Vim für Nicht-Mehr-Beginner und Noch-Nicht-Fortgeschrittene

(Edition 2016)

Agenda

- vimtutor auffrischen
- Einführung in das Hilfesystem/Windows/Buffers
- vim != vi
- Bewegen und Editieren in Dateien
- .vimrc/.viminfo
- Shortcuts
- Makros
- Tabs/visual mode/Skeletons/Sonstiges
- noch Fragen?

Persönliches

- Name: Toni Zimmer
- Beruf: „SysAdmin“
  - Konfigurationsdateien editieren
    → vim
  - Log-Dateien durchstöbern
    → vim -R
  - Dokumentation schreiben
    → Wiki → firefox → it's all text → (g)vim
  - Python/Bash-Scripte
    → vim
  - Mailing
    → mutt → vim

Vorbereitung

- Wiederholung
  - modaler editor
  - spezielle „Befehle“ zum Navigieren/Editieren
  - <Esc> is your friend (or <C-c>)

- vimtutor anyone?
  - Definition: vimtutor durchgespielt == Nicht-Mehr-Beginner

vimtutor (1)

- hjkl – zum „einfachen“ Bewegen
- :q! – zum Beenden
- x – löscht ein Zeichen
- i – zum Einfuegen (insert (mode))
- a / A – zum Anhaengen nach Cursor/ans Zeilenende (**„append“**)
- :wq – zum Speichern und Schliessen (**„write/quit“**)

vimtutor (2)

- dw – ein Wort loeschen
- d$ / D – bis zum Ende einer Zeile loeschen
- w / e – zum Anfang/Ende des naechsten Wortes bewegen
- 2w – zum Anfang des übernächsten Wortes bewegen
- 3e – zum Ende des dritten Wortes bewegen
- d2w – lösche die folgenden zwei Wörter
- 0 / $ – zum Anfang/Ende der aktuellen Zeile bewegen
- dd / 2dd– lösche aktuelle (und kommende) Zeile
- u / U – mache Änderung (in ganzer Zeile) rückgängig
- <C>-r – stelle Änderung wieder her („redo“)

vimtutor (3)

- p – hänge zuvor gelöschten Text nach dem Cursor an (put)
- r – ersetze (replace) Zeichen unter Cursor
- ce – ändere (change) bis zum Ende des aktuellen Wortes
- c$ / C – ändere bis zum Zeilenende

- <C-g> – zeige Dateiposition/-status an
- gg – gehe zum Dateianfang
- G – gehe zum Dateiende
- 3G – gehe an den Anfang der dritten Zeile

vimtutor (4)

- / – suche vorwärts
- ? – suche rückwärts
- n / N – gehe zum nächsten/vorherigen Treffer der letzten Suche
- <C-o> – gehe zum vorherigen „Standort“ des Cursors (**„older“**)
- <C-I> – gehe wieder „vor“
- % – gehe zur zugehörigen Klammer ({[

vimtutor (5)

- :%s/alt/neu/gc – ersetze jedes "alt" durch "neu" in allen Zeilen, frage aber vorher nach
- :!ls – fuehrt den Befehl "ls" aus
- :w name – speichert aktuellen „buffer“ unter "name" ab
- :r name – liest Inhalt der Datei "name" in aktuellen „buffer“ ein
- :r!ls – schreibt Ausgabe des Befehls "ls" in aktuellen „buffer“

vimtutor (6)

- o / O – fuege neue Zeile unter/über aktueller Position ein
- e – springe zum Wortende
- y – kopiere (**„yank“**) Text
- p – fuege kopierten Text ein (**„put“**)
- R – ueberschreibe Text bis zum Beenden des Insert-Mode
- :set bla setze Option „bla“
- :set nobla deaktiviere Option „bla“ (Bsp: ic, is, hls)

Help! (1)

- :help bzw. :h bzw. <F1> (wenn das Terminal das erlaubt)
- :h command/option/key
- jump via tags: <C-]> (zurueck mit <C-o> - older position oder <C-t>)
- :h help-summary
- :h i_CTRL-R (listet Hilfe zu <C-r> im Insert-Mode auf
- :help hls <C-d> (mit Tab durch die Liste wandern)

Help! (2 - windows)

- :help windows
- <C-w><C-w> lässt Hilfe offen und springt in ursprüngliches Fenster
- <C-w>_ maximiert aktuelles Fenster
- <C-w>= bringt beide Fenster auf gleiche Größe
- <C-w>x wechselt die Fenster („e**x**change“)
- <C-w>s öffnet neues Fenster (**„split“**)
- <C-w>c schliesst aktuelles Fenster (**„close“**)

Help! (3 – buffers)

- :ls / :buffers
- :b <Tab> / :b 1
- :b# - springe zwischen aktuellem/letzten Buffer
- :bd – schliesse buffer (**„delete“**)
- :sbuffer

Help! (4 - Übung)

- Hilfe zum Thema xyz Starten
- zu einem anderen Thema navigieren (und wieder zurückspringen)
- Window auf „fullscreen“ vergrößern
- :help! → was bedeutet die Fehlermeldung?
- Hilfe schließen

Exkurs: vim != vi

- (nearly) compat mode
- vim nicht auf allen Plattformen verfügbar
- kein Syntax highlighting, keine Hilfe, kein undo-Stack, kein visual mode, keine Macros
- :h vi_diff

Bewegen (1)

