ in diesem Modus wird der Text eingegeben. Mit Tasten wie den Pfeiltasten, kann man waehrend des Editierens die Position wechseln.

„command-line mode“ Dies ist der Modus indem Ex Kommandos abgesetzt werden. Die Kommandos werden in der unteren Zeile eingegeben. Man wechselt aus dem „normal mode“ mit `:` in den „command-line mode“. Hier kann man suchen, ersetzen oder den Text filtern zum Beispiel so: `:.,+20 s/foo/bar/g`

„ex mode“ Das ist wie der „command-line mode“ nur bleibt man nach dem Kommando im „ex mode“. Aus dem „normal mode“ wechselt man mit `Q` in den „ex mode“ und mit `vi` verlässt man ihn wieder.

„visual mode“ das entspricht dem „normal mode“ mit dem Unterschied, dass Bewegungskommandos den Text highlighten und Befehle sich auf diesen Bereich beziehen. Mit der Tastenkombination `CTRL-v` wechelt man aus dem „normal mode“ in den „visual mode“.

„select mode“ die Befehle beziehen sich auf den markierten Text. Er sieht ähnlich wie der „visual mode“ aus, aber die Befehle unterscheiden sich zu diesem. Wenn man sich im „visual mode“ befindet, kann man z.B. mit `gh` in den „select mode“ umstellen.

3 Die wichtigsten Grundbefehle

Moduswechsel:

`i`	vom „normal mode“ in den „insert mode“ wecheln
`<Esc>`	vom „insert mode“ in den „normal mode“ wecheln

Bewegungsbefehle im „normal mode“:

`j`	Bewegung nach unten
`k`	Bewegung nach oben
`l`	Bewegung nach rechts
`h`	Bewegung nach links

Editierbefehle im „normal mode“:

`dd`	Zeile löschen
`x`	Zeichen löschen
`r`	Zeichen unterhalb des Cursers ersetzen
`yy`	Zeile kopieren
`p`	kopierte Zeile unterhalb des Cursers einfügen
`P`	kopierte Zeile oberhalb des Cursers einfügen
`u`	Undo der letzten Änderung
`<CTRL>-R`	Redo Rücknahme der letzten Undo

Speichern und beenden:

`:w`	Datei speichern
`:w!`	schreibgeschützte Datei speichern
`:wq`	Datei speichern und Vim beenden
`:wq!`	schreibgeschützte Datei speichern und Vim beenden
`ZZ`	Vim beenden und Datei speichern

4 Die Konfiguration von Vim

Die Datei `.vimrc`

Meine Konfiguration kann man von hier: http://typedef.ch/stotz/config/vim/ herunterladen. Sie enthält Plugins und ein update Script.

```
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
" Download:  http://typedef.ch/stotz/config/vim/.vimrc
" File:      $HOME/.vimrc
" Version:   1.1.0
" Date:      2005-01-15
" Author:    Urs Stotz <stotz@typedef.ch>
" Comment:   Please send me suggestion for improvement
"            Tested on: Debian, Sun Solaris
"            and Cygwin on Windows hell.
"            For vim and gvim you will need only .vimrc
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
" vim settings "
""""""""""""""""""
version 6.0
" set autoindent
" the same as |:set backspace=indent,eol,start|
set backspace=2
" keyword completion with dictionary the command is: |i_CTRL-X_CTRL-K|
set dict=$HOME/.vim/dict/perl,$HOME/.vim/dict/programming
set formatoptions=tcqr
" by a compressed helpfile
" set helpfile=$VIMRUNTIME/doc/help.txt.gz
set incsearch
" always show status line
set laststatus=2
" use extended regular expressions
set magic
set nobackup
" use Vim features
set nocompatible
" quiet and peaceful (your colleges thanks you)
set noerrorbells
" exrc is a potential security leak
set noexrc
" I don't like hlsearch
set nohlsearch
```

```
set nonumber
set wrap
" show the cursor position all the time
set ruler
" Show (partial) command in status line.
set showcmd
set showmode
set smartindent
" create new window below current one
set splitbelow
set splitright
set suffixes=.bak,~,.swp,.o,.info,.aux,.dvi,.bbl,.blg
set suffixes+=.brf,.cb,.ind,.idx,.ilg,.inx,.out,.toc
" Don't wrap words by default
set textwidth=0
" For terminals where scrolling is very slow and redrawing is faster.
" set ttyscroll=3
set viminfo=\"200,%,'200,/100,:100,@100,f1,h,n$HOME/.vim/viminfo
" wildmenu : command-line completion operates in an enhanced mode
set wildmenu

""""""""""""
" tabstop "
""""""""""""
" see :help 'tabstop'
set shiftwidth=2
set tabstop=2
set smarttab
set expandtab

"""""""""""""""
" c settings "
"""""""""""""""
set cindent
set cinoptions=:0,p0,t0
set cinwords=if,else,while,do,for,switch,case

"""""""""""""""""""
" color settings "
"""""""""""""""""""
if !has ("gui_running")
    " if vim in a terminal
    " http://www-2.cs.cmu.edu/~maverick/VimColorSchemeTest/index-perl.html
    colorscheme default
```

```
    set background=dark
    " hi Normal  ctermfg=white ctermbg=black
    " hi PreProc ctermfg=gray  ctermbg=black
else
    " set guifont=fixed
    " set guifont=Monospace\ 10
    " http://www-2.cs.cmu.edu/~maverick/VimColorSchemeTest/index-perl.html
    colorscheme blackbeauty
    set background=dark
    " colorscheme default
    " set background=light
    " hi PreProc guifg=#c0c0c0 guibg=#000040
    " hi Normal  guifg=black   guibg=white
endif

if has("syntax") && &t_Co > 2
  syntax on
endif

""""""""""""
" maping "
""""""""""""
" useful for Comments with username and date
map #log <ESC>mq:r!date +"\#$USER \%Y-\%m-\%d \%k:\%M:\%S"<CR>`qJ
" map CTRL-] to \\ (useful by german keyboards)
map \\  <C-]>
iab _PERL #!/usr/bin/perl -w<CR><CR>use strict;<CR>use warnings;<CR>
" I forget always the last digest
iab _NPI    3.14159265358979323846264338327950288419 72
iab _NEULER 2.71828182845904523536028747135266249775 73
" Make p in Visual mode replace the selected text with the "" register.
vnoremap p <Esc>:let current_reg = @"<CR>gvdi<C-R>=current_reg<CR><Esc>

""""""""""""""""""""""""""""""""""""""""
" Additional settings and parameters "
""""""""""""""""""""""""""""""""""""""""
filetype plugin on
filetype indent on
let html_use_css = 1
let use_xhtml = 1
```

