

Ein `*` in einem der Felder sorgt dafür, dass der Cronjob für jeden Wert in diesem Feld gilt. Hier ein Beispiel, das ein Skript namens *cleanup.sh* jeden Freitag um 18:00 Uhr ausführt:

```
0 18 * * * 5 /usr/bin/skripten/cleanup.sh
```

Das folgende Beispiel ruft zu jeder vollen Stunde das Skript *stuendlich.sh* auf:

```
0 * * * * /usr/bin/skripten/stuendlich.sh
```

Auch Bereiche in der Form Start-Ende sind für ein Feld möglich. Das folgende Beispiel führt montags bis samstags von 8 bis 18 Uhr jeweils stündlich ein Skript namens *arbeit.sh* aus:

```
0 8-18 * * * 1-6 /usr/bin/skripten/arbeit.sh
```

Hinter `*` oder Start-Ende können Sie optional einen `/` und ein Intervall angeben. Das folgende Beispiel führt alle zwei Stunden `who -a` aus, um eine vollständige Liste der angemeldeten Benutzer und virtuellen Terminals zu erzeugen:

```
0 */2 * * * who -a
```

Eventuelle Ausgaben der betreffenden Skripte werden mithilfe des Kommandos `mail` an den betreffenden Benutzer geschickt. Optional fügen Sie als erste Zeile der Crontab eine Anweisung wie diese ein, um die Mails an einen anderen User zu schicken (sinnvoll beispielsweise für die *root*-Crontab):

```
MAILTO=sascha
```

Geben Sie `mail` ein, um solche Nachrichten zu lesen. Die jeweilige Nummer zeigt eine bestimmte Nachricht an; `[Q]` beendet sowohl die einzelne Nachricht als auch das Programm.

Der Befehl `crontab` mit der Option `-e` ermöglicht die Bearbeitung der Crontab im Editor *vi*, der im nächsten Abschnitt vorgestellt wird. `crontab -l` liefert die aktuelle Crontab, und `crontab -r` entfernt sie. Als *root* können Sie auch die Option `-u Username` hinzufügen, um die Crontab eines anderen Benutzers zu lesen oder zu ändern.

7.4 Editoren

Es wurden in diesem Kapitel bereits einige Programme vorgestellt, mit deren Hilfe sich Textdateien manipulieren lassen. Allerdings wurde bisher noch keine Möglichkeit gezeigt, solche Dateien einzugeben oder zu bearbeiten. Die grafischen Oberflächen für Linux sind reichlich mit mausgesteuerten und fensterbasierten Editoren ausgestattet. Dennoch ist es nützlich, den Umgang mit einem klassischen Kommandozeileneditor zu beherrschen.

In diesem Abschnitt werden die entscheidenden Funktionen der beiden wichtigsten Editoren vorgestellt. *vi* (gesprochen: »wie ei«) ist der ursprüngliche Unix-Editor und steht deshalb in jeder beliebigen Unix-Umgebung zur Verfügung. *Emacs* stammt von Richard Stallman und verfügt über eine Reihe komfortabler Fähigkeiten.

7.4.1 vi

Der eigentliche *vi*-Texteditor wurde von *Bill Joy* an der UC Berkeley programmiert. Es gibt eine Reihe mehr oder weniger kompatibler Editoren, die die Funktionalität des ursprünglichen *vi* verbessern. Unter Linux kommt beispielsweise häufig *Vim (ViImproved)* von *Bram Molenaar* zum Einsatz.

Der *vi*-Texteditor wird durch die Eingabe von *vi* beziehungsweise *vi* *Dateiname* gestartet; bei modernen Distributionen ist *vi* in der Regel ein Symlink auf *vim*. Wenn Sie einen Dateinamen angeben, wird diese Datei zum Bearbeiten geöffnet beziehungsweise neu angelegt.

Nachdem *vi* gestartet wurde, befindet er sich im Befehlsmodus. Hier können Sie in der untersten Zeile Befehle eingeben, die in der Regel aus einem oder zwei Buchstaben bestehen. Einige dieser Befehle wechseln in den Editiermodus, in dem Sie Text eingeben oder ändern können, andere dienen dem Verschieben, Löschen oder Kopieren von Text. Mit [Esc] wechseln Sie aus dem Editiermodus wieder zurück in den Befehlsmodus.

Anders als bei den meisten anderen Editoren werden Befehle also nicht durch spezielle Tastenkombinationen mit der [Strg]- oder [Alt]-Taste getätigt, sondern durch normale Tasten, solange Sie sich im Befehlsmodus befinden.

Im Folgenden werden jeweils genau die Tasten angegeben, die Sie auf der Tastatur drücken müssen – beispielsweise bedeutet [A] nicht, dass Sie ein großes A eingeben sollen, sondern dass die Taste [A] gedrückt werden soll. Ein großes A würde dagegen als [⇧] + [A] angegeben. Wenn mehrere Tasten nacheinander gedrückt werden müssen, werden diese einfach der Reihe nach aufgeführt. Lediglich einige ganz eindeutige Sonderzeichen werden so aufgeführt, als seien sie einzelne Tasten – bedeutet beispielsweise, dass Sie [⇧] + [9] drücken müssen.

Bei einigen selteneren Sonderzeichen wird außerdem einmal erklärt, wie Sie sie eingeben können. Da *vi* auch bei macOS (siehe [Kapitel 8](#), »macOS«) mitgeliefert wird und Sie Linux natürlich auch auf dem Macintosh installieren können, finden Sie die Tastenkombinationen für PC und Mac, falls diese sich unterscheiden.

