HOWTO für den Emacs Einsteiger

Jeremy D. Zawodny (Jeremy@Zawodny.com) und Christel Weyrauch (christelweyrauch@gmx.de)
v1.12, 20. Januar 2002

Dieses Dokument führt Linux-Anwender in den Emacs-Editor ein. Es geht von einer geringen Vertrautheit mit dem `vi` oder einem ähnlichen Editor aus.

Inhaltsverzeichnis

1 Einführung

1.1 Copyright

1.2 Bezugsquellen

Die aktuellste englische Version dieses Dokuments ist gewöhnlich unter

`http://www.wcnet.org/jzawodn/emacs/`

verfügbar. Die deutsche Version findet man beim DLHP:

`http://www.tu-harburg.de/dlhp/`

1.3 Leserkreis und Ziel

Dieses Dokument richtet sich an den Linux-Anwender, der daran interessiert ist, ein bisschen über den Emacs zu lernen und ihn auszuprobieren. Es entstand eigentlich als Entwurf für ein kurzes Tutorial, das ich bei einem Treffen der Toledo Area Linux User Group geben musste:

`http://www.talug.org/`

Dank des hilfreichen Feedbacks, das ich von der Gruppe bekam, ist es seitdem etwas gewachsen. Näheres dazu finden Sie bei den Danksagungen.

Ich habe gesagt, dass sich dieses Dokument an den Linux-Anwender richtet, doch gibt es in diesem Dokument praktisch nichts Linux-spezifisches. Es ist auf alle Unix-Varianten und selbst auf dem Emacs unter Microsoft Windows anwendbar. Doch da dieses Dokument Teil des Linux-Dokumentationsprojektes ist, möchte ich noch einmal festhalten, dass es für Linux-Anwender entwickelt wurde, weil es das eben wurde.

Und zum Schluss noch ein Hinweis für diejenigen von Ihnen, die die Bezeichnung »GNU/Linux« dem ganz einfachen »Linux« vorziehen: Sie können gerne in Gedanken bei jedem Vorkommen von »Linux« in diesem Dokument dieses durch »GNU/Linux« ersetzen. Lesen sie hierzu auch folgendes Dokument:

`http://www.gnu.org/gnu/linux-and-gnu.html`

Zwar bin ich mit der Argumentation und dem Geist hinter der Idee nicht uneins, doch fühle ich mich nicht gezwungen, GNU/Linux zu schreiben.

1.4 Was ist der Emacs?

Der Emacs ist Unterschiedliches für unterschiedliche Menschen. Abhängig davon, wen man fragt, könnte man irgendeine der folgenden Antworten bekommen:

- ein Text-Editor
- ein Mail Client
- ein News Reader
- ein Word-Prozessor
- eine Religion
- eine integrierte Entwicklungsumgebung
- Was immer Sie wollen, das er ist!

Doch für unsere Zwecke lassen Sie uns einfach so tun, als wäre er ein Text-Editor - ein erstaunlich flexibler Text-Editor. Wir werden später tiefer in die Frage einsteigen. Der Emacs wurde von Richard Stallman geschrieben (Gründer der Free Software Foundation:

`http://www.fsf.org/`

und des GNU Projektes),

`http://www.gnu.org/`

und er pflegt ihn noch heute.

Der Emacs ist einer der populärsten und leistungsfähigsten Text-Editoren für Linux (und Unix). In Popularität steht er an zweiter Stelle nur hinter dem `vi`. Er ist bekannt für sein umfangreiches Bündel an Leistungsmerkmalen, seine Fähigkeit, leicht angepasst werden zu können und seinen Mangel an Fehlern. Sein enormes Bündel an Leistungsmerkmalen und seine Fähigkeit, leicht angepasst werden zu können, sind im Grunde das Resultat davon, wie der Emacs entwickelt und implementiert wurde. Ohne auf alle Details einzugehen, weise ich einfach darauf hin, dass der Emacs nicht »nur ein Editor« ist. Es ist ein Editor, der hauptsächlich in der Programmiersprache Lisp geschrieben wurde. Im Kern des Emacs befindet sich ein kompletter Lisp Interpreter, der in C geschrieben wurde. Nur die grundlegendsten und elementarsten Teile vom Emacs sind in C geschrieben. Der größere Teil des Editors wurde eigentlich in Lisp geschrieben. So hat der Emacs, in gewissem Sinne, eine ganze Programmiersprache »in sich eingebaut«, die Sie benutzen können, um ihn anzupassen, zu erweitern und sein Verhalten zu verändern.

