Vim Partywissen für absolute Beginner

TÜBIX

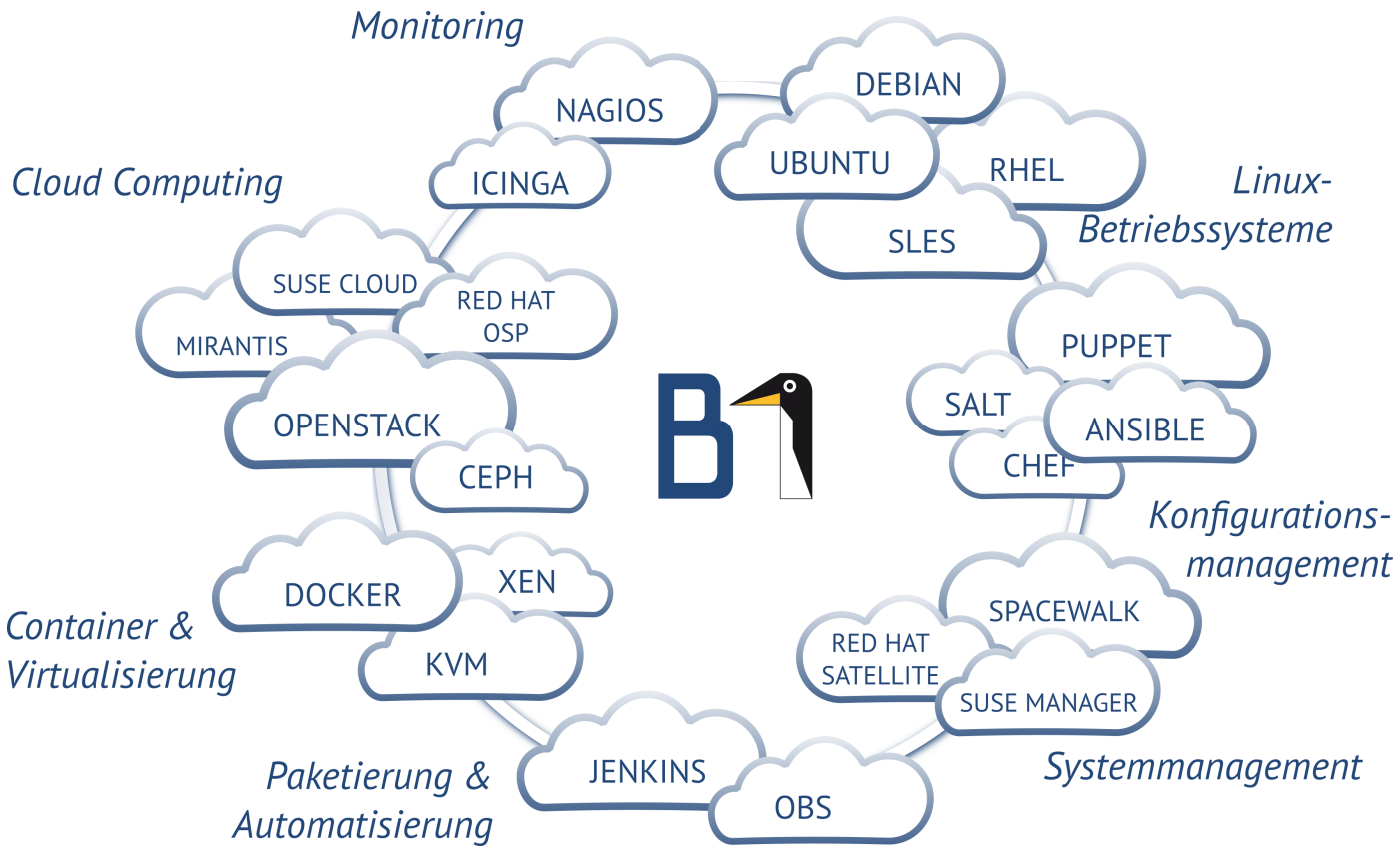
24. Juni 2017

Philipp Kammerer
Media, Design, Development
B1 Systems GmbH
kammerer@b1-systems.de

Vorstellung B1 Systems

- gegründet 2004
- primär Linux/Open Source-Themen
- national & international tätig
- ca. 100 Mitarbeiter
- unabhängig von Soft- und Hardware-Herstellern
- Leistungsangebot:
  - Beratung & Consulting
  - Support
  - Entwicklung
  - Training
  - Betrieb
  - Lösungen
- Büros in Rockolding, Köln, Berlin & Dresden

Schwerpunkte



vim - vi improved

Warum denn vim, wenn es BrandX gibt?

