VIM - Vi IMproved

- ganze Textbereiche ein- oder auskommentieren
- Mein VIM-Setup (vimrc und Plugins)
- Serial Incrementer für DNS SOA per Tastendruck
- VIM - Tipps und Tricks
- VIM HowTo

ganze Textbereiche ein- oder auskommentieren

Immer wieder steht man vor dem Problem mit dem VI schnell mal einen ganzen Codeblock auskommentieren zu müssen. Mit dem Blockmodus von VI geht das auch ganz einfach.

1. in die Anfangszeile wechseln und STRG+V drücken, dies aktiviert den Blockmodus (im Status erscheint: – VISUAL BLOCK –)
2. jetzt den Cursor bis zur gewünschten Zeile nach unten (Cursortaste oder j) verschieben, der Block dazwischen wird dadurch ausgewählt
3. Mit shift+i (großes I) in den Eingabemodus wechseln und das gewünschte Kommentarzeichen (#, / /) eingeben.
4. Mit ESX zurück in den Befehlsmodus, VI hat jetzt an den Anfang jeder Zeile das gewünschte Zeichen eingefügt
5. Sollen die Zeilen wieder einkommentiert werden muss der Block zuerst wie vorher wieder markiert werden. Bei mehr als einem Zeichen werden die weiteren mit der Cursortaste oder l markiert. Zum löschen x drücken.

Mein VIM-Setup (vimrc und Plugins)

Tastenkürzel	Aktion
STRG+N	nächste Datei öffnen (wenn Vim mit mehreren Dateien gestartet wurde)
STRG+P	vorherige Datei öffnen (wenn Vim mit mehreren Dateien gestartet wurde)
F3	Tabs als >- anzeigen umschalten

Folgende Pakete müssen installiert sein:

```
xxxxxxxxxx
```

```
1
```
```
apt update
```

```
2
```
```
apt install vim-common vim-nox vim-runtime vim-tiny vim-scripts vim-addon-manager vim-pathog
```

Für Fancy-Aussehen, Syntax-Checks und so weiter habe ich diese Vim-Addons installiert:

```
xxxxxxxxxx
```

```
mkdir -p /opt/vim/bundle
```

```
# a fancy file tree (envoked by CTRL+k):
```

```
git clone https://github.com/scrooloose/nerdtree /opt/vim/bundle/nertdtree
```

```
?
```

```
# this pimps the status line
```

```
git clone https://github.com/itchyny/lightline.vim /opt/vim/bundle/lightline
```

```
?
```

```
# this is for smart copy and paste (without automatic indent and messing around with :set pa
```

```
git clone https://github.com/ConradIrwin/vim-bracketed-paste /opt/vim/bundle/bracketed-paste
```

```
?
```

```
# make some often used shell commands available
```

```
git clone https://github.com/tpope/vim-eunuch.git /opt/vim/bundle/eunuch
```

```
?
```

```
# extended syntax checks
```

```
git clone https://github.com/vim-syntastic/syntastic.git /opt/vim/bundle/syntastic
```

```
?
```

```
# optimzize colors
```

```
git clone https://github.com/altercation/vim-colors-solarized.git /opt/vim/bundle/colors-sol
```