Die wichtigste Taste im Befehlsmodus ist [I] – sie wechselt in den Einfügemodus, in dem Sie Text eingeben können. In diesem Modus können Sie sich grundsätzlich mit den Pfeiltasten durch den Text bewegen. Die Löschstasten funktionieren dagegen je nach Terminal nicht immer. Statt mit [I] gelangen Sie auch mit anderen Tasten in den Eingabemodus, die den Cursor unterschiedlich platzieren und gegebenenfalls Text ersetzen: [A] startet die Eingabe hinter der Cursorposition. [O] fügt unter der Cursorposition zunächst eine neue Zeile ein, während [⇧] + [O] zunächst die aktuelle Zeile nach unten verschiebt. [C] [W] ersetzt das nächste Wort und [C] [C] die aktuelle Zeile. Ersetzen heißt in diesem Fall, dass der jeweilige Text vor der Aktivierung des Eingabemodus entfernt wird.

Im Befehlsmodus können Sie sich mit folgenden Tasten schrittweise durch den Text bewegen: [H] nach links, [J] nach unten, [K] nach oben und [L] nach rechts. Wie Sie sehen, liegen

diese vier Tasten nebeneinander, was ihre Verwendung sehr bequem macht. In den meisten Terminal-Emulationen funktionieren zwar auch die Pfeiltasten, aber allgemein kompatibel sind nur diese vier Buchstabentasten.

Den meisten Befehlen können Sie eine Zahl voranstellen, um sie mehrmals auszuführen. Beispielsweise geht `[3][H]` um drei Schritte nach links und `[4][J]` um vier Zeilen nach unten. `[C][3][W]` entfernt drei Wörter und schaltet dann den Eingabemodus ein.

`[W]` bewegt den Cursor um ein Wort nach rechts, `[B]` um ein Wort nach links. Auch diese Tasten lassen sich mit einer Zahl kombinieren, um mehrere Wörter weiter- beziehungsweise zurückzugehen.

`[0]` (die Null) springt zum Zeilenanfang, `[$]` (`[⇧] + [4]`) zum Zeilenende. Runde Klammern dienen dem Bewegen zwischen Sätzen: Beispielsweise bewegt sich `[⇧] + [8]` zum Anfang des vorangegangenen Satzes, `[⇧] + [9]` zum nächsten. Geschweifte Klammern – `[{]` und `[}]` – springen zum Anfang des nächsten beziehungsweise des letzten Absatzes. Auf einem PC werden sie mithilfe der Tastenkombinationen `[Alt Gr] + [7]` beziehungsweise `[Alt Gr] + [0]` erzeugt, auf dem Mac mithilfe von `[Alt] + [8]` beziehungsweise `[Alt] + [9]`.

Eine Zahl mit einem Pipe-Zeichen `[]` (PC: `[Alt Gr] + [<]`, Mac: `[Alt] + [7]`) dahinter springt zum angegebenen Zeichen in der aktuellen Zeile: `[7>[]` bewegt sich beispielsweise zur Position 7. Ist in einer Zeile die Anzahl der Zeichen geringer als der Wert, den Sie angeben, bewegt sich der Cursor zum letzten Zeichen.

`[-]` bewegt den Cursor zum Anfang der vorigen Zeile, während `[+]` oder `[↵]` zum ersten Nichtleerzeichen in der nächsten Zeile springen.

Besonders angenehm sind die eingebauten Suchfunktionen, die Sie im Befehlsmodus durchführen können: `[/]` leitet einen regulären Ausdruck im `grep`-Format ein. Wenn Sie `[↵]` drücken, wird vorwärts danach gesucht. Der Cursor bewegt sich auf das erste Zeichen der Fundstelle. `[?] Suchmuster [↵]` sucht rückwärts nach dem angegebenen beziehungsweise vorangegangenen Muster. `[N]` sucht nach dem nächsten Vorkommen des aktuellen Suchmusters in der jeweiligen Suchrichtung, `[⇧] + [N]` sucht dagegen in der anderen Richtung.

Beispielsweise sucht `[/][0-9][↵]` vorwärts nach der nächsten Ziffer, `[N]` sucht erneut danach. `[?][a-z][↵]` sucht rückwärts nach dem nächstgelegenen Kleinbuchstaben, ein anschließendes `[N]` sucht wiederum den vorherigen.

Die Befehle zum automatischen Suchen und Ersetzen beginnen mit einem `[:]`, weil es sich um Befehle des Zeileneditors *ex* handelt, auf dem *vi* basiert. Alle *ex*-Befehle werden mit `[↵]` abgeschlossen, deshalb ist es im Folgenden nicht aufgeführt. Mit `[:] S [/]` Suchmuster `[/]` Ersatztext `[/]` ersetzen Sie das Suchmuster in der aktuellen Zeile durch den Ersatztext. Für globales Ersetzen im gesamten Text ist die etwas komplexere Form `[:] % S [/]` Suchmuster `[/]` Ersatztext `[] [G]` zuständig; die Option *G* steht hier für *global*. Verwenden Sie `[C]` als zusätzliche Option, wenn Sie aus Sicherheitsgründen jede einzelne Änderung bestätigen möchten. Anstelle des Prozentzeichens können Sie auch zwei durch Komma getrennte Zei-

lennummern angeben, um das Ersetzen auf den entsprechenden Bereich des Dokuments zu beschränken.