+ auf nahezu allen Linux-Systemen verfügbar
+ für SSH-Sessions oder Konsolen-Junkies (gibt auch eine grafische Variante und einen Windows-Port)
+ schnelle Navigation in der Datei
+ hochgradig konfigurierbar
+ für Entwickler geeignet: Syntaxhervorhebung, Einrückung, Klammer, Code falten...

Kurz: Bedienung mit Tastaturbefehlen, ohne Maus

Warum denn Vim, wenn es BrandX gibt?

- ungewöhnliches Bedienkonzept
- ...

Kurz: Vim lernen ist manchmal etwas holprig

Öffnen, :Bewegen, Bearbeiten:, Speichern, Schließen

Datei öffnen — Übung

```
$ vim /etc/vimrc
```

- Finger auf `j`, `k`, `h`, `l`
- j = runter; k = rauf; h = links; l = rechts

Datei schließen

`:q`	schließt aktuelle Datei, warnt bei ungesicherten Änderungen
`:q!`	schließen und Änderungen verwerfen

Etwas trockene Theorie

Die drei (vier, fünf, sechs) Betriebsmodi von Vim

`<esc>`	Normaler Modus (normal)
`:`	Befehlsmodus (command)
`i`	Einfügemodus (insert)
`v`	Markieren (visual)
`gh`	Markieren und überschreiben (select)
`Q`	Persistenter Befehlsmodus (ex)

Etwas trockene Theorie

Wenn man hängen bleibt

> *However, zie most important is the Escape button*
>
> — Jurgen Guntherswarchzhaffenstrassen

Durch einmaliges (manchmal zweimaliges) Drücken der <esc>-Taste kommt man in den normalen Modus zurück.

Einstellungen, die das Leben erleichtern

~/.vimrc

```
1 set number
2 set ruler
3 syntax on
4 set hlsearch
```

- Datei öffnen : `$ vim ~/.vimrc`
- `O (Shift + oscar)` : Neue Zeile vor aktueller einfügen
- Einfügemodus wird gestartet, es kann losgetippt werden
- `<esc>` : Beende Einfügemodus
- `:wq` : Befehlsmodus starten, Datei speichern, Vim beenden

Grundbewegungen

Bewegen

Übung: Lipsum

http://vimdoc.sourceforge.net/htmldoc/usr_03.html

`j,k,h,l`	Cursor runter, rauf, links, rechts
`0, $`	Zeilenanfang, Zeilenende
`w, e`	Wortanfang, Wortende (Sonderzeichen)
`W, E`	Wortanfang, Wortende (Leerzeichen)
`b, ge`	Wortanfang, Wortende rückwärts (Sonderzeichen)
`B, gE`	Wortanfang, Wortende rückwärts (Leerzeichen)
`<Zahl> gg`	Springe zu Zeile <Zahl>

- Bewegungen können zum Wiederholen mit Zahlen kombiniert werden:
  `5j` geht fünf Zeilen nach unten
  `3e` springt drei Wortenden weiter

Wie man den Einfügemodus startet

Einfügen

`i`	Füge vor Cursorposition ein
`I`	Füge am Zeilenanfang ein
`a`	Füge nach Cursorposition ein
`A`	Füge am Zeilenende ein
`o`	Füge neue Zeile nach aktueller ein
`O`	Füge neue Zeile vor aktueller ein

- Wechsel von Normaler Modus nach Einfügemodus.
- Falls *nicht* der Normale Modus gewählt ist, muss zunächst mit `<esc>` in denselben gewechselt werden.

Bearbeiten und Ändern von bestehendem Text

Bearbyten und Changen

`r`	Replace. Ersetze Zeichen unter Cursor
`R`	Replace. Ersetze bestehenden Text (Überschreiben)
`d`	Delete. Lösche nächste Bewegung
`dd`	Delete Line. Lösche akuelle Zeile
`x`	Lösche Zeichen unter dem Cursor
`X`	Lösche Zeichen links vom Cursor
`c`	Change. Lösche nächste Bewegung und starte Einfügemodus
`C`	Change. Lösche bis Zeilenende und starte Einfügemodus

- vorher ggf. in den Normalen Modus wechseln (<esc>)

Visual Basic - Markierungsmodus

Die drei Visual Modi

http://vimdoc.sourceforge.net/htmldoc/usr_04.html#04.4

`v`	Visual Character. Markiert einzelne Zeichen
`V`	Visual Line. Markiert ganze Zeile
`STRG + v`	Visual Block. Markiert einen Textblock