- hjkl UND cursor-Tasten
- gg/G/20G/20gg/:20
- / und ? (suchen)
- n/N (nächster/vorheriger Treffer)
- w/e/b/ge
- W/E/B/gE
- 0/^/$
- H/M/L
- */#

Bewegen (2)

- zt (top)
- zb (bottom)
- zz
- <C-f> / <C-b> – forward/backwards
- <C-u> / <C-d> – up/down
- <C-e> / <C-y>
- :set scroll

Bewegen (3) - marker

- mx - setze Marke mit dem "Namen" x
- 'x - gehe in die Zeile, in der die Marke x gesetzt wurde
- `x - gehe direkt an die Stelle, an der die Marke gesetzt wurde
- `` - springe zwischen aktueller Marke und letzter Postion hin und her
- `. - springe zur letzten Aenderung
- :marks

Editieren (1) – copy'n'paste

- ywP – Wort kopieren
- ddp – Zeile tauschen
- yyp – Zeile verdoppeln
- x / X – lösche Zeichen / vor dem Cursor
- xp
- “ayy / Y – kopiere (**„yank“**) Zeile in Register „a“
- :reg / <C-r> + Register

Editieren (2) – text objects

- aw / iw
- a“ / i“
- a' / i'
- a(/ i(
- at / it
- ap / ip
- as / is
- :help text-objects

Editieren (3) – text object combos

- dw
- diw
- daw
- cw
- ciw
- yw
- ...

Editieren (4) – misc

- <C-p> / <C-n> – Wort vervollständigen
- <C-x><C-l> – Zeile vervollständigen
- J/gJ – Zeilen „joinen“ (ohne Whitespace)
- <C-a> / <C-x> – hoch-/runterzaehlen (Obacht bei fuehrender Null)
- :read filename
- :read !command
- :!%
- :%s/alt/neu/gc
- :set paste

Editieren (5) – Übung

- Gedicht (mit Hilfe von Registern) in die richtige Reihenfolge bringen

  purr purr purr

  ball fur little of

  kitty kitty soft warm

  happy sleepy kitty kitty

  →

  soft kitty warm kitty

  little ball of fur

  happy kitty sleepy kitty

  purr purr purr

.vimrc

- :set hlsearch → set hlsearch
- :set number → set number
- ...
- vimscript
- Kommentar beginnt mit “

.viminfo

- „history“ (Suchbegriffe, editierte Dateien, „command line“, Register-Inhalte, Markierungen, Makros...)

Abkürzungen

- :ab fa Feierabend
  - wirkt im Insert-Mode UND in ex-command line
- :iab fa Feierabend
  - wirkt nur im Insert-Mode
- :iab HAllo Hallo
- wirkt auch bei copy'n'paste via MMB!
- temp. deaktivieren: set paste
- :h :ab

Mappings

- :map <F4> ddp
- :map <F5> ddkP
- :map V IViele Grüße<CR><CR><Esc>
  - V ist aber eigentlich belegt
- :map <Leader>V IViele Grüße<CR><CR><Esc>

- :let mapleader=“,“
- :[verbose] nmap/vmap/imap/map

Mappings – Übung

- definiere ein Mapping, welches zwei benachbarte Wörter vertauscht

Abkürzungen – Übung

- Schreibe folgende „Mail“ und lasse die Abk. durch den „richtigen“ Text ersetzen:

  Hallo,

  hab Dich tel nicht erreicht, dh baW ist der WLAN-AP für die VoIP-Telko NA.

  MfG

Makros

- Arbeitsschritte zur Wdhlg. aufzeichnen
- :reg
- qx – beginne mit Aufzeichnung und speichere in Register 'x'
- <Esc> ggf. zum Beenden Insert-Mode
- q – beende Aufzeichnung
- @x – „Abspielen“ des Makros 'x'
- Q → :vi

Makros – Übung

- erzeuge „munin.conf“:
  - 10 Rechnernamen (host1 – host10) in folgendes Schema pressen:

    [workstation;**host1**]

    address **host1**

    use_node_name yes

Tabs

- vim -p file1 file2 file3
- gt/gT – geh zum nächsten/vorherigen Tab
- 2gt – geh zum zweiten Tab
- :tabnew – öffne neuen unbenannten Tab
- :tabe blub – öffne neuen Tab mit Datei „blub“
- :tab help gt – öffne Hilfe zu „gt“ in neuem Tab

Visual mode

- v – starte visual mode (s. Status-Zeile)
- V – starte visual mode zeilenweise
- <Esc> – beende visual mode
- gv – markiere letzten Bereich
- :help visual-operators
- :help **V**isual

Visual block mode

- <C-v> – starte visual block mode (s. Status-Zeile)
- beliebige Bewegung um Block zu markieren
- o – gehe an das andere Ende des Blocks
- O – gehe an das andere Ende des Blocks auf gleicher Zeile
- Zeilenende ($) wird "intelligent" erkannt/angepasst

Visual (block) mode – Übung

- IMGP001.jpg 10x kopieren und hochzählen
- anschl. in „html“-Format bringen:

  IMGP001.jpg

  IMGP002.jpg

  ...

Skeletons

- „templates“ für neu erzeugte Dateien
- :help skeleton
- vimrc:
  - **au BufNewFile *.py 0r /path/to/skeleton.py**

Sonstiges

- vimdiff
- g/re/p
- :h i_Ctrl-o
- g??

Danke für die Aufmerksamkeit!