Für das Löschen im Befehlsmodus sind die folgenden Tasten und Tastenfolgen definiert: `[X]` löscht das aktuelle Zeichen, `[X] + [X]` das Zeichen vor dem Cursor. `[D] [W]` löscht das ganze Wort, während `[D] [D]` die gesamte aktuelle Zeile entfernt. Wenn Sie beispielsweise drei Wörter löschen möchten, müssen Sie `[D] [3] [W]` eingeben, für vier Zeilen lautet die Eingabe dagegen `[D] [4] [D]`. Die Tastenkombination `[X] + [D]` löscht den Text bis zum Ende der Zeile. `[P]` oder `[X] + [P]` setzen den zuletzt gelöschten Text hinter beziehungsweise vor dem Cursor wieder ein. Der jeweilige Löschvorgang stellt den gelöschten Text nacheinander in nummerierte Löschpuffer. Mit `[1] [P]` fügen Sie zum Beispiel den Text aus dem Löschpuffer Nummer 1 – dem ältesten – ein. Insgesamt existieren neun nummerierte Löschpuffer.

Wenn Sie Text kopieren möchten, können Sie `[Y]` (Abkürzung für *yank*) und ein Navigationskommando verwenden, beispielsweise `[Y] [W]` für ein Wort oder `[Y] [5] [Y]` für fünf Zeilen. Der Text landet – genau wie beim Löschen – nacheinander in den nummerierten Puffern 1 bis 9 und kann durch die entsprechenden `[P]`-Sequenzen wieder eingefügt werden. Wenn Sie `["]` drücken, dann einen Buchstaben von `[A]` bis `[Z]` und schließlich zweimal `[X] + [Y]`, wird die aktuelle Zeile dagegen in einen von 26 benannten Puffern kopiert. Wird der Name des Puffers mit `[X]` kombiniert (Großbuchstabe), dann wird der Inhalt des Puffers nicht überschrieben, sondern um die neu kopierte Zeile ergänzt. Mit `["]`, dem Buchstaben eines benannten Puffers und `[P]` oder `[X] + [P]` können Sie den Text des entsprechenden Puffers wieder einfügen.

Die Taste `[U]` macht den letzten Lösch-, Einfüge- oder Änderungsbefehl rückgängig. Beachten Sie, dass nur in *Vim* mehrere Schritte rückgängig gemacht werden können; beim klassischen *vi* ist lediglich ein Schritt möglich. `[X] + [U]` stellt die gesamte zuletzt modifizierte Zeile wieder her, und der Punkt `[.]` wiederholt den letzten Editierbefehl beliebig oft.

Wichtig sind schließlich noch die Dateioperationen. Auch bei ihnen handelt es sich um *ex*-Befehle. Geben Sie `:[W] Dateiname` ein, um die aktuell bearbeitete Datei unter einem neuen Namen zu speichern; beachten Sie aber, dass Sie danach mit der alten Datei weiterarbeiten. `:[W] [!]` speichert sie nach erfolgten Änderungen erneut unter diesem Namen – das `[!]` dient jeweils dem Umgehen von Schutzmaßnahmen. `:[R] Dateiname` öffnet die angegebene Datei und fügt ihren Inhalt an der aktuellen Cursorposition ein. `:[Q]` beendet *vi*, aber nur, wenn die aktuelle Datei gespeichert wurde. `:[Q] [!]` erzwingt das Beenden und verwirft Änderungen. `:[X]` ist eine Kurzfassung für `:[W] [Q]`: Beide bedeuten, dass *vi* zuerst speichern und anschließend beendet werden soll.

Diese kurze Übersicht ermöglicht es Ihnen, sich grundsätzlich in *vi* zurechtzufinden und effizient Texte zu bearbeiten. In Wirklichkeit verfügt er über Unmengen weiterer Befehle, immerhin gibt es ganze Bücher über die Arbeit mit diesem Editor. [Tabelle 7.4](#) zeigt eine Übersicht über die wichtigsten dieser Kommandos; zum Teil kamen diese nicht in der Beschrei-

bung vor. Aber ob Sie nun mit der vorliegenden kurzen Anleitung oder mit einem umfangreichen Tutorial arbeiten – Geläufigkeit erreichen Sie nur, wenn Sie ständig üben.

Tastenkürzel/Befehl	Bedeutung
Moduswechsel	
<code>[I]</code>	Wechsel in den Eingabemodus an aktuelle Position
<code>[A]</code>	Wechsel in den Eingabemodus nach aktueller Position
<code>[⇧] + [I]</code>	Wechsel in den Eingabemodus am Zeilenanfang
<code>[⇧] + [A]</code>	Wechsel in den Eingabemodus am Zeilenende
<code>[O]</code> (Oh, nicht null)	Wechsel in den Eingabemodus unter aktuelle Zeile
<code>[⇧] + [O]</code> (Oh)	Wechsel in den Eingabemodus über aktuelle Zeile
<code>[Esc]</code>	Wechsel aus Eingabemodus in den Befehlsmodus
Navigation im Text	
<code>[H]</code>	Cursor um ein Zeichen nach links bewegen
<code>[J]</code>	Cursor um ein Zeichen nach unten bewegen
<code>[K]</code>	Cursor um ein Zeichen nach oben bewegen
<code>[L]</code>	Cursor um ein Zeichen nach rechts bewegen
<code>[O]</code> (null, nicht Oh)	Cursor an den Zeilenanfang
<code>[⇧] + [4]</code> , d. h. <code>[\$]</code>	Cursor ans Zeilenende
<code>[⇧] + [8]</code> , d. h.	Cursor zum vorangegangenen Satzanfang
<code>[⇧] + [9]</code> , d. h.	Cursor zum nächsten Satzanfang
<code>[Alt Gr] + [7]</code> (Windows) oder <code>[Alt] + [8]</code> (Mac), d. h. <code>[f]</code>	Cursor zum vorangegangenen Absatzanfang
<code>[Alt Gr] + [0]</code> (Windows) oder <code>[Alt] + [9]</code> (Mac), d. h. <code>[j]</code>	Cursor zum nächsten Absatzanfang
n <code>[Alt Gr] + [←]</code> (Windows) bzw. n <code>[Alt] + [7]</code> (Mac), d. h. n <code>[^]</code>	Cursor zu Zeichen n in der aktuellen Zeile