- Verändern der Markierung mit Bewegungsbefehlen
- Abbrechen mit <esc>.
- Weiterverarbeiten der Markierung mit weiteren Befehlen
- `o`, `O`, um ans andere Ende der Markierung zu springen
- `g STRG + g`, um die Zeichen innerhalb der Markierung zu zählen

Weiterverarbeiten von Markierungen

Visual : `v` — Übung: visual

`r`	Ersetzt die gesamte Markierung durch das nächste Zeichen
`c`	Löscht Markierung und startet Insert
`x`	Löscht Markierung
`y`	Yankt (Kopiert) Markierung in Register
`g STRG + g`	Zählt Zeichen in der Markierung
`o`	Gehe zum anderen Ende der Markierung

- bei schlecht zählbaren Bewegungen (halbe Wörter)
- visuelle Repräsentation vom Bereich, der bearbeitet wird

Weiterverarbeiten von Markierungen

Visual Line : `V` — Übung: visual

`r`	Ersetzt die Markierung durch das nächste Zeichen
`c`	Löscht Markierung und startet Insert
`R, C`	Löscht Zeile bis auf führende Einrückung und startet Insert
`x, X`	Löscht Markierung
`y, Y`	Yankt (Kopiert) Markierung in Register
`g STRG + g`	Zählt Zeichen in der Markierung
`o`	Gehe zum anderen Ende der Markierung

Weiterverarbeiten von Markierungen

Visual Block : `STRG + v` — Übung: visual

`I`	Füge am Anfang für alle markierten Zeilen ein
`A`	Hänge am Ende für alle markierten Zeilen an
`c`	Löscht Markierung und startet Insert (alle Zeilen)
`r`	Ersetzt die Markierung durch das nächste Zeichen
`R`	Löscht Zeile bis auf führende Einrückung und startet Insert
`x, X`	Löscht Markierung
`y, Y`	Yankt (Kopiert) Markierung in Register
`g STRG + g`	Zählt Zeichen in der Markierung
`o, O`	Gehe zum diagonalen, horizontalen Ende der Markierung

- Visual Block Modus nach dem Bearbeiten mit <esc> beenden, damit die Änderungen wirksam werden.

Mehr Bewegungen

Springe zu einem bestimmten Zeichen

Übung: movement.*

`f<Zeichen>`	Springe zum nächsten <Zeichen> nach rechts
`F<Zeichen>`	Springe zum nächsten <Zeichen> nach links
`t<Zeichen>`	Springe vor nächstes <Zeichen> nach rechts
`T<Zeichen>`	Springe vor nächstes <Zeichen> nach links
`%`	Springe zur zugehörigen Klammer

- Kombinierbar mit `c`, `d` und Zahlen zur Wiederholung

Formatierungen

Formatierungen : g

Übung: lipsum

http://vimdoc.sourceforge.net/htmldoc/usr_25.html

`:set textwidth`	Automatischer Zeilenumbruch
`gq<Bewegung>`	Übertrage Zeilenumbruch auf <Bewegung>
`gu<Bewegung>`	Ändere alles in <Bewegung> zu Kleinbuchstaben
`gU<Bewegung>`	Ändere alles in <Bewegung> zu Großbuchstaben
`g~<Bewegung>`	Vertausche Groß-/Kleinschreibung in <Bewegung>

- kombinierbar mit Visual
- Bewegung zum nächsten Absatz: ap
- ~ ändert Groß-/Kleinschreibung vom Zeichen unter dem Cursor

Es ist eine Repetition, Repetition, Repetition...

Undo/Redo

`u`	Rückgängig
`STRG + r`	Wiederholen

Makros

Übung: makros.tex

`q<Taste>`	Beginne und beende die Aufzeichnung
`@<Taste>`	Spiele die Aufzeichnung von <Taste> ab
`.`	Wiederhole den letzten Befehl

- Das Makro wird als Text ins Register <Taste> gespeichert
- `:reg` zeigt die Inhalte aller Register an

Register

Unbenannte Register

http://vimdoc.sourceforge.net/htmldoc/change.html#registers

`d, x, y`	Lösche, yanke Bewegung oder Markierung und ersetze Register
`p`	Inhalt aus unbenanntem Register nach Cursor einfügen
`P`	Inhalt aus unbenanntem Register vor Cursor einfügen

Benannte Register

`:reg`	Liste alle Inhalte aller Register auf
`"<Taste>`	Register <Taste> ansprechen
`q<Taste>`	Beginne und beende die Aufzeichnung
`@<Taste>`	Spiele die Aufzeichnung von <Taste> ab

- Das Makro wird als Text ins Register <Taste> gespeichert.