Und hier noch die globale vimrc:

```
xxxxxxxxxx
```

```
1
"
2
" File managed by salt.ovtec.it (packages/vim).
3
" Your changes will be overwritten.
4
"
5

6
runtime! debian.vim
7

8
if exists('g:loaded_sensible') || &compatible
9
  finish
10
else
11
  let g:loaded_sensible = 1
12
endif
13

14
if has('autocmd')
15
  filetype plugin indent on
16
endif
17
if has('syntax') && !exists('g:syntax_on')
18
```

```
  syntax enable
endif

set autoindent
set autoread
set background=dark
set backspace=indent,eol,start
set cmdheight=2
set complete=i
set expandtab " enable this to use whitespaces instead of tabs
set formatoptions=qtj
set hidden
set incsearch
set iskeyword+=_,$,@,%,#,-
set laststatus=2
set lazyredraw
set modeline
set modelines=2
set nobackup
```

```
set nohlsearch
set nrformats=octal,hex,alpha
set ruler
set shiftwidth=2
set showcmd
set showmatch
set showmode
set smarttab
set softtabstop=2
set tabstop=2
set ttimeout
set ttimeoutlen=100
set visualbell
set wildmenu

" Use <C-L> to clear the highlighting of :set hlsearch.
if maparg('<C-L>', 'n') ==# ''
  nnoremap <silent> <C-L> :nohlsearch<C-R>=has('diff')?'<Bar>diffupdate':''<CR><CR><C-L>
```

```
endif
```

```

```

```
if !&scrolloff
```

```
  set scrolloff=1
```

```
endif
```

```
if !&sidescrolloff
```

```
  set sidescrolloff=5
```

```
endif
```

```
set display+=lastline
```

```

```

```
if &encoding ==# 'latin1' && has('gui_running')
```

```
  set encoding=utf-8
```

```
endif
```

```

```

```
if &listchars ==# 'eol:$'
```

```
  set listchars=tab:>-,trail:-,extends:>,precedes:<,nbsp:+
```

```
endif
```

```

```

```
if has('path_extra')
```

```
   setglobal tags-=./tags tags-=./tags; tags^=./tags;
 endif

 if &shell =~# 'fish$'
   set shell=/bin/bash
 endif

 if &history < 1000
   set history=1000
 endif
 if &tabpagemax < 50
   set tabpagemax=50
 endif
 if !empty(&viminfo)
   set viminfo^=!
 endif
 set sessionoptions-=options

```

```
" Allow color schemes to do bright colors without forcing bold.
```

```
if &t_Co == 8 && $TERM !~# '^linux\|^Eterm'
```

```
  set t_Co=16
```

```
endif
```

```

```

```
" I don't remember what that does
```

```
inoremap <C-U> <C-G>u<C-U>
```

```

```

```
" next and previous buffer / file
```

```
nnoremap <C-N> :next<Enter>
```

```
nnoremap <C-P> :prev<Enter>
```

```

```

```
" toggle displaying listchars
```

```
nnoremap <F3> :set list!<Enter>
```

```

```

```
au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "normal! g'\"" | endi
```

```
au BufRead,BufNewFile *.sls setfiletype sls
```

```
10
9
```

```
au FileType yaml,yml,sls setlocal ts=2 sts=2 sw=2 ai expandtab fo-=c fo-=r fo-=o
```

```
11
0
```

```
"au FileType * setlocal ts=2 sts=2 sw=2 ai expandtab fo-=c fo-=r fo-=o
```

```
11
1
```

```

```

```
11
2
```

```
" Load plugins
```

```
11
3
```

```
execute pathogen#infect('bundle/{}', '/opt/vim/bundle/{}')
```

```
11
4
```

```

```

```
11
5
```

```
" colorscheme
```

```
11
6
```

```
let g:solarized_termcolors=256
```

```
11
7
```

```
colorscheme solarized
```

```
11
8
```

```

```

```
11
9
```

```
" syntastic syntax checker
```

```
12
0
```

```
set statusline+=%#warningmsg#
```

```
12
1
```

```
set statusline+=%{SyntasticStatuslineFlag()}
```

```
12
2
```

```
set statusline+=%*
```

12
3

```
let g:syntastic_always_populate_loc_list = 1
```

12
4

```
let g:syntastic_auto_loc_list = 1
```

12
5

```
let g:syntastic_check_on_open = 1
```

12
6

```
let g:syntastic_check_on_wq = 0
```

12
7

```

```

12
8

```
" vim:set ft=vim et sw=2:
```

Serial Incrementer für DNS SOA per Tastendruck

Ein praktisches Tool für alle, die öfters mal DNS-Zonen editieren müssen (Autor Folke Ashberg).

