

Texteditor Vi

© 2008 Dipl.Phys. Gerald Kempfer
Lehrbeauftragter an der TFH-Berlin

Internet: www.tfh-berlin.de/~kempfer
www.kempfer.de

E-Mail: gerald@kempfer.de

Stand: 07. Oktober 2008

Inhaltsverzeichnis

1. DER TEXTEDITOR VI.....	4
1.1. EINFÜHRUNG	4
1.2. VI STARTEN	4
1.3. DIE VIER VERSCHIEDENEN MODI DES VI.....	5
1.4. TEXTE EINGEBEN	5
1.5. TEXTE BEARBEITEN	5
1.6. TEXTE MARKIEREN, KOPIEREN UND EINFÜGEN	6
1.7. SUCHEN	6
1.8. ZEILEN EIN- UND AUSRÜCKEN.....	7
1.9. TEXTMARKEN	7
1.10. MIT MEHREREN FENSTERN IM VI ARBEITEN	7
1.11. EINSTELLUNGEN FÜR DEN VI	8
2. ÜBERSICHT ÜBER DIE BEFEHLE DES VI.....	9
2.1. STARTEN EINER VI-SITZUNG	9
2.2. TEXTSICHERUNG UND VI BEENDEN.....	9
2.3. TEXTE EINGEBEN	10
2.4. TEXTE LÖSCHEN	10
2.5. ZEILEN VERBINDEN	10
2.6. TEXTE SUCHEN	11
2.7. TEXT MARKIEREN. KOPIEREN UND EINFÜGEN.....	11
2.8. RÜCKGÄNGIG MACHEN UND WIEDERHOLEN VON KOMMANDOS.....	11
2.9. CURSOR BEWEGEN UND BILDSCHIRM AUSRICHTEN	11
2.10. TEXT EINRÜCKEN	13
2.11. TEXTMARKEN SETZEN	13
2.12. MEHRERE BEREICHE	13
2.13. OPTIONEN SETZEN	14
3. EINSTELLUNGEN BEIM STARTEN DES VI MITLADEN	15

Wichtig: Beim Namen der Textdatei ist unter Linux und unter Unix (anders als unter Windows) auf die Groß-/Kleinschreibung zu achten!

Der Vi startet und öffnet gleich die angegebene Textdatei. Existiert die Textdatei nicht, wird eine leere Datei geöffnet.

1.3. Die vier verschiedenen Modi des Vi

Das Konzept des Vi unterscheidet sich deutlich von dem anderer Editoren. Der Vi verwendet verschiedene Modi, in denen jeweils eine bestimmte Funktionalität bereit gestellt wird. Es kann zwischen den folgenden vier Modi unterschieden werden:

- Normalmodus
- Eingabemodus
- Befehlsmodus
- Markierungsmodus

Der jeweils aktuelle Modus sowie weitere Informationen zu dem gerade gewählten Befehl werden unten in der Statuszeile angezeigt.

1.4. Texte eingeben

Nach dem Start des Editors befindet sich dieser im Normalmodus. Um einen Text eingeben zu können, muss erst in den Eingabemodus gewechselt werden. Dies geschieht durch Drücken der Taste `[i]` (wie *insert*). Alternativ kann auch die `[Einf]`-Taste verwendet werden. Die Statuszeile sieht anschließend wie folgt aus:

```
-- EINFÜGEN --                                0,1          Alles
```

Wird nun Text eingegeben, wird dieser vor dem Cursor eingefügt. Neben dem Einfügen gibt es auch das Ersetzen. Im Eingabemodus kann zwischen beiden mit der `[Einf]`-Taste gewechselt werden; in der Statuszeile wird jeweils angegeben, ob der Text eingefügt oder ersetzt wird.

Vom Normalmodus kann auch direkt in das Ersetzen vom Eingabemodus gewechselt werden. Hierzu muss die Taste `[R]` (wie *replace*; wichtig: Großes R!) gedrückt werden.

```
-- ERSETZEN --                                0,1          Alles
```

Der Eingabemodus wird durch Drücken der `[ESC]`-Taste wiederverlassen. Das Wort Einfügen bzw. Ersetzen in der Statuszeile verschwindet und Sie befinden sich wieder im Normalmodus.