Spaß mit Registern

Benannte Register

`"my3w`	Yanke 3 Wörter ins Register m
`"mp`	Füge den Inhalt aus Register m rechts vom Cursor ein
`"MfX`	Füge alles bis zum nächsten X an Register m an

- zwischendurch mit `:reg` prüfen, was in den Registern gespeichert ist.
- Makroinhalt ändern: Inhalt vom Register als Text einfügen, bearbeiten und dann wieder in das Register speichern

Search and Destroy

Suchen allgemein

`*`	Suche String unter dem Cursor
`:s`	Substitute starten, sucht nur in der aktuellen Zeile
`:<Range>s`	Zeilenbereich angeben
`:s/foo/bar`	Ersetzt in aktueller Zeile das erste foo durch bar
`:nohl`	Entferne die Hervorhebung

- Wie auch bei sed kann der Delimiter frei gewählt werden.
- vollständiges Kommando: `:<Range>s/foo/bar/<flags>`

Search and Destroy: Range

Range
http://vimdoc.sourceforge.net/htmldoc/usr_10.html#10.3

`:%s`	Alle Zeilen
`:1,5s`	Zeile 1 bis 5
`:.,$s`	Aktuelle Zeile bis Ende

Search and Destroy: Flags

Flags

http://vimdoc.sourceforge.net/htmldoc/change.html#:s_flags

`c`	confirm; jede Ersetzung bestätigen
`g`	global; alle Treffer in einer Zeile
`i`	ignore case; Ignoriere Groß-/Kleinschreibung
`n`	number; Zähle die Anzahl der Matches

Tab-Dance

Tabbing

`:tabe <Datei>`	Öffne Datei in neuem Tab
`gt`	Gehe zum nächsten Tab
`gT`	Gehe zum vorherigen Tab

Split Windows

Splitting

http://vimdoc.sourceforge.net/htmldoc/usr_08.html

`:split , :new <datei>`	Teile Fenster horizontal und öffne Datei
`:vsplit <datei>`	Teile Fenster vertikal und öffne Datei
`STRG + w w`	Gehe ins nächste Fenster
`STRG + w W`	Gehe ins vorherige Fenster
`:close`	Schließe dieses Fenster
`:only`	Schließe alle außer dem aktuellen Fenster
`<zahl>STRG + w _`	Setze Größe von Fenster auf <zahl> Zeilen
`STRG + w +`	Vergrößere Fenster
`STRG + w -`	Verkleinere Fenster

- Mit `screen` und dem `:tabe`-Befehl kann man hier tolle Verschachtelungen konstruieren.

Navigation im Fenster

Navigation im Fenster

`STRG + e`	Runter scrollen (Cursor bleibt in Datei stehen)
`STRG + y`	Rauf scrollen (Cursor bleibt in Datei stehen)
`zz`	Aktuelle Zeile in die Mitte des Fenster schieben
`zt`	Aktuelle Zeile an den oberen Rand schieben
`zb`	Aktuelle Zeile an den unteren Rand schieben

- bewegt das Dokument innerhalb des Fensters
- Die Cursorposition bleibt relativ zur Datei gleich

Arbeiten mit Code

Autocomplete

Übung: movement.php

`STRG + n, p`	Im Einfüge Modus: Keyword Autocomplete

Code Folding

`zf<bewegung>`	Erstelle Fold über <bewegung>
`zo`	Open Fold
`zc`	Close Fold

- Fun Fact: Das `z` soll ein gefaltetes Blatt Papier symbolisieren.

Code Folding

Code Folding...

`zr`	Alle Folds öffnen (reduce)
`zm`	Alle Folds schließen (more)
`zn`	Alle geschachtelten Folds öffnen
`zN`	Alle geschachtelten Folds schließen

Folding in View speichern und laden

`:set viewdir`	Wo werden die Viewinformationen gespeichert
`:mkview <0-9>`	Speichere View Nummer <0-9> für diese Datei
`:loadview <nummer>`	Öffne View <nummer>

Happy Hacking

Vielen Dank für Ihre Aufmerksamkeit!

Bei weiteren Fragen wenden Sie sich bitte an info@b1-systems.de
oder +49 (0)8457 - 931096