Einfach den Code von unten in einer Datei im plugin-Verzeichnis vom VIM ablegen. Beim nächsten Start kann die Serial in der SOA mit dem neuen Kommando :DNSserial hochgesetzt werden. Die vorhandene Konvention wird dabei beibehalten, z.B. YYYYMMDDNN (NN=fortlaufende Nummer). Dieses Kommando kann man noch einfach einer Taste zuweisen, mein Beispiel legt den Befehl auf F6.

```
"DNSserial auf F6 legen
nmap <F6> :DNSserial<cr>
```

und hier noch der Code des Plugins (kann auch vom Autor direkt runtergeladen werden):

```
" DNS Serial Incrementer
" Author: Folke Ashberg <folke@ashberg.de>
" Copyright: 2001 by Folke Ashberg
" LAST MODIFICATION: Fre Sep 14 19:05:32 CEST 2001
" CVS: $Id: dnstools.vim,v 1.4 2002/08/15 11:08:15 folke Exp $
" Usage:
"         Serial Updater:
"         Just execute the command DNSserial and this tiny script
"         will increment the serial number, preserving that style you use :)



function DNS_getnum(oldnum)
    let oldnum = a:oldnum
    if oldnum < 19700101
 " 1, 2, 3 style
 let retval = oldnum + 1
    elseif oldnum < 1970010100
 " YYYYMMDD style
 let dateser = strftime("%Y%m%d")
 if dateser > oldnum
     let retval = dateser
 else
     let retval = oldnum + 1
 endif
    else
 " YYYYMMDDNN style
 let dateser = strftime("%Y%m%d00")
 if dateser > oldnum
     let retval = dateser
 else
     let retval = oldnum + 1
 endif
    endif
    return retval
endfun

function DNSserial()
    let restore_position_excmd = line('.').'normal! '.virtcol('.').'|'
    let oldignorecase = &ignorecase
    set ignorecase
    " substitute now ( there's a bug in VIM's vi Syntax :(   )
    " silent
    %s/\(soa[[:space:]]\+[a-z0-9.-]\+[[:space:]]\+[a-z0-9.-]\+[[:space:]]*(\?[\n\t ]*\)\([0-9]\+
    " restore position
    exe restore_position_excmd
    " disable hls
    if 1 == &hls
 noh
    else
 set hls
    endif
```

```
    " restore old case behave
    let &ignorecase=oldignorecase
endfun

command DNSserial :call DNSserial()



function DNSzone()
    let zone = input("Name der Zone: ")
    let ip   = input("IP: " , "62.26.219.121")
    let ins=""
    let ins = ins . "$INCLUDE /var/named/ttl-file\n"
    let ins = ins . ";       File: \"" . zone . ".dom\"\n"
    let ins = ins . ";       FQDN: \"" . zone . "\"\n"
    let ins = ins . "@       IN SOA ns1.dns-zone.net. hostmaster.dns-zone.net. (\n"
    let ins = ins . "                " . strftime("%Y%m%d00") . "; serial number\n"
    let ins = ins . "                3H              ; refresh\n"
    let ins = ins . "                1H              ; retry\n"
    let ins = ins . "                1W              ; expiry\n"
    let ins = ins . "                1D )            ; minimum\n"
    let ins = ins . "\n"
    let ins = ins . ";        Zone NS records\n"
    let ins = ins . "\n"
    let ins = ins . "@                               IN      NS      ns1.dns-zone.net.\n"
    let ins = ins . "@                               IN      NS      ns2.dns-zone.net.\n"
    let ins = ins . "\n"
    let ins = ins . ";        Zone MX records\n"
    let ins = ins . "\n"
    let ins = ins . "@                               IN      MX      30      mail\n"
    let ins = ins . "@                               IN      MX      60      mail3.iok.net.\n"
    let ins = ins . "@                               IN      MX      90      mail3.csl-gmbh.net.
    let ins = ins . "\n"
    let ins = ins . ";        Zone records\n"
    let ins = ins . "\n"
    let ins = ins . "                                IN      A       " . ip ."\n"
    let ins = ins . "www                             IN      CNAME   " . zone . ".\n"
    let ins = ins . "ftp                             IN      CNAME   " . zone . ".\n"
    let ins = ins . "smtp                            IN      CNAME   " . zone . ".\n"
    let ins = ins . "\n"
    put!=ins
endfun
```