1.5. Texte bearbeiten

Im Eingabemodus können die zuletzt eingegebenen Zeichen mit der Rückschritttaste `[←]` (Backspace) wieder gelöscht werden (löscht das Zeichen links vom Cursor). Haben Sie die Zeile mit dem Cursor verlassen oder den Eingabemodus beendet, können Sie mit der Rückschritttaste keine Zeichen mehr löschen.

Dagegen können Sie mit der `[Entf]`-Taste jederzeit (im Eingabe- wie auch im Normalmodus) Zeichen löschen (löscht das Zeichen, auf dem der Cursor steht).

Eine Besonderheit stellen die Zeilenumbrüche dar; diese können nämlich nicht direkt gelöscht werden. Ein Zeilenumbruch kann nur gelöscht werden, indem die nächste Zeile an die aktuelle angehängt wird. Dazu drücken Sie im Normalmodus die Taste `[J]` (es wird ein Leerzeichen zwischen beiden Zeilen eingefügt; wichtig: Großes J!) oder die Tasten `[g][J]` (es wird kein Leerzeichen zwischen beiden Zeilen eingefügt). Dabei ist es unwichtig, an welcher Stelle der Cursor in der Zeile steht; er springt dabei automatisch an das Ende der aktuellen Zeile.

Im Normalmodus können Sie durch die Eingabe von `[d][d]` die aktuelle Zeile löschen. Geben Sie eine Zahl vor dem `[d][d]` ein und es wird die angegebene Anzahl von Zeilen gelöscht. Z.B. löscht die Eingabe `[5][d][d]` 5 Zeilen (die aktuelle sowie die nächsten 4 Zeilen).

Um Eingaben oder Befehle rückgängig zu machen, drücken Sie im Normalmodus die Taste `[u]`. Diese Taste können Sie auch mehrmals hintereinander drücken, um mehrere Eingaben und/oder Befehle rückgängig zu machen; alternativ dazu können Sie auch eine Zahl vor dem `[u]` eingeben, um die angegebene Anzahl von Eingaben und/oder Befehlen rückgängig zu machen. Möchten Sie alle Änderungen der zuletzt geänderten Zeile rückgängig machen, drücken Sie die Taste `[U]` (Wichtig: Großes U!).

Mit `[Strg]+[R]` lässt sich der letzte, rückgängig gemachte Befehl bzw. die letzte, rückgängig gemachte Eingabe wiederherstellen (sozusagen wird das Rückgängig-Machen rückgängig gemacht).

1.6. *Texte markieren, kopieren und einfügen*

Um Texte markieren zu können, muss erst vom Normalmodus in den Markierungsmodus (auch Visueller Modus) gewechselt werden. Im Markierungsmodus wird zwischen Zeichen-, Zeilen- und Blockmarkierung unterschieden. Bei der Zeichenmarkierung wird die Markierung mit den Cursortasten bzw. den Tasten zum Bewegen des Cursors (siehe Befehlsübersicht im nächsten Kapitel) zeichenweise erweitert bzw. reduziert; bei der Zeilenmarkierung werden immer ganze Zeilen markiert und bei der Blockmarkierung wird ein rechteckiger Block markiert.

Um in den Markierungsmodus zu wechseln, drücken Sie im Normalmodus die Taste `[v]` (Zeichenmarkierung), `[V]` (Zeilenmarkierung) oder die Tastenkombination `[Strg]+[V]` (Blockmarkierung). Es kann jederzeit mit diesen Tasten zwischen den verschiedenen Markierungsmodi gewechselt werden. In der Statuszeile wird jeweils der entsprechende Markierungsmodus angezeigt, z.B. für Zeichenmarkierung:

```
-- VISUELL --
```

```
7,24
```

```
Alles
```

Nun kann die Markierung durch Bewegen des Cursors erweitert bzw. reduziert werden. Dabei gibt es im Markierungsmodus noch einige spezielle Befehle: Mit dem Befehl `[a][w]` wird das ganze Wort, auf dem der Cursor steht, markiert. Mit `[a][s]` wird entsprechend der ganze Satz und mit `[a][p]` der ganze Absatz markiert. Das Ende eines Absatzes wird anhand einer Leerzeile erkannt. Mit dem Befehl `[a][b]` wird der ganze Bereich um den Cursor markiert, der in runden Klammern steht, und mit dem Befehl `[a][B]` entsprechend der ganze Bereich, der in geschweiften Klammern steht.