Tabelle 7.4 Die wichtigsten »vi«-Kommandos im Überblick

Tastenkürzel/Befehl	Bedeutung
	Cursor zum Anfang der vorangegangenen Zeile
oder	Cursor zum ersten Nichtleerzeichen der nächsten Zeile
+	Cursor zum Anfang der ersten sichtbaren Zeile
+	Cursor zum Anfang der Zeile in der Bildschirmmitte
+	Cursor zum Anfang der letzten sichtbaren Zeile
n + , d. h. n	Cursor in die Zeile, die dem angegebenen prozentualen Anteil des Textes entspricht
	Cursor ein Wort weiter (Satzzeichen als eigene Wörter)
+	Cursor ein Wort weiter (inklusive Satzzeichen)
	Cursor ein Wort zurück (Satzzeichen als eigene Wörter)
+	Cursor ein Wort zurück (inklusive Satzzeichen)
	Cursor zum nächsten Wortende
	Cursor zum vorangegangenen Wortende
n oder n	Cursor in Zeile n
	Cursor an den Textanfang
+	Cursor an den Anfang der letzten Zeile
n Tastenkürzel	die angegebene Funktion n-mal ausführen (z. B. für drei Zeichen nach links; nicht nur für Navigation verfügbar)
Scrollen (Bildschirm bewegen, ohne den Cursor zu verschieben)	
+	eine Zeile nach oben scrollen
+	eine Zeile nach unten scrollen
+	einen halben Bildschirm nach oben scrollen
+	einen halben Bildschirm nach unten scrollen
+	einen ganzen Bildschirm nach oben scrollen
+	einen ganzen Bildschirm nach unten scrollen

Tabelle 7.4 Die wichtigsten »vi«-Kommandos im Überblick (Forts.)























Tastenkürzel/Befehl	Bedeutung
 	die Zeile mit dem Cursor in die Bildschirmmitte scrollen
 	die Zeile mit dem Cursor an den oberen Bildschirmrand scrollen
 	die Zeile mit dem Cursor an den unteren Bildschirmrand scrollen
Zeichen suchen	
 Zeichen	erstes Vorkommen des angegebenen Zeichens in der aktuellen Zeile suchen
n  Zeichen	das n. Vorkommen des angegebenen Zeichens in der aktuellen Zeile suchen
 +  Zeichen	rückwärts in der aktuellen Zeile nach dem angegebenen Zeichen suchen
	den vorherigen Suchbefehl wiederholen
	vorherigen Suchbefehl in die andere Richtung wiederholen
	zwischen korrespondierenden Klammern (auch eckigen oder geschweiften) hin- und herspringen; mit Zähler völlig andere Bedeutung (siehe unter »Navigation im Text« weiter oben)
Volltextsuche und Ersetzen	
 RegExp 	vorwärts im Text nach dem angegebenen regulären Ausdruck suchen
 RegExp 	rückwärts im Text nach dem angegebenen regulären Ausdruck suchen
	in die aktuelle Richtung erneut nach dem vorherigen regulären Ausdruck suchen
 + 	in die entgegengesetzte Richtung erneut nach dem vorherigen regulären Ausdruck suchen
 wrapscan 	Suche darf Dokumentende/-anfang überschreiten (Standard)

Tabelle 7.4 Die wichtigsten »vi«-Kommandos im Überblick (Forts.)

Tastenkürzel/Befehl	Bedeutung
<code>: nowrapscan ↵</code>	Suche darf Dokumentende/-anfang nicht mehr überschreiten
<code>: sethlsearch ↵</code>	künftig alle gefundenen Ergebnisse hervorheben
<code>: nohlsearch ↵</code>	die aktuelle Hervorhebung ausblenden
<code>: setnohlsearch ↵</code>	die gefundenen Ergebnisse nicht mehr hervorheben
<code>: setincsearch ↵</code>	inkrementelle Suche – bereits während der Eingabe zu suchen beginnen
<code>: setnoincsearch ↵</code>	die inkrementelle Suche abschalten
<code>: [S] / Text1 / Text2 ↵</code>	das nächste Vorkommen von Text1 in der aktuellen Zeile durch Text2 ersetzen
<code>: [S] / Text1 / Text2 ↵</code>	alle Vorkommen von Text1 in der aktuellen Zeile durch Text2 ersetzen
<code>: [%] [S] / Text1 / Text2 ↵</code>	das erste Vorkommen von Text1 in jeder Zeile durch Text2 ersetzen
<code>: [%] [S] / Text1 / Text2 ↵</code>	alle Vorkommen von Text1 in jeder Zeile (das heißt alle Vorkommen im gesamten Text) durch Text2 ersetzen
<code>: [%] [S] / Text1 / Text2 ↵</code>	wie der vorherige Befehl, aber vor jedem einzelnen Ersetzungsvorgang nachfragen (die Option <code>[C]</code> steht auch für alle anderen Optionen zur Verfügung)
Sprungmarken	
<code>~ ~</code>	Sprung zur letzten Sprungmarke (Ergebnis einer Suche oder eines <code>[G]</code> -Sprungbefehls)
<code>[Strg] + [O] (Oh)</code>	in der Liste der Sprungmarken schrittweise rückwärts blättern
<code>[Strg] + [I]</code>	in der Liste der Sprungmarken schrittweise vorwärts blättern
<code>[M] [A] bis [M] [Z]</code>	an der aktuellen Cursorposition eine der benannten Marken A bis Z einrichten
<code>~ [A] bis ~ [Z]</code>	zu einer der benannten Marken A bis Z springen

Tabelle 7.4 Die wichtigsten »vi«-Kommandos im Überblick (Forts.)

Tastenkürzel/Befehl	Bedeutung
<code>'</code> <code>A</code> bis <code>'</code> <code>Z</code>	zum Anfang der jeweiligen Zeile springen, in der sich eine der benannten Marken A bis Z befindet
<code>:</code> marks <code>↵</code>	Liste aller Marken ausgeben
Löschen, Kopieren und Einfügen	
<code>X</code>	das Zeichen unter dem Cursor ausschneiden (löschen und in einem Puffer speichern)
<code>↵</code> + <code>X</code>	das Zeichen links neben dem Cursor ausschneiden
<code>D</code> Navigationsbefehl	die dem Navigationsbefehl entsprechende Menge Text ausschneiden; <code>D</code> <code>3</code> <code>H</code> löscht beispielsweise drei Zeichen nach links
<code>D</code> <code>D</code>	ganze Zeile ausschneiden
<code>C</code> Navigationsbefehl	wie <code>D</code> , aber anschließend in den Eingabemodus wechseln
<code>C</code> <code>C</code>	wie <code>D</code> <code>D</code> , aber anschließend in den Eingabemodus wechseln
<code>Y</code> Navigationsbefehl	die dem Navigationsbefehl entsprechende Menge Text in den Puffer kopieren (<i>yank</i>)
<code>Y</code> <code>Y</code>	ganze Zeile kopieren
<code>P</code>	den Text aus dem Puffer an der aktuellen Cursorposition einfügen
<code>↵</code> + <code>P</code>	den Text aus dem Puffer hinter der aktuellen Cursorposition einfügen
<code>X</code> <code>P</code>	das aktuelle und das folgende Zeichen vertauschen
Wiederholen, Rückgängigmachen, Wiederherstellen	
<code>U</code>	den letzten Befehl rückgängig machen (in <i>vi</i> nur einmal, in <i>Vim</i> mehrmals möglich – auch mit Zähler)
<code>Strg</code> + <code>R</code>	den zuletzt rückgängig gemachten Befehl wiederherstellen (nur <i>Vim</i>)
<code>.</code>	den vorangegangenen Befehl wiederholen

Tabelle 7.4 Die wichtigsten »vi«-Kommandos im Überblick (Forts.)

Tastenkürzel/Befehl	Bedeutung
Datei- und Fensterverwaltung	
<code>↵</code>	die aktuelle Datei speichern
<code>:w Neuer Name ↵</code>	eine Kopie der aktuellen Datei unter Neuer Name speichern; mit der aktuellen weiterarbeiten
<code>:edit Dateiname ↵</code>	das Editieren der aktuellen Datei beenden (muss gespeichert sein) und die angegebene laden
<code>:edit ! Dateiname ↵</code>	das Editieren der aktuellen Datei abbrechen (wird nicht gespeichert!) und die angegebene laden
<code>:wedit Dateiname ↵</code>	die aktuelle Datei automatisch speichern und schließen und die angegebene laden
<code>:hide edit Dateiname ↵</code>	die aktuelle Datei verbergen und die angegebene laden
<code>:buffers ↵</code>	nummerierte Liste aller Puffer (geöffneten Dateien)
<code>:buffer Nummer ↵</code>	zum Puffer mit der angegebenen Nummer wechseln
<code>:Q</code>	vi beenden (alle Dateien müssen gespeichert sein)
<code>:Q !</code>	Beenden von vi erzwingen (nicht gespeicherte Dateien werden verworfen)

Tabelle 7.4 Die wichtigsten »vi«-Kommandos im Überblick (Forts.)

7.4.2 Emacs

Der Name *Emacs* steht für *Editor Macros*, weil er ursprünglich als Satz von Makrobefehlen für einen älteren Editor entworfen wurde. Inzwischen ist er ein eigenständiger Editor, der in vielen verschiedenen Varianten für etliche Plattformen verfügbar ist, nicht nur für alle erdenklichen Unix-Arten, sondern beispielsweise auch für Windows. Die Hauptentwicklungslinie ist der *GNU Emacs*, den Sie von der Website des GNU-Projekts (www.gnu.org) oder den zahlreichen weltweiten Mirror-Sites herunterladen können. Die aktuelle Hauptversion ist 26.x, was Ihnen einen Eindruck davon vermitteln mag, wie lange diese Software bereits existiert⁷ – das Programm ist ausgereift und sehr stabil. Daneben gibt es den sogenannten *XEmacs*, was ein wenig verwirrend ist, weil beide Varianten sowohl auf der Konsole als auch in einer grafischen X-Window-Oberfläche laufen.

⁷ Heutzutage vielleicht nicht mehr so ganz wie früher, da Versionsnummern inzwischen recht inflationär vergeben werden. Beispielsweise sind die aktuellen Versionen (Anfang Mai 2019) der Browser Chrome und Mozilla Firefox 73.0 beziehungsweise 66.0.

Emacs verfügt über einen erheblich größeren Befehlsumfang als *vi*; es handelt sich letzten Endes nicht nur um einen Editor, sondern um einen vollwertigen Shell-Ersatz mit zusätzlichen Funktionen wie E-Mail, unzähligen Betriebsmodi für diverse Textsorten und sogar Spielen.

Die meisten *Emacs*-Befehle werden durch Tastenkombinationen mit **Strg** (auf englischen PC-Tastaturen und auf dem Mac **Ctrl**) oder **Meta** gebildet. Wenn Sie auf Ihre Tastatur schauen, werden Sie feststellen, dass Sie gar keine **Meta**-Taste haben; die gibt es nur an einigen alten Terminals und verschiedenen Unix-Workstations. In den meisten Terminalprogrammen und in der GUI-Version von *Emacs* kann die **Alt**-Taste dafür verwendet werden. Einige Terminal-Emulationen bestehen dagegen darauf, dass Sie **Meta**-Tastenkombinationen durch Drücken der Taste **Esc** und anschließendes Drücken der entsprechenden Kombinationstaste eingeben.



Die *Emacs*-Dokumentation verwendet eine etwas eigenwillige Schreibweise für die Tastenkombinationen. Diese Schreibweise wird hier kurz erläutert und anschließend verwendet, damit Sie sich nicht umzugewöhnen brauchen, wenn Sie die zahlreichen Hilfeseiten in *Emacs* selbst lesen. **Strg**-Tastenkombinationen werden durch C- (*control*), gefolgt von dem entsprechenden Zeichen angegeben, beispielsweise steht C-x für **Strg** + **X**. **Meta**-Tastenkürzel werden dagegen mit vorangestelltem M- dargestellt. M-x bedeutet also je nach der Umgebung, in der Sie arbeiten, entweder **Alt** + **X** oder **Esc**, gefolgt von **X**. Letzteres funktioniert übrigens immer.


Emacs wird durch Eingabe von `emacs` oder `emacs Dateiname` gestartet. Wenn Sie ihn nicht in einem GUI-Fenster, sondern in der aktuellen Konsole aufrufen möchten, lautet das Kommando `emacs -nw [Dateiname]`. Sie können mehrere Dateien auf einmal bearbeiten; die einzelnen Arbeitsbereiche für Dateien werden als *Buffer* bezeichnet. Falls Sie keinen Dateinamen angegeben haben, befinden Sie sich zunächst im Scratch-Buffer (Notizblock), der für ungespeicherte Notizen vorgesehen ist. Wenn Sie eine speicherbare Datei editieren möchten, müssen Sie diese zunächst mit C-x C-f Dateiname »besuchen« (die Original-*Emacs*-Dokumentation verwendet den Ausdruck »visit«). Falls diese Datei bereits existiert, wird sie geöffnet, ansonsten neu angelegt.




Wenn mehrere Buffer geöffnet sind, können Sie mithilfe der Tastenkombination C-x b Buffername **↵** zum Buffer mit dem angegebenen Namen wechseln (ohne Angabe eines Buffers wird reihum zum jeweils nächsten gewechselt). Eine Liste aller verfügbaren Buffer erhalten Sie mit C-x C-b in einem abgetrennten Bereich, der als *Fenster* bezeichnet wird. Mit C-x 2 können Sie ein solches Fenster selbst erzeugen; C-x 4 f Dateiname **↵** öffnet eine neue Datei in einem separaten Fenster. Mit C-x o wechseln Sie zwischen den beiden Fenstern hin und her. Mit M-C-v können Sie den Inhalt des Fensters scrollen, in dem Sie zurzeit nicht arbeiten. C-x 1 schließt das nicht aktive Fenster und behält nur das aktuelle.



Je nach Dateityp (den *Emacs*, ähnlich wie Windows, an der Dateiendung erkennt) befinden Sie sich in einem der verschiedenen *Emacs*-Arbeitsmodi, die vor allem Programmierern das

Leben erleichtern. Einige der wichtigsten Modi sind `Fundamental` (der grundlegende Modus für einfachen Text), `C` (für die gleichnamige Programmiersprache) oder `HTML` (zum Editieren von Webseiten). Jeder dieser Modi verfügt über besondere Funktionen wie automatische Einrückung, Anzeigen der korrespondierenden Klammer beim Schließen oder Ähnliches.


Sie können den Modus auch manuell über `M-x Modusname` wechseln: `M-x fundamental` wechselt beispielsweise in den Fundamental Mode, `M-x c-mode` in den C-Modus oder `M-x html-mode` in den HTML-Modus. `M-x` Schlüsselwort dient allgemein der Eingabe eines Befehls in der Langform und muss jeweils mit  abgeschlossen werden. Wenn Sie einfach `M-x`  eingeben, wird eine Liste aller möglichen Befehle angeboten; alternativ können Sie auch einen oder mehrere Anfangsbuchstaben eines Befehls eintippen und erhalten dann eine Liste der passenden Befehle.






Der einzelne Bereich am unteren Bildschirm- oder Fensterrand, in den Sie mit `M-x` gelangen, wird *Minibuffer* genannt. Sie können ihn durch dreimaliges Drücken von  wieder verlassen.


Zeichenweise durch den Text navigieren Sie mit den Pfeiltasten oder alternativ mit `C-f` (nach rechts, *forward*), `C-b` (nach links, *backward*), `C-p` (nach oben, *previous line*) und `C-n` (nach unten, *next line*). `C-a` bewegt den Cursor zum Zeilenbeginn und `C-e` zum Zeilenende. `C-v` scrollt um einen ganzen Bildschirm nach unten, `M-v` nach oben. `M-f` wandert um ein Wort nach rechts, `M-b` nach links. `M-<` springt zum Anfang der gesamten Datei, `M->` ( +  + ) zu ihrem Ende.

`C-d` löscht das Zeichen unter dem Cursor, während  das vorherige Zeichen entfernt. `M-d` löscht das folgende Wort, `M-`  das vorangegangene. `C-k` tilgt den Bereich vom Cursor bis zum Ende der Zeile.

`C-Leertaste` setzt eine Markierung. Der markierte Bereich reicht automatisch von der gesetzten Markierung bis zur aktuellen Cursorposition. Um den gesamten Bereich zu löschen, genauer gesagt, auszuschneiden, wird die Tastenkombination `C-w` verwendet. `M-w` kopiert den Bereich dagegen zum späteren Einfügen. In beiden Fällen können Sie ihn mithilfe von `C-y` wieder einfügen.

`C-s` leitet eine inkrementelle Suche ein – während der Eingabe des Suchbegriffs springt der Cursor zur jeweils nächstgelegenen Stelle, die diesem Begriff entspricht. Entsprechend sucht `C-r` rückwärts. `C-s C-s` wiederholt die letzte Suche.  verlässt das Suchprogramm vollständig. `M-C-s` sucht nach einem regulären Ausdruck anstelle einer einfachen Zeichenfolge. Die Syntax der regulären Ausdrücke entspricht *grep*.

`M-%` ist der Befehl zum Suchen und Ersetzen. Zunächst wird nach der zu ersetzenden Zeichenfolge gefragt, anschließend nach dem Ersetzungstext. Die Eingabe beider Zeichenfolgen muss mit  abgeschlossen werden. An jeder einzelnen Fundstelle werden Sie gefragt, ob Sie die Ersetzung durchführen möchten.  oder Leertaste bedeutet Ja,  oder  steht für Nein, und  heißt, dass alle künftigen Fundstellen ohne weitere Rückfrage ersetzt werden sollen.

Wenn Sie Ihre aktuelle Arbeit speichern möchten, verwenden Sie dazu den Befehl `C-x C-s`. `C-x s` speichert alle geänderten Buffer auf einmal. `C-x C-w Dateiname`  sichert die Datei dagegen unter dem angegebenen neuen Namen. Mit `C-x C-c` beenden Sie *Emacs*. Vorher werden Sie gefragt, ob Sie die modifizierten Buffer speichern möchten.

Was bereits über *vi* gesagt wurde, gilt für *Emacs* erst recht: Es gibt Unmengen weiterer Befehle, für die in diesem Buch leider kein Platz ist. Mit dem hier gebotenen Ausschnitt aus der Funktionsvielfalt von *Emacs* können Sie beliebige Textdateien bearbeiten. Mit etwas Übung werden Sie damit irgendwann schneller arbeiten können als mit einem grafisch orientierten Editor, in dem Sie oft zur Maus greifen müssen, um spezielle Befehle zu verwenden. Dennoch ist auch *Emacs* mit einem umfangreichen Menü ausgestattet, solange er in einem GUI läuft. Da die Tastenkürzel jeweils neben den einzelnen Menübefehlen stehen, können Sie mithilfe des Menüs leicht neue Befehle erlernen.

Auch für *Emacs* folgt in [Tabelle 7.5](#) eine etwas ausführlichere Kommandoliste.

Tastenkürzel/Befehl	Bedeutung
Dateien, Buffer und Fenster	
<code>C-x C-f Dateiname</code>	die angegebene Datei laden (<i>Emacs</i> -Begriff <i>visit</i>); falls sie noch nicht existiert, wird sie neu angelegt
<code>C-x C-v</code>	den Namen der aktuellen Datei im Minibuffer anzeigen, um ihn zu editieren und die resultierende Datei zu laden/anzulegen
<code>C-x i Dateiname</code>	die angegebene Datei an der aktuellen Cursorposition in die derzeitige Datei einfügen
<code>C-x b Buffername</code>	zum Buffer mit dem angegebenen Namen wechseln
<code>C-x k [Buffername]</code>	den angegebenen bzw. den aktuellen Buffer schließen
<code>C-x C-b</code>	Liste aller derzeit geöffneten Buffer in separatem Fenster (Abschnitt)
<code>e</code> oder <code>f</code>	in der Bufferliste den aktiven Buffer im anderen Fenster öffnen
<code>d</code>	in der Bufferliste einen Buffer zum Löschen markieren
<code>s</code>	in der Bufferliste einen Buffer zum Speichern markieren
<code>u</code>	in der Bufferliste eine Markierung aufheben
<code>x</code>	in der Bufferliste alle markierten Lösch- und Speichervorgänge ausführen