5 Das Hilfesystem

Das Hilfesystem umfasst mehr als 60'000 Textzeilen. Die Hilfe zur Hilfe erhält man mit dem Befehl `:he he` Die Hilfe bezieht sich auf Suchwörter und Befehle. Da die Befehle vom Modus abhängig sind, kann man den Modus bezeichen. Für den „normal mode“ kann man ganz normal den Befehl eingeben z.B. `:he dd`

Der Syntax für die Hilfe ist:

Syntax	Beispiel	
(nichts)	`:he dd`	„normal mode“ z.B.
`c_`	`:he c_<Left>`	„command-line mode“ z.B.
`:`	`:he :m`	„ex mode“ z.B.
`i_`	`:he i_CTRL-W`	„insert mode“ z.B.
`v_`	`:he v_u`	„visual mode“ z.B.
`CTRL-`	`:he CTRL-A`	Kontroll-Zeichen z.B.
`-`	`:he -c`	Vim Startoptionen z.B.
`'`	`:he 'background'`	Vim Optionen z.B.

Weitere Informationen oder Hilfe findet man unter: www.vim.org, www.google.com und im Usenet in der Usergroup de.comp.editoren

6 Schneller editieren

i_CTRL-P	Wort ergänzen mit Suche nachdem Curser
i_CTRL-N	Wort ergänzen mit Suche vordem Curser
*	Vorwärtssuche nach dem Wort unter dem Curser
#	Rückwärtssuche nach dem Wort unter dem Curser
gd	springt zu einer local definierten Variable
gD	springt zu einer global definierten Variable
m{a-z}	Marke setzen gilt er nur innerhalb der aktuellen Datei
m{A-Z}	Marke setzen gilt auch ausserhalb aktuellen Datei
'{a-zA-Z}	springt zum Zeilenbeginn der gesetzten Marke
`{a-zA-Z}	springt zur Position der gesetzten Marke
[count]|	springt zur [count] Spaltenposition
CTRL-v	Block markieren
CTRL-V	Block markieren Zeilenorientiert
i_CTRL-W	löscht Wort vor dem Cursor
i_CTRL-U	löscht alle Wörter vor dem Cursor
i_CTRL-T	fügt „shiftwidth“ Leerzeichen ein
i_CTRL-D	löscht „shiftwidth“ Leerzeichen
i_CTRL-A	fügt vorher eingegebenen Text ein
q{0-9a-zA-Z"}	Macro aufzeichen
@{0-9a-zA-Z"}	Macro [count] ausführen
@@[count]	vorherges [count] Macro Aufzeichnung ausführen
[count]CTRL-A	Zahl [count] oder 1 addieren
[count]CTRL-X	Zahl [count] oder 1 subtrahieren
:[range]ce	Zeilen zentrieren „textwidth“
"ayw	kopiere Wort in benanntes Register a
"ap	paste Wort aus benanntem Register a
:registers	zeigt den Inhalt der 9 verschiedenen Register

7 Editieren mit ex

:%s/foo/any/c	ersetzt 1. foo in Zeile mit any mit Rückfrage
:%s/\(foo\) \(any\)/\2 \1/g	ersetzt jedes Vorkommen von foo mit any ohne Rückfrage

8 Das Window System

`:sp .`	splittet das aktuelle Window
`:vert diff foo`	splittet das aktuelle Window vertical mit diff zu einen anderen Datei
`CTRL-W [count] W`	springt zum nächsten `[count]` Window
`CTRL-W [count] P`	springt zum vorhergehenden `[count]` Window
`CTRL-W s`	Window horizontal splitten
`CTRL-W v`	Window vertical splitten
`CTRL-W v`	Window close
`CTRL-W o`	Window only

9 Mehrere Dateien editieren mit buffers

`:buffers`	zeigt alle buffers
`:[N]bu`	wechselt zu `[N]` buffer
`:[N]sbu`	wechselt zu `[N]` buffer in einem gesplitteten Windwo

10 Programmieren mit Vim

`:make`	make ausführen
`:clist`	listet die Fehler auf
`:cn`	zum nächsten Fehler springen
`:cp`	zum vorhergehenden Fehler springen
`:CTRL-]`	zur Definition des Tags unter dem Curser springen
`:tags`	tags auflisten
`:ts`	tags zum selektieren auflisten
`:ta foo`	zu tag foo springen
`:sta foo`	zu tag foo springen und Window splitten