Mit dem Befehl `[g][v]` kann der zuletzt markierte Bereich nochmals markiert werden. Dieser Befehl kann auch im Normalmodus zum Wechseln in den Markierungsmodus verwendet werden.

Der markierte Bereich kann mit dem Befehl `[y][y]` oder `[Y]` in einen Puffer kopiert und mit `[x]` oder `[d]` in einen Puffer ausgeschnitten werden. Dabei bedeutet Ausschneiden, dass der markierte Bereich in den Puffer verschoben wird. Nach dem Kopieren bzw. Ausschneiden wechselt der Vi automatisch wieder in den Normalmodus.

Nun wird der Cursor an die Stelle verschoben, an der der Text eingefügt werden soll. Mit `[p]` wird der Inhalt des Puffers rechts vom Cursor und mit `[P]` links vom Cursor eingefügt. Der Inhalt vom Puffer kann beliebig oft eingefügt werden (vergleichbar mit der Zwischenablage unter Windows).

1.7. *Suchen*

Zum Suchen im Text nach einer Zeichenkette wird der Befehl `/Zeichenkette[↵]` verwendet. Damit wird von der Cursorposition aus das (in Richtung Textende) nächste Vorkommen der Zeichenkette gesucht und angezeigt. Um von der Cursorposition aus in Richtung Textanfang zu suchen, verwenden Sie den Befehl `?Zeichenkette[↵]`.

Um nach weiteren Vorkommen der Zeichenkette zu suchen (entsprechend der vorher verwendeten Suchrichtung), geben Sie den Befehl `[n]` ein. Mit dem Befehl `[N]` kommen Sie dagegen zum vorigen Vorkommen der gesuchten Zeichenkette (also eine Suche entgegen der vorher verwendeten Suchrichtung).

1.8. Zeilen ein- und ausrücken

Gerade bei Quelltexten ist es wichtig, dass die Programmstrukturen durch Einrückungen der einzelnen Zeilen deutlich gemacht werden. Diese Einrückungen sind führende Leerzeichen oder Tabulatoren. Jeder Einrückungsschritt besteht aus 3 Leerzeichen oder einem Tabulator, der die Breite von 3 Leerzeichen hat.

Für das Einrücken von Quelltexten gibt es im Vi verschiedene Befehle: Zum Einen gibt es Einstellungen für die Breite der Einrückung sowie für die Verwendung des Tabulators (Einstellungen siehe Abschnitt weiter unten) und zum Anderen gibt es Befehle, mit denen die Einrückung für eine oder mehrere Zeilen nachträglich verändert werden kann.

Mit dem Befehl `<<` wird die aktuelle Zeile um eine Einrückung nach links gerückt (ausgerückt), sofern diese Zeile eingerückt ist. Entsprechend wird die aktuelle Zeile mit `>>` nach rechts eingerückt. Um mehrere Zeilen ein- oder auszurücken, muss vor dem entsprechenden Befehl die Anzahl der Zeilen angegeben werden, z.B. `5 >>` rückt die aktuelle und die vier nachfolgenden Zeilen um eine Ebene ein.

1.9. Textmarken

Mittels Textmarken können beliebige Positionen im Text gemerkt und später wieder direkt angesprungen werden. Eine Textmarke wird mit dem Befehl `m c` gesetzt, wobei *c* ein beliebiger Buchstaben von a bis z oder A bis Z sein kann. Dabei können Textmarken mit einem Großbuchstaben auch Datei-übergreifend aufgerufen werden, während sich Textmarken mit einem Kleinbuchstaben nur auf die aktuelle Datei beziehen.

Um zu einer Textposition mit Hilfe einer Textmarke zu springen, wird der Befehl `' c` verwendet. Dabei ist *c* der Buchstabe der gewünschten Textmarke.

Mit dem Befehl `:marks` erhalten Sie eine Liste von allen Textmarken. Dabei werden Sie noch weitere Textmarken entdecken, die vom Vi selber erzeugt und verwaltet werden.

1.10. Mit mehreren Fenstern im Vi arbeiten

Natürlich können unter einer grafischen Oberfläche wie KDE, Gnome usw. auch mehrere Terminalfenster geöffnet werden, in denen jeweils der Vi gestartet wird. Aber auch im Vi selber kann der Bildschirm aufgeteilt und in den jeweiligen Bereichen verschiedene Textdateien angezeigt werden. Dazu wird der Befehl `:split` verwendet. Hiermit wird der Bildschirm waagrecht geteilt und in beiden Teilen wird nun die Textdatei angezeigt. Wird dagegen der Befehl `:new` eingegeben, wird auch der Bildschirm waagrecht geteilt, aber in dem oberen Bereich wird eine leere Textdatei angezeigt. Der Bildschirm sieht dann z.B. folgendermaßen aus:

```
Hier wurde ein neuer Text eingegeben.
```

```
~
~
~
~
~
~
~
```

```
[Unbekannt] [+] 1,38 Alles
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    int i = 0, j = 0;

    for (i = 0; i < 10; i++)
test.c 1,1 Anfang
-- EINFÜGEN --
```

Das Aufteilen des Bildschirms kann so oft gemacht werden, bis in jedem Bereich nur noch eine Zeile vom Text angezeigt werden kann.

Alle Bereiche sind nun unabhängig voneinander. D.h. in einem Bereich kann auch eine andere Datei eingelesen werden, ohne dass dies Auswirkungen auf die anderen Bereiche hat.

Jeder Bereich hat eine eigene Statuszeile, in der der Dateiname und die Cursorposition angegeben wird. Wird eine Datei verändert, wird in der Statuszeile ein [+] angezeigt, um deutlich zu machen, dass diese Datei noch nicht gespeichert wurde.

Die Statuszeile des Bereiches, der gerade aktuell ist, wird fett gedruckt angezeigt. Mit dem Befehl `Strg + W j` können Sie zum nächsten (darunter liegenden) Bereich und mit dem Befehl `Strg + W k` Bereich wechseln. Dabei wird aber nicht vom letzten zum ersten Bereich gesprungen bzw. umgekehrt. Dies kann dafür der Befehl `Strg + W w` bzw. `Strg + W Strg + W`. Auch mit diesem wird immer zum nächsten Bereich gewechselt, aber beim letzten Bereich wird bei Aufruf dieses Befehls zum ersten Bereich gewechselt.

Um das Aufteilen zu beenden, wird der Befehl `:only` (oder kurz `:on`) verwendet. Der aktuelle Bereich bleibt als einziger übrig, alle anderen Bereiche werden geschlossen. Ausnahme: Die Bereiche, die noch nicht gespeichert wurden, bleiben auch übrig.

1.11. Einstellungen für den Vi

Es gibt eine ganze Reihe von Einstellungen für den Vi. Auch hier sind es Befehle, die eingetippt werden müssen. Jeder (bzw. fast jeder) dieser Befehle fängt mit einem Doppelpunkt und dem Wort `set` an. Dahinter kommt ein Leerzeichen und die gewünschte Option. Soll die Option ausgeschaltet werden, muss vor der Option das Wort `no` vorgesetzt werden (ohne Leerzeichen). So schaltet beispielsweise der Befehl

```
:set number
```

die Zeilennummerierung ein. Um diese wieder auszuschalten, muss der Befehl


```
:set nonumber
```

einggegeben werden.

Zu (fast) jedem Befehl gibt es auch eine Abkürzung. Eine Liste der wichtigsten Einstellungen für den Anfang finden Sie im letzten Abschnitt des zweiten Kapitels.

2. Übersicht über die Befehle des Vi

Diese Übersicht besitzt nicht den Anspruch auf Vollständigkeit. Sofern nicht anders angegeben sind alle Befehle im Normalmodus einzugeben.

Befehle, die mit einem Doppelpunkt, ein Schrägstrich (Slash) oder Fragezeichen beginnen, werden mit der Eingabetaste  gestartet. Die anderen Befehle starten gleich nach der Eingabe des Buchstabens bzw. der Buchstabenfolge.

Für kursive Buchstaben oder Texte müssen die entsprechend der Beschreibung gewünschten Zahlen oder Texte eingesetzt werden.