Tabelle 7.5 Die wichtigsten Emacs-Kommandos im Überblick

Tastenkürzel/Befehl	Bedeutung
C-x 2	ein zweites Fenster erzeugen
C-x 4 f Dateiname	die angegebene Datei in einem neuen Fenster öffnen/erstellen
C-x o	Wechsel zum jeweils anderen Fenster (auch in die Bufferliste)
C-x v	im jeweils anderen Fenster scrollen
C-x 1	das nicht aktive Fenster schließen
Zähler	
C-u n Befehl	den angegebenen Befehl so oft durchführen, wie durch die Anzahl <i>n</i> angegeben
M-n Befehl	den angegebenen Befehl so oft durchführen, wie durch die Anzahl <i>n</i> angegeben
Navigation im Text	
C-f	Cursor um ein Zeichen nach rechts bewegen (<i>forward</i>)
C-b	Cursor um ein Zeichen nach links bewegen (<i>backward</i>)
C-p	Cursor um eine Zeile nach oben bewegen (<i>previous line</i>)
C-n	Cursor um eine Zeile nach unten bewegen (<i>next line</i>)
M-b	Cursor um ein Wort nach rechts bewegen
M-f	Cursor um ein Wort nach links bewegen
C-a	Cursor zum Zeilenanfang bewegen
C-e	Cursor zum Zeilenende bewegen
M-a	Cursor zum vorangegangenen Satzbeginn bewegen
M-e	Cursor zum nächsten Satzbeginn bewegen
C-v	um einen ganzen Bildschirm nach unten scrollen
M-v	um einen ganzen Bildschirm nach oben scrollen
M-<	Cursor zum Textanfang bewegen
M->	Cursor zum Textende bewegen

Tabelle 7.5 Die wichtigsten Emacs-Kommandos im Überblick (Forts.)







Tastenkürzel/Befehl	Bedeutung
Löschen	
C-d	das Zeichen unter dem Cursor löschen
	das Zeichen links vom Cursor löschen
M-d	das folgende Wort löschen
M- 	das vorangegangene Wort löschen
C-k	vom Cursor bis zum Ende der Zeile löschen
M-k	vom Cursor bis zum Satzende löschen
C-t	das aktuelle und das vorangegangene Zeichen vertauschen
Markieren, Kopieren und Einfügen	
C-Leertaste	Markierung setzen
C-w	den Text von der Markierung bis zur aktuellen Cursorposition ausschneiden
M-w	den Text von der Markierung bis zur aktuellen Cursorposition kopieren
C-y	den zuletzt ausgeschnittenen oder kopierten Text an der aktuellen Cursorposition einfügen
M-y	durch die zuvor ausgeschnittenen oder kopierten Texte »blättern«, um den richtigen auszusuchen
Suchen und Ersetzen	
C-s	inkrementelle Suche vorwärts einleiten
C-r	inkrementelle Suche rückwärts einleiten
C-s C-s	erneut nach dem vorangegangenen Suchbegriff suchen
	den Suchmodus verlassen
M-C-s	nach einem regulären Ausdruck anstelle eines einfachen Strings suchen
M-%	interaktives Suchen und Ersetzen: Suchbegriff eingeben, Ersetzungsstring eingeben, für jede Stelle mit  oder Leertaste bestätigen, mit  oder  ablehnen oder mit ! für alle Fundstellen bestätigen

Tabelle 7.5 Die wichtigsten Emacs-Kommandos im Überblick (Forts.)

Tastenkürzel/Befehl	Bedeutung
Speichern und Beenden	
C-x C-s	den aktuellen Buffer speichern
C-x s	alle Buffer (mit Nachfrage) speichern
C-x C-w Dateiname	die aktuelle Datei unter dem angegebenen neuen Namen speichern
C-x C-c	<i>Emacs</i> beenden (mit Nachfrage, ob die geänderten Buffer gespeichert werden sollen)
Spiel und Spaß	
M-x doctor	startet eine Sitzung mit dem eingebauten Psychotherapeuten (eine Variante von Joseph Weizenbaums Eliza)
M-x tetris	startet das gleichnamige Spiel; je nach <i>Emacs</i> -Variante und -Umgebung mit ASCII-Zeichen oder Farbgrafik

Tabelle 7.5 Die wichtigsten Emacs-Kommandos im Überblick (Forts.)

Abgesehen davon, können Sie diesen Editor völlig frei konfigurieren. Zum einen können Sie die Tastenkürzel selbst frei belegen, zum anderen ist *Emacs* beliebig erweiterbar: Er ist mit einer eigenen Variante der Programmiersprache LISP ausgestattet. Diese Sprache könnte Programmierer durch ihren eigenwilligen Ansatz abschrecken, sie ist aber speziell an die Funktionen von *Emacs* angepasst und erleichtert so das Programmieren von Zusatzfunktionen.

7.5 Grafische Benutzeroberflächen

Neben der bis jetzt behandelten Konsole sind alle modernen Linux-Distributionen auch mit einer grafischen Benutzeroberfläche ausgestattet. Diese Oberfläche besteht aus zwei verschiedenen Bestandteilen, die aufeinander aufbauen: dem *X-Window-Server* oder kurz *X-Server* und dem *Window-Manager*.

7.5.1 Der X-Server

Ein *X-Server* ist das grundlegende Programm, das die allgemeinen Zeichenfunktionen zur Verfügung stellt. Er kann mit anderen Worten die verschiedensten Grafikobjekte wie Rechtecke, Linien oder Bilddateien auf den Bildschirm zeichnen, außerdem verarbeitet er die grundlegende Steuerung mit der Maus. Dem *X-Server* ist es übrigens egal, ob die Anwendung, die er anzeigt und für die er Befehle entgegennimmt – der *X-Client* –, auf demselben Rechner läuft oder irgendwo im Netzwerk.