VIM - Tipps und Tricks

Zeilen sortieren

```
# einen bestimmten Bereich sortieren (z.B. 5,10 -> Zeile 5 bis 10)
:{range}sort

# das ganze Dokument einfach sortieren
:sort

# nach Zahlen sortieren
:sort n

# umkehrte Sortierung
:sort!

# doppelte Zeilen entfernen (uniq)
:%sort u

# das externe UNIX Sort-Utility verwenden und nach Monatsnamen sortieren
:%!sort -M
```

TAB-Zeichen visualisieren

Bei Problemen mit Einrückungen kann man sich die Tabzeichen auch anzeigen lassen, damit läßt sich unterscheiden, ob es Leerzeichen oder Tabs sind.

```
:set list
:set listchars=tab:>-     " >
```

Copy&Paste mit aktiviertem autoindent

Wenn man bei aktiviertem Autoindent (automatisches Einrücken) vorformatierten Code einfügt, wird jede Zeile noch weiter eingerückt. Am Ende sieht das dann ziemlich kaputt aus. Das läßt sich einfach umgehen:

im Command-Mode:

```
:r! cat<enter>
<shift + insert>
<CTRL + d>
```

oder paste ein-/ausschalten:

```
:set paste
<insert modus>
<shift + insert>
<esc>
:set nopaste
```

oder auf eine Taste legen:

```
set pastetoggle=<F10>
```

VIM HowTo

Hier ein kleines HowTo mit Tipps und Tricks zu vim („vi improved“). Unten findet Ihr auch meine aktuelle vimrc.local

Mehrere Dateien editieren / Navigation

Befehl	was passiert
:n	nächste Datei öffnen
STRG-^ oder :e#	vorherige Datei öffnen (== „alternate file“)

Suchen

Befehl	was passiert
/muster	es wird vorwärts nach dem Begriff „muster“ gesucht
/muster/e	sucht vorwärts, Cursor springt ans Ende des Musters
?muster	es wird rückwärts gesucht
n	letzte Suche wiederholen

Ersetzen

die Suche unterstützt verschiedene Modifier, die werden am Ende gesetzt:

c	Nachfrage vorm Ersetzen
g	ganze Zeile bearbeiten
i	Groß-/Kleinschreibung beachten
I	Groß-/Kleinschreibung nicht beachten

Befehl	was passiert
:s/old/new/	ersetzt old durch new beim ersten Auftreten in der aktuellen Zeile
:s/old/new/g	ersetzt old durch new in der kompletten Zeile
:s/old/&_suffix/g	ersetzt old durch old_suffix in der aktuellen Zeile
:5,10s/old/new/g	ersetzt old durch new in Zeile 5 bis 10
:%s/old/new/g	ersetzt old durch new im kompletten Dokument
:s/^/text/	Text am Zeilenanfang einfügen
:s/$/text/	Text am Zeilenende einfügen

meine /etc/vim/vimrc.local

```
"
" magenbrot vimrc
"
" $Id: vimrc 125 2013-01-10 09:59:41Z magenbrot $
"

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "normal! g'\"" | endif
endif

" The following are commented out as they cause vim to behave a lot
" differently from regular Vi. They are highly recommended though.
set showcmd  " Show (partial) command in status line.
set ignorecase  " Do case insensitive matching
set showmatch  " Show matching brackets.
set smartcase  " Do smart case matching
set incsearch  " Incremental search
```