2.1. *Starten einer Vi-Sitzung*

Die folgenden Befehle werden auf Konsolen-Ebene (Terminal) aufgerufen.

<code>vi</code>	Vi starten ohne Angabe einer Textdatei.
<code>vi datei</code>	Vi starten und <i>datei</i> in den Editor laden.
<code>vi -r datei</code>	Vi starten und die zuletzt gesicherte Version der Datei <i>datei</i> nach einem System- oder Editor-Crash laden.
<code>vi -R datei</code>	Vi starten und <i>datei</i> schreibgeschützt in den Editor laden. Um diese Datei zu speichern, muss ihr entweder ein anderer Name gegeben werden oder sie muss mit <code>:w!</code> gespeichert werden.
<code>vi +n datei</code>	Vi starten, <i>datei</i> in den Editor laden und den Cursor in der Zeile <i>n</i> plazieren.
<code>vi + datei</code>	Vi starten, <i>datei</i> in den Editor laden und den Cursor in der letzten Zeile plazieren.
<code>vi +/str datei</code>	Vi starten, <i>datei</i> in den Editor laden und den Cursor in der ersten Zeile plazieren, die den Text <i>str</i> enthält.

2.2. *Textsicherung und Vi beenden*

<code>:wq</code> oder <code>ZZ</code> oder <code>:x</code>	Sichert den Text und beendet den Vi.
<code>:w</code>	Sichert den aktuellen Text, ohne den Vi zu beenden.
<code>:w datei</code>	Sichert den Text in <i>datei</i> , ohne den Vi zu beenden. Dabei wird davon ausgegangen, dass die Datei noch nicht existiert.
<code>:w! datei</code>	Sichert den Text in <i>datei</i> , ohne den Vi zu beenden. Existiert die Datei bereits, wird sie überschrieben. Dies gilt auch, wenn der Text schreibgeschützt geöffnet wurde.
<code>:n,mw datei</code>	Sichert die Zeilen <i>n</i> bis <i>m</i> in <i>datei</i> , ohne den Vi zu beenden.
<code>:q</code>	Beendet den Vi, sofern der aktuelle Text gesichert wurde.
<code>:q!</code>	Beendet den Vi, auch wenn der aktuelle Text noch nicht gesichert wurde. Die Änderungen am aktuellen Text gehen dann verloren.
<code>:e datei</code>	Schließt den aktuelle Text (sofern dieser gesichert wurde) und öffnet die angegebene Datei bzw. erstellt eine neue Datei mit dem angegebenen Namen.
<code>:e! datei</code>	Schließt den aktuelle Text (auch wenn dieser noch nicht gesichert wurde; er geht dabei verloren) und öffnet die angegebene Datei bzw. erstellt eine neue Datei mit dem angegebenen Namen.

2.3. *Texte eingeben*

Die folgenden Befehle wechseln vom Normalmodus in den Eingabemodus. Um vom Eingabemodus wieder in den Normalmodus zurückzuwechseln, muss die `[ESC]`-Taste gedrückt werden.

a	Wechselt in den Eingabemodus (Einfügen) und setzt den Cursor ein Zeichen nach rechts. Es wird also nach der aktuellen Cursorposition eingefügt.
A	Wechselt in den Eingabemodus (Einfügen) und setzt den Cursor an das Ende der aktuellen Zeile.
i	Wechselt in den Eingabemodus (Einfügen). Der Cursor wird dabei nicht verschoben; es wird also vor der aktuellen Cursorposition eingefügt.
I	Wechselt in den Eingabemodus (Einfügen) und setzt den Cursor an den Anfang der aktuellen Zeile.
o	Wechselt in den Eingabemodus (Einfügen), fügt eine neue Zeile nach der aktuellen Zeile ein und setzt den Cursor in diese neue Zeile.
O	Wechselt in den Eingabemodus (Einfügen), fügt eine neue Zeile vor der aktuellen Zeile ein und setzt den Cursor in diese neue Zeile.
R	Wechselt in den Eingabemodus (Ersetzen). Der Cursor wird dabei nicht verschoben; es wird also das aktuelle Zeichen ersetzt.

2.4. *Texte löschen*

dd oder :d	Löscht die aktuelle Zeile.
ndd :nd	Löscht n Zeilen (die aktuelle und $n-1$ weitere Zeilen). Löscht Zeile n . Der Cursor steht anschließend in der nachfolgenden Zeile.
D oder d\$	Löscht den Text in der aktuellen Zeile vom Cursor bis zum Ende der Zeile.
x oder <code>[Entf]</code>	Löscht das aktuelle Zeichen.
nx oder n <code>[Entf]</code>	Löscht n Zeichen (das aktuelle und $n-1$ weitere Zeichen rechts vom Cursor).
X	Löscht das Zeichen links vom Cursor.
nX	Löscht n Zeichen links vom Cursor; das aktuelle Zeichen wird nicht gelöscht.

2.5. *Zeilen verbinden*

J	Hängt die nächste Zeile an das Ende der aktuellen Zeile mit einem Leerzeichen dazwischen.
gJ	Hängt die nächste Zeile an das Ende der aktuellen Zeile ohne einem Leerzeichen dazwischen.
nJ	Hängt die nächsten $n-1$ Zeilen an das Ende der aktuellen Zeile mit jeweils einem Leerzeichen dazwischen.
ngJ	Hängt die nächsten $n-1$ Zeilen an das Ende der aktuellen Zeile ohne Leerzeichen dazwischen.

2.6. *Texte suchen*

<code>/str</code>	Sucht im Text nach dem Vorkommen von <i>str</i> ab der Cursorposition in Richtung Textende (Vorwärtssuche).
<code>?str</code>	Sucht im Text nach dem Vorkommen von <i>str</i> ab der Cursorposition in Richtung Textanfang (Rückwärtssuche).
<code>/</code>	Wiederholt die letzte Textsuche in Richtung Textende.
<code>?</code>	Wiederholt die letzte Textsuche in Richtung Textanfang.
<code>n</code>	Wiederholt die letzte Textsuche (<code>/</code> oder <code>?</code>).
<code>N</code>	Wiederholt die letzte Textsuche (<code>/</code> oder <code>?</code>) in entgegengesetzter Richtung.

2.7. *Text markieren. kopieren und einfügen*

<code>v</code>	Wechselt zwischen Normalmodus und Markierungsmodus (Zeichenmarkierung).
<code>V</code>	Wechselt zwischen Normalmodus und Markierungsmodus (Zeilenmarkierung).
<code>[Strg]+V</code>	Wechselt zwischen Normalmodus und Markierungsmodus (Blockmarkierung).
<code>gv</code>	Wechselt in den Markierungsmodus (Zeichenmarkierung) und markiert den zuletzt markierten Befehl als Vorgabe. Die Markierung kann dann erweitert bzw. reduziert werden.

Die folgenden Befehle beziehen sich auf den Markierungsmodus.

<code>aw</code>	Markiert das ganze Wort, auf dem der Cursor steht.
<code>as</code>	Markiert den ganzen Satz, in dem der Cursor steht.
<code>ap</code>	Markiert den ganzen Absatz, in dem der Cursor steht. Ein Absatz endet mit einer Leerzeile.





















Der Markierungsmodus kann auch mit der `[ESC]`-Taste beendet werden.

2.8. *Rückgängig machen und Wiederholen von Kommandos*

<code>u</code>	Macht den letzten Befehl oder die letzte Eingabe rückgängig.
<code>nu</code>	Macht die letzten <i>n</i> Befehle oder Eingaben rückgängig.
<code>U</code>	Bringt die zuletzt geänderte Zeile wieder in den Originalzustand.
<code>[Strg]+R</code>	Stellt den letzten, rückgängig gemachten Befehl oder die letzte, rückgängig gemachte Eingabe wieder her.
<code>n[Strg]+R</code>	Stellt die <i>n</i> letzten, rückgängig gemachten Befehle oder die <i>n</i> letzten, rückgängig gemachten Eingaben wieder her.
<code>.</code>	Wiederholt den letzten Befehl.
<code>n.</code>	Wiederholt den letzten Befehl <i>n</i> mal.

2.9. *Cursor bewegen und Bildschirm ausrichten*

Die Cursortasten funktionieren nur beim Vim, beim ursprünglichen Vi stehen diese Tasten nicht zur Verfügung.

h oder 	Cursor um ein Zeichen nach links bewegen.
nh oder n 	Cursor um <i>n</i> Zeichen nach links bewegen.
l oder 	Cursor um ein Zeichen nach rechts bewegen.
nl oder n 	Cursor um <i>n</i> Zeichen nach rechts bewegen.
k oder 	Cursor um eine Zeile nach oben bewegen.
nk oder n 	Cursor um <i>n</i> Zeilen nach oben bewegen.
-	Cursor um eine Zeile nach oben und dann auf das erste Zeichen der Zeile bewegen.
n-	Cursor um <i>n</i> Zeilen nach oben und dann auf das erste Zeichen der Zeile bewegen.
j oder 	Cursor um eine Zeile nach unten bewegen.
nj oder n 	Cursor um <i>n</i> Zeilen nach unten bewegen.
+	Cursor um eine Zeile nach unten und dann auf das erste Zeichen der Zeile bewegen.
n+	Cursor um <i>n</i> Zeilen nach unten und dann auf das erste Zeichen der Zeile bewegen.
b	Cursor um ein Wort (erkannt durch Leerzeichen, Satzzeichen, Klammern usw.) nach links bewegen.
B oder  + 	Cursor um ein Wort (erkannt nur durch Leerzeichen) nach links bewegen.
nb	Cursor um <i>n</i> Wörter (erkannt durch Leerzeichen, Satzzeichen, Klammern usw.) nach links bewegen.
nB oder n  + 	Cursor um <i>n</i> Wörter (erkannt nur durch Leerzeichen) nach links bewegen.
w	Cursor um ein Wort (erkannt durch Leerzeichen, Satzzeichen, Klammern usw.) nach rechts bewegen.
W oder  + 	Cursor um ein Wort (erkannt nur durch Leerzeichen) nach rechts bewegen.
nw	Cursor um <i>n</i> Wörter (erkannt durch Leerzeichen, Satzzeichen, Klammern usw.) nach rechts bewegen.
nW oder n  + 	Cursor um <i>n</i> Wörter (erkannt nur durch Leerzeichen) nach rechts bewegen.
0 oder 	Cursor zum Zeilenanfang bewegen.
\$ oder 	Cursor zum Zeilenende bewegen.
gg oder  + 	Cursor an den Textanfang bewegen.
ngg oder	Cursor in die <i>n</i> -te Zeile des Textes und dort auf das erste Zeichen bewegen.

<code>n</code> <code>Strg</code> + <code>Pos1</code>	
<code>Strg</code> + <code>Ende</code>	Cursor an das Textende bewegen.
<code>n</code> <code>Strg</code> + <code>Ende</code>	Cursor in die n -te Zeil des Textes von oben und dort auf das letzte Zeichen bewegen.
<code>G</code>	Cursor in die letzte Zeile und dann auf das erste Zeichen der Zeile bewegen.
<code>H</code>	Cursor in die oberste Bildschirmzeile bewegen.
<code>nH</code>	Cursor in die n -te Bildschirmzeile von oben bewegen.
<code>M</code>	Cursor in die Mitte des Bildschirms bewegen.
<code>L</code>	Cursor in die unterste Bildschirmzeile bewegen.
<code>nL</code>	Cursor in die n -te Bildschirmzeile von unten bewegen.

2.10. Text einrücken

<code><<</code>	Rückt die aktuelle Zeile um eine Ebene nach links.
<code>n<<</code>	Rückt die aktuelle und die weiteren $n-1$ Zeilen um eine Ebene nach links.
<code>>></code>	Rückt die aktuelle Zeile um eine Ebene nach rechts.
<code>n>></code>	Rückt die aktuelle und die weiteren $n-1$ Zeilen um eine Ebene nach rechts.

2.11. Textmarken setzen

<code>mc</code>	Textmarke c (c ist ein Buchstabe; a bis z oder A bis Z) an aktueller Position setzen.
<code>'c</code>	Zur Position der Textmarke c springen. Bei Textmarken mit Kleinbuchstaben nur innerhalb der aktuellen Datei, bei Textmarken mit Großbuchstaben Dateifübergreifend.
<code>:marks</code>	Gibt eine Liste aller Textmarken aus.

2.12. Mehrere Bereiche

<code>Strg</code> + <code>W</code> <code>s</code> oder <code>:split</code>	Teilt das Fenster waagrecht in zwei Bereiche auf. Im neuen Bereich wird die gleiche Textdatei angezeigt.
<code>Strg</code> + <code>W</code> <code>n</code> oder <code>:new</code>	Teilt das Fenster waagrecht in zwei Bereiche auf. Im neuen Bereich wird eine neue Textdatei angezeigt.
<code>Strg</code> + <code>W</code> <code>o</code> oder <code>:only</code> oder <code>:on</code>	Alle Bereiche außer dem aktuellen und allen noch nicht gespeicherten Bereiche werden wieder geschlossen.
<code>Strg</code> + <code>W</code> <code>j</code>	Wechselt in den nächsten (darunter liegenden) Bereich.
<code>Strg</code> + <code>W</code> <code>k</code>	Wechselt in den vorigen (darüber liegenden) Bereich.
<code>Strg</code> + <code>W</code> <code>w</code> oder <code>Strg</code> + <code>W</code> <code>Strg</code> + <code>W</code>	Wechselt reihum in den nächsten (darunter liegenden) Bereich.

2.13. Optionen setzen

`:set all` Hiermit werden alle Optionen angezeigt.

`:set ai` oder `:set autoindent` Aktiviert die automatische Einrückung.

`:set noai` oder `:set noautoindent` Deaktiviert die automatische Einrückung

`:set eb` oder `:set errorbells` Bei Fehlermeldungen piept der Computer zusätzlich einmal.

`:set noeb` oder `:set noerrorbells` Bei Fehlermeldungen piept der Computer nicht.

`:set ic` oder `:set ignorecase` Bei der Suche wird die Groß-/Kleinschreibung ignoriert.

`:set noic` oder `:set noignorecase` Bei der Suche wird die Groß-/Kleinschreibung berücksichtigt.

`:set nu` oder `:set number` Numeriert die Zeilen auf dem Bildschirm.

`:set nonu` oder `:set nonumber` Schaltet die Zeilennumerierung ab.

`:set list` Zeigt alle Tabulatoren als `^I` und alle Zeilenumbrüche als `$` an.

`:set nolist` Tabulatoren und Zeilenumbrüche werden nicht mehr als Zeichen angezeigt.

`:set ts=n` oder `:set tabstop=n` Setzt die Schrittweite des Tabulators auf n Zeichen.

`:set et` oder `:set expandtab` Bei der Eingabe des Tabulators wird dieser gleich in die entsprechende Anzahl von Leerzeichen umgewandelt.

`:set noet` oder `:set noexpandtab` Bei der Eingabe des Tabulators wieder dieser nicht in Leerzeichen umgewandelt.

`:set sw=n` oder `:set shiftwidth=n` Setzt die Einrückungsweite für den Tabulator auf n Zeichen.

`:syn on` oder `:syntax on` Schaltet das Syntax-Highlighting ein.

`:syn off` oder `:syntax off` Schaltet das Syntax-Highlighting aus.

3. Einstellungen beim Starten des Vi mitladen

Damit die Einstellungen beim nächsten Start des Vi nicht erneut eingegeben werden müssen, können diese auch in einer Textdatei gespeichert werden. Die Datei heißt `.exrc` und liegt im Homeverzeichnis. Häufig existiert diese Datei bereits, so dass sie nur noch ergänzt werden muss.

Dabei muss der Doppelpunkt bei den `set`-Befehlen weggelassen werden. Kommentare beginnen mit einem Anführungszeichen.

Im folgenden ein Beispiel für eine `.exrc`-Datei:

```
" Zeilen nummerieren
set number

" Syntax farblich hervorheben
syntax on

" automatisch einrücken
set autoindent
set smartindent

" Tabulator und Einrückung 3 Zeichen breit und in Leerzeichen umwandeln
set tabstop=3
set shiftwidth=3
set expandtab

" Koordinatenanzeige aktivieren
set ruler

" Einfügen/Ersetzen/Visuell in Statuszeile anzeigen
set showmode

" passende Klammerpaare anzeigen
set showmatch
```