Der Emacs ist außerdem einer der ältesten Editoren, die es so gibt. Die Tatsache, dass er über die letzten 20 (?) Jahre hinweg von tausenden von Programmierern benutzt wurde, bedeutet, dass es viele Zusatzpakete gibt. Diese Zusätze erlauben es Ihnen, den Emacs Dinge tun zu lassen, die Stallman wahrscheinlich niemals für möglich gehalten hätte, als er erstmals die Arbeit am Emacs begann. Mehr davon in einem späteren Abschnitt.

Es gibt viele andere Dokumente und Websites, die einen besseren Überblick über den Emacs, seine Geschichte und damit zusammenhängende Themen bieten. Anstatt zu versuchen, an dieser Stelle davon viel zu reproduzieren, schlage ich vor, Sie testen einige der Angaben, die im Abschnitt 6 (Andere Quellen) in diesem Dokument aufgeführt werden.

1.4.1 Portierungen und Versionen

Es ist wichtig drauf hinzuweisen, dass es eigentlich zwei unterschiedliche Emacs-Editoren gibt: GNU Emacs und Xemacs. Beide haben den gleichen Ursprung und verfügen im Wesentlichen über die gleichen Eigenschaften. Dieses Dokument basiert auf dem GNU Emacs (insbesondere Version 20.3), aber viel von dem, was Sie hier lesen werden, trifft ebenso gut auf den Xemacs und ältere Versionen des GNU Emacs zu. In dem ganzen Dokument werde ich einfach vom »Emacs« sprechen. Wenn ich das tue, haben Sie dies bitte im Hinterkopf.

1.4.2 Wie Sie den Emacs bekommen können

Den Emacs zu bekommen, ist einfach. Wenn Sie eine bekannte Linux-Distribution benutzen, wie Debian, RedHat, Slackware, SuSE oder irgendeine der anderen, ist der Emacs wahrscheinlich ein Zusatzpaket, das Sie von Ihrem Distributionsmedium aus installieren können. Wenn nicht, können Sie sich den Emacs Source Code holen und ihn selbst kompilieren. Besuchen Sie die GNU Website, um die genaue Adresse zu bekommen:

`http://www.gnu.org/software/emacs/emacs.html`

2 Mit dem Emacs arbeiten

2.1 Den Emacs starten und beenden

Wenn Sie ein neuer Anwender sind, werden Sie wahrscheinlich den Emacs starten, um damit herumzuspielen und ihn auszuprobieren. Ist der Emacs erst einmal gestartet und Sie wollen ihn schließen, wissen Sie vielleicht nicht, was zu tun ist. Wenn Sie also vorher noch nie mit dem Emacs gearbeitet haben, versuchen Sie es jetzt gleich einmal. Tippen Sie `emacs` an Ihrem Shellprompt und drücken Sie die »Return«-Taste. Der Emacs sollte dann starten. Wenn nicht, ist er entweder nicht installiert oder nicht in Ihrem Pfad.

Wenn Sie sich den Emacs angesehen haben, müssen Sie wissen, wie er beendet wird. Mit der Tastenkombination »C-x C-c« wird er verlassen. Die »C-x« Schreibweise bedeutet: halten Sie die »Ctrl«-Taste (auf einer deutschen Tastatur ersetzen Sie bei Befehlen die Control-Taste bitte immer durch die Steuerungs-Taste, »Strg«) gedrückt und drücken Sie »x«. In diesem Fall, müssen Sie dann die »Ctrl«-Taste gedrückt halten und auf »c« drücken, um den Befehl zu vollenden.

Die beim Emacs benutzten Tastenkombinationen erscheinen Ihnen vielleicht zuerst etwas komisch, fremd und wahrscheinlich sogar unbequem - besonders dann, wenn Sie ein `vi`-Anwender sind. Anders als der `vi` hat der Emacs keine besonderen Modi, um Text zu editieren und Befehle auszuführen.

Zur Erinnerung: `emacs` startet den Emacs. Mit »C-x C-c« wird er geschlossen.

2.1.1 Was Sie sehen werden

Wenn der Emacs gestartet wird, nimmt er ein ganzes X Fenster ein (oder den gesamten Bildschirm, wenn Sie von einer Konsole aus arbeiten, anstelle des X Window Systems). Entlang des oberen Randes sehen Sie ein Menue, etwas Text im Hauptteil des Bildschirms, und ein paar Zeilen am unteren Rand.

Es wird in etwa so aussehen wie diese ASCII-Skizze:

```
+----------------------------------------------------------------------+
|Buffers Files Tools Edit Search Mule Help                              |
```

```
|                                                                     |
|Welcome to GNU Emacs, one component of a Linux-based GNU system.     |
|                                                                     |
|                                                                     |
|                                                                     |
| ...                                                                 |
|                                                                     |
|---1:---F1  *scratch*         (Lisp Interaction)--L1--All------------|
|For information about the GNU Project and its goals, type C-h C-p.   |
+---------------------------------------------------------------------+
```

Beachten Sie: der Emacs wird normalerweise den gesamten Bildschirm oder das ganze Fenster ausfüllen. Ich habe das obige Beispiel kleiner dargestellt, um hier Platz zu sparen. Sie werden auch einen Begrüßungstext im Emacs sehen, wenn Sie ihn das erste Mal starten. Ich habe ihn auch ausgelassen und anstatt dessen durch »`...`« ersetzt. Der Begrüßungstext gibt lediglich die genaue Emacs-Version an, die Sie benutzen und verweist Sie auf die Online-Hilfe und damit verwandte Themen.

Die Menueleiste Die allererste Zeile der Emacs-Oberfläche ist ein Menue. Wenn Sie mit X arbeiten, werden Sie sie als die üblichen Pull-Down-Menues wahrnehmen, auf die Sie mit Ihrer Maus zugreifen können. Wenn nicht, müssen Sie Tastenkürzel verwenden (werden hier nicht behandelt), um auf die Menues zugreifen zu können.

Die Statuszeile und der Mini-Puffer Von den letzten zwei Zeilen der Emacs-Oberfläche ist die oberste im Wesentlichen eine *Statuszeile*. Sie beinhaltet Informationen über den Puffer in dem Sie arbeiten, in welchem Modus sich der Emacs befindet, und verschiedene andere Dinge. Merken Sie sich für den Moment einfach, dass sie da ist.

Die unterste Zeile wird *Mini-Puffer* genannt. Er wird vom Hauptpuffer durch die Statuszeile getrennt, die wir gerade besprochen haben. Sie können den Mini-Puffer als die »Kommandozeile« des Emacs betrachten. Hier werden Befehle angezeigt, die Sie dem Emacs erteilen und hier werden - je nach dem, was Sie tun - Statusmeldungen ausgegeben.

Das, was ich die Statuszeile genannt habe, wird in Dokumentationen zum Emacs normalerweise als Moduszeile bezeichnet. Hier zeigt der Emacs Informationen über den/die augenblicklichen Modus/Modi, die Sie vielleicht benutzen, aber auch Dinge wie das aktuelle Datum und die Zeit, die Zeilennummer, Dateigröße und fast alles, was Sie dort möglicherweise sehen möchten.

2.2 Einige Fachausdrücke

Dieser Abschnitt behandelt die grundlegendsten Emacs-Fachausdrücke, denen Sie bei der Benutzung des Emacs und dem Lesen über den Emacs begegnen werden.

2.2.1 Puffer & Dateien

Anders als bei einigen Editoren, bleibt beim Emacs eine Datei an der Sie arbeiten, nicht die ganze Zeit »geöffnet«. Anstatt dessen, liest der Emacs die Datei in einen *Speicherpuffer* ein. Während Sie den Puffer editieren und mit den Daten arbeiten, ändert sich nichts auf der Festplatte. Nur wenn Sie wirklich den Puffer speichern, wird die Datei auf der Festplatte aktualisiert. Es gibt bei diesem Verfahren Vor- und Nachteile, es ist aber nur wichtig, dass Sie verstehen, dass es so funktioniert.

Aus diesem Grund trifft man in der Emacs-Dokumentation bei Modi, Paketen und so weiter auf den Begriff »Puffer«. Machen Sie sich einfach klar, dass Puffer »eine Kopie der Datei, die gerade im Speicher ist« bedeutet. Oh, es ist noch wichtig, darauf hinzuweisen, dass ein Puffer sich nicht immer auf eine bestimmte Datei auf der Festplatte bezieht. Der Emacs erzeugt oft Puffer aufgrund der Befehle, die Sie eingeben. Diese Puffer können das Ergebnis eines Befehls, eine Auswahlliste etc. beinhalten.

2.2.2 Point & Region

Im Emacs-Jargon werden Sie oft Hinweise auf den sogenannten »Point« (Eingabepunkt) hören oder sehen. Im allgemeinen ist der Point der Cursor. Der eigentliche Unterschied zwischen Point und Cursor ist wahrscheinlich nicht wichtig, wenn man gerade angefangen hat, sich mit dem Emacs zu beschäftigen. Aber wenn Sie neugierig sind, sollten Sie es so betrachten: der Cursor ist die visuelle Darstellung des Points. Der Cursor ist immer »auf« einer bestimmten Buchstabenposition im aktuellen Puffer. Der Point, andererseits, ist im Raum »zwischen den Buchstaben«. Man könnte also sagen, dass wenn der Cursor sich auf dem Buchstaben »h« im Wort »the« befindet, der Point zwischen »t« und »h« ist.

Wie viele moderne Editoren kann der Emacs auf einen Abschnitt des aktuellen Puffers begrenzt Aktionen ausführen: Texteinzug, Rechtschreibprüfung, neu Formatieren, Ