

Proseminar Linux für Umsteiger: VI und VIM

Betreuer: Julian Kunkel

Daniel Kruck
daniel.kruck@gmx.net

Universität Heidelberg, 10.06.2008

Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

Visual Interface: Historisches

- VI wurde von Billy Joy entwickelt
- Löst 1976 "ed" als Standard-Editor für Linux ab
- 1984 steigt der Editor Emacs auf, es entstehen zwei Lager:
VI- und Emacs-Fans



Visual Interface: Zitate



Visual Interface: Modi

Vi <Dateiname>

startet

Command-line mode	Normal mode	Insert mode
Beenden (:q) Eingabe und Ausführung von komplexen Befehlen	Tasten haben Funktionen z.b. y = yank d = delete	Zum normalen Text tippen

Visual Interface: Vor- und Nachteile

Vorteile

- Schnelles Arbeiten durch kurze Befehle
- Wenig Gebrauch von <strg> oder <alt> durch verschiedene Modi
- Sehr klein & flexibel
- Weit verbreitet

Nachteile

- Viel zu Lernen für Neueinsteiger durch Modi-Konzept

Visual Interface: Klone

- Vim
 - Tolle Features
 - Longtermsupport
- Elvis
 - Keine neuen Features für 2 Jahre, nur super Sourcecode --> interessiert User nicht --> das Aus für Elvis
- Nvi
 - Schnell, als erster Klon tolle Features wie Multibuffers usw
 - Aber: keine neuen Versionen mehr

Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

Vim Tutorial

- Survival Befehle: starten, beenden, Hilfe

Befehl	VIM	Emacs
Starte in shell	vim	emacs -nw
Starte als gui	gvim	emacs
beenden	:q	<strg>-x <strg>-c
Hilfe	:help	<strg>-h i

Tutorial Zusammenfassung

- „vim“ startet Vim in der shell
- :q <--schließt vim
- :help [<stichwort>]

Tutorial Zusammenfassung

- „vim“ startet Vim in der shell
- :q <--schließt vim
- :help [<stichwort>]
- Emacs-Fans: `rm -r /bin/vi /`

Gliederung

- Visual Interface
- Vim Tutorial
- **VIMRC**
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

VIMRC für Version 7

- Ist individuell für jeden Nutzer (~/.vimrc)
- Startet die gewünschte Konfiguration automatisch
- Editieren über :e \$MYVIMRC
- Erlaubt sind alle command-line Befehle

VIMRC - Beispiel

```
set nocompatible      "Nutzt VIM mit allen erweiterten Funktionalitäten. Es gibt
                      "eigentlich keinen Grund, dieses Kommando nicht zu
                      "verwenden. Weil VIM ist echt besser als VI;-)
set noswapfile        "soll verhindern, dass Fehler bei der Präsentation
                      "durch viele Zugriffe auf das gleiche File entstehen

set autoindent
set backspace=indent,eol,start

##### Speicherrelevantes #####
set history=200       "begrenzt die Anzahl der Undo-Vorgänge auf 200

##### Arbeiten mit verschiedenen Dateitypen #####
syntax on             "Syntaxhighlighting wird angestellt
set filetype plugin indent on  "lädt zugehörige Plugins
```

VIMRC - Beispiel

Optisches

set background=dark

set ruler

set showcmd

"dunkler Hintergrund

"zeigt rechts unten die Cursorposition an

"zeigt an, welche Tasten für ein

"Kommando schon gedrückt wurden

Suche

set incsearch

"fängt schon bei der Eingabe des ersten

"Buchstaben an nach passenden Mustern zu

"suchen:

"/a markiert erstmal das nächste a

"/ab sucht zunächst das nächste ab usw

"incsearch = incremental search

set hlsearch

"highlight search

"markiert alle treffenden Muster

Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

Effective Text Editing

- 1) Ineffiziente Arbeitsweise erkennen
- 2) Suche nach besseren Methoden:
 - ♦ Kollegen, Freunde, Irc-Community befragen
 - ♦ Suche im Netz nach besseren Möglichkeiten
 - ♦ Buch kaufen?
- 3) Neue Methoden zur Gewohnheit machen
 - ♦ .vimrc verändern
 - ♦ Vim-Script schreiben

Effectiv Text Editing

- Vier Beispiele:
 - 1) Lange Wörter mittels Abkürzungen (:abbr)
 - 2) Suchen und Ersetzen von Kommentaren
 - 3) Benutze das Python-Filetype Plugin
 - 4) Nutze Macros für häufige Arbeit

1) Abbreviations

- Gehe in vim
- Tippe `:abbr kub kubuntu`
- Gehe in den insert-mode und teste `kub<space>`

2) Search and Replace

- In shell: `vim kommentare.txt`
- Dort sind Kommentare durch `//` markiert
- Du willst sie durch `#` ersetzen
- Tippe `:%s/\\/#/g`

3) Python

- In shell: vim PythonAndVim.py
- ```
def quadratFunktion(a):
 a *= a
 if a < 10:
 print a
 else:
 print 'Quadrat zu groß ;-)
 return a
```
- `quadratFunktion(3)`

# 4) Macros

- 1) Drücke im normal mode qa ,um ein Macro aufzuzeichnen
- 2) Drücke q im normal mode, um die Aufzeichnung zu beenden
- 3) Drücke @a ,um das Macro auszuführen. Alles was zwischen 1) und 2) gesehen ist, wird genauso, als ob Du es eintippen würdest, wiederholt

# Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

# Weitere Informationsquellen

- Hilfen:

- `:help <Befehl>`
- `Irc: Network: freenode; common sever: irc.freenode.net;channel: #vim`
- Vim mailinglist @ <http://www.vim.org/>  
(viele Userfragen und Antworten)

- Tutorials:

- `shell: vimtutor`
- <http://www.vi-improved.org/tutorial.php> (sehr gut als Einstieg)
- <http://www.selflinux.org/selflinux/html/vim.html>

# Weitere Informationsquellen

- Sprachenspezifische Hilfe
  - <http://vim.wikia.com/wiki/Category:LanguageSpecific>
- Effektives Texte Editieren
  - <http://www.moolenaar.net/habits.html> (kurz und gut)
  - <http://video.google.de/videoplay?docid=2538831956647446078>
- Schnell schreiben
  - <http://speedtest.schnell-schreiben.de/>
  - <http://ktouch.sourceforge.net/>

# Fazit und Ausblick

- Vim ist sehr gut geeignet, Texte zu editieren
- Anfänger brauchen etwas Einarbeitungszeit
  
- Ausblick
- Das Zusammenspiel mit anderen Programmen dürfte noch verbessert werden:
  - Als externes Proggi:(KMail--> External Composer: `gvim "+set ft=mail" -f %f`); eine interne Lösung wäre schön

# Quellen

- <http://www.vim.org/>
- <http://de.wikipedia.org/wiki/Vi>
- [http://de.wikipedia.org/wiki/Billy\\_Joy](http://de.wikipedia.org/wiki/Billy_Joy)
- [http://www.theregister.co.uk/2003/09/11/bill\\_joy\\_s\\_greatest\\_gift/](http://www.theregister.co.uk/2003/09/11/bill_joy_s_greatest_gift/)
- <http://thomer.com/vi/vi.html#versions>
- <http://www.moolenaar.net/habits.html>
- <http://video.google.de/videoplay?docid=2538831956647446078>
- <http://www.vi-improved.org/>
- <http://de.youtube.com/watch?v=S76pHIYx3ik>
- <http://wiki.ubuntuusers.de/Vim>

# Proseminar Linux für Umsteiger: VI und VIM

Betreuer: Julian Kunkel

Daniel Kruck  
[daniel.kruck@gmx.net](mailto:daniel.kruck@gmx.net)

Universität Heidelberg, 10.06.2008

# Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

VI("Visual Interface"):  
der Weg von der Schreibmaschine zum Kulteditor  
VIM = VI IMPROVED --- die Verbesserung

VIM Tutorial:  
Grundlagen über das Arbeiten mit VIM

VIMRC:  
Tune your own VIM. Starte Vim mit  
maßgeschneiderten Einstellungen

Effektives Editieren von Texten:  
Unabhängig von dem Editor: Tipps für schnelles  
Editieren von Texten in Anlehnung an Bram  
Moolenaar: 7 habits of effectiv text editing

<http://www.moolenaar.net/habits.html>

[http://video.google.de/videoplay?  
docid=2538831956647446078](http://video.google.de/videoplay?docid=2538831956647446078)

# Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

# Visual Interface: Historisches

- VI wurde von Billy Joy entwickelt
- Löst 1976 "ed" als Standard-Editor für Linux ab
- 1984 steigt der Editor Emacs auf, es entstehen zwei Lager:  
VI- und Emacs-Fans



Visual Interface

Proseminar Linux für Umsteiger: VI/VIM

(4/28)

Vor 1976: ein Zeileneditor namens "ed" war der gebräuchlichste Editor unter Linux  
ab 1984 liefern sich VI und Emacs-Anhänger "flamewars";-)

zu Billy Joy:

[http://www.theregister.co.uk/2003/09/11/bill\\_joys\\_gr](http://www.theregister.co.uk/2003/09/11/bill_joys_gr)

Bild von:

[http://de.wikipedia.org/w/index.php?title=Bild:Bill\\_Joy\\_](http://de.wikipedia.org/w/index.php?title=Bild:Bill_Joy_)

Kurze Demo von ed in shell, wirklich lustiger Editor.

```
~/ed
```

```
a
```

```
erste Datei mit ed.
```

```
und mit damit musste man früher schreiben??;-)
```

```
.
```

```
w erstesEd.txt
```

```
q
```

```
~/cat erstesEd.txt
```

## Visual Interface: Zitate



Visual Interface

Proseminar Linux für Umsteiger: VI/VIM

(5/28)

Bild bezieht sich auf die Flamewars zwischen VI und Emacs-Anhänger:

Quelle:

<http://www.io.com/~dierdorf/emacsvi.html>

Zitate von:<http://de.wikipedia.org/wiki/Vi>

„Sure vi is user-friendly; it's just peculiar about who it makes friends with.“

„Vi - It's small but smart.“

– Chris Snowwhite in der Zeitschrift "EasyLinux" nach dem Sieg des dig.biz Award Austria 2007

# Visual Interface: Modi

Vi <Dateiname>

startet

| Command-line mode                                                | Normal mode                                            | Insert mode              |
|------------------------------------------------------------------|--------------------------------------------------------|--------------------------|
| Beenden (:q)<br>Eingabe und Ausführung von<br>komplexen Befehlen | Tasten haben Funktionen<br>z.b. y = yank<br>d = delete | Zum normalen Text tippen |

<http://de.wikipedia.org/wiki/Vi>

Wegen der verschiedenen Eingabemodi unterscheidet sich VI von den meisten anderen Editoren zum Beispiel das modi-losen Notepad

# Visual Interface: Vor- und Nachteile

## Vorteile

- Schnelles Arbeiten durch kurze Befehle
- Wenig Gebrauch von <strg> oder <alt> durch verschiedene Modi
- Sehr klein & flexibel
- Weit verbreitet

## Nachteile

- Viel zu Lernen für Neueinsteiger durch Modi-Konzept

# Visual Interface: Klone

- Vim
  - Tolle Features
  - Longtermsupport
- Elvis
  - Keine neuen Features für 2 Jahre, nur super Sourcecode --> interessiert User nicht --> das Aus für Elvis
- Nvi
  - Schnell, als erster Klon tolle Features wie Multibuffers usw
  - Aber: keine neuen Versionen mehr

<http://thomer.com/vi/vi.html#versions>

Vorteile von Vim gegenüber Gvim und Emacs:

sehr kompakt: ~1mb Speicher

gvim: ~4mb

emacs: ~10mb

Wie macht vim das?

Es nutzt mitunter externe Kommandos z.b. `:!sort`

# Gliederung

- Visual Interface
- [Vim Tutorial](#)
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

# Vim Tutorial

- Survival Befehle: starten, beenden, Hilfe

| Befehl          | VIM   | Emacs             |
|-----------------|-------|-------------------|
| Starte in shell | vim   | emacs -nw         |
| Starte als gui  | gvim  | emacs             |
| beenden         | :q    | <strg>-x <strg>-c |
| Hilfe           | :help | <strg>-h i        |

Starte in shell: vim grundlagen.txt

Die Tabelle dient zur Verdeutlichung, dass VI-Befehle weniger Tasten brauchen, als Emacs-Befehle  
Ausführliche Tabelle unter  
<http://www.io.com/~dierdorf/emacsvi.html>

Blindtext:

[http://de.wikipedia.org/wiki/Lorem\\_ipsum](http://de.wikipedia.org/wiki/Lorem_ipsum)

# Tutorial Zusammenfassung

- „vim“ startet Vim in der shell
- :q <--schließt vim
- :help [<stichwort>]

# Tutorial Zusammenfassung

- „vim“ startet Vim in der shell
- :q <--schließt vim
- :help [<stichwort>]
- Emacs-Fans: `rm -r /bin/vi /`

# Gliederung

- Visual Interface
- Vim Tutorial
- **VIMRC**
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

## VIMRC für Version 7

- Ist individuell für jeden Nutzer (~/.vimrc)
- Startet die gewünschte Konfiguration automatisch
- Editieren über :e \$MYVIMRC
- Erlaubt sind alle command-line Befehle

<http://www.vi-improved.org/vimrc.php>

# VIMRC - Beispiel

```
set nocompatible "Nutzt VIM mit allen erweiterten Funktionalitäten. Es gibt
"eigentlich keinen Grund, dieses Kommando nicht zu
"verwenden. Weil VIM ist echt besser als VI;-)
set noswapfile "soll verhindern, dass Fehler bei der Präsentation
"durch viele Zugriffe auf das gleiche File entstehen
set autoindent
set backspace=indent,eol,start

Speicherrelevantes #####
set history=200 "begrenzt die Anzahl der Undo-Vorgänge auf 200

Arbeiten mit verschiedenen Dateitypen #####
syntax on "Syntaxhighlighting wird angestellt
set filetype plugin indent on "lädt zugehörige Plugins
```

# VIMRC - Beispiel

##### Optisches #####

set background=dark

set ruler

set showcmd

"dunkler Hintergrund

"zeigt rechts unten die Cursorposition an

"zeigt an, welche Tasten für ein

"Kommando schon gedrückt wurden

##### Suche #####

set incsearch

"fängt schon bei der Eingabe des ersten

"Buchstaben an nach passenden Mustern zu

"suchen:

"/a markiert erstmal das nächste a

"/ab sucht zunächst das nächste ab usw

"incsearch = incremental search

set hlsearch

"highlight search

"markiert alle treffenden Muster

# Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- Weitere Informationsmöglichkeiten

# Effective Text Editing

1) Ineffiziente Arbeitsweise erkennen

2) Suche nach besseren Methoden:

- Kollegen, Freunde, Irc-Community befragen
- Suche im Netz nach besseren Möglichkeiten
- Buch kaufen?

3) Neue Methoden zur Gewohnheit machen

- .vimrc verändern
- Vim-Script schreiben

# Effectiv Text Editing

- Vier Beispiele:
  - 1) Lange Wörter mittels Abkürzungen (:abbr)
  - 2) Suchen und Ersetzen von Kommentaren
  - 3) Benutze das Python-Filetype Plugin
  - 4) Nutze Macros für häufige Arbeit

# 1) Abbreviations

- Gehe in vim
- Tippe `:abbr kub kubuntu`
- Gehe in den insert-mode und teste `kub<space>`

## 2) Search and Replace

- In shell: `vim kommentare.txt`
- Dort sind Kommentare durch `//` markiert
- Du willst sie durch `#` ersetzen
- Tippe `:%s/\\/#/g`

## 3) Python

- In shell: vim PythonAndVim.py
- ```
def quadratFunktion(a):  
    a *= a  
    if a < 10:  
        print a  
    else:  
        print 'Quadrat zu groß ;-)  
    return a
```
- `quadratFunktion(3)`

Nachdem das erste Mal `quadratFunktion` geschrieben wurden, kann man im Insert-mode nach diesem Ausdruck suchen. Dafür beginnt man zu schreiben „qu“ und drückt `<strg>-p` und sucht somit im Text rückwärts alle Wörter die mit „qu“ anfangen. `<strg>-n` sucht vorwärts

4) Macros

- 1) Drücke im normal mode qa ,um ein Macro aufzuzeichnen
- 2) Drücke q im normal mode, um die Aufzeichnung zu beenden
- 3) Drücke @a ,um das Macro auszuführen. Alles was zwischen 1) und 2) gesehen ist, wird genauso, als ob Du es eintippen würdest, wiederholt

Gliederung

- Visual Interface
- Vim Tutorial
- VIMRC
- Effektives Editieren von Texten
- [Weitere Informationsmöglichkeiten](#)

Weitere Informationsquellen

- Hilfen:

- `:help <Befehl>`
- Irc: Network: freenode; Common sever: irc.freenode.net;channel: #vim
- Vim mailinglist @ <http://www.vim.org/>
(viele Userfragen und Antworten)

- Tutorials:

- shell: vimtutor
- <http://www.vi-improved.org/tutorial.php> (sehr gut als Einstieg)
- <http://www.selflinux.org/selflinux/html/vim.html>

Weitere Informationsquellen

- Sprachenspezifische Hilfe
 - <http://vim.wikia.com/wiki/Category:LanguageSpecific>
- Effektives Texte Editieren
 - <http://www.moolenaar.net/habits.html> (kurz und gut)
 - <http://video.google.de/videoplay?docid=2538831956647446078>
- Schnell schreiben
 - <http://speedtest.schnell-schreiben.de/>
 - <http://ktouch.sourceforge.net/>

Fazit und Ausblick

- Vim ist sehr gut geeignet, Texte zu editieren
- Anfänger brauchen etwas Einarbeitungszeit

- Ausblick
- Das Zusammenspiel mit anderen Programmen dürfte noch verbessert werden:
 - Als externes Proggi:(KMail--> External Composer: `gvim "+set ft=mail" -f %f`); eine interne Lösung wäre schön

Quellen

- <http://www.vim.org/>
- <http://de.wikipedia.org/wiki/Vi>
- http://de.wikipedia.org/wiki/Billy_Joy
- http://www.theregister.co.uk/2003/09/11/bill_joy_s_greatest_gift/
- <http://thomer.com/vi/vi.html#versions>
- <http://www.moolenaar.net/habits.html>
- <http://video.google.de/videoplay?docid=2538831956647446078>
- <http://www.vi-improved.org/>
- <http://de.youtube.com/watch?v=S76pHIYx3ik>
- <http://wiki.ubuntuusers.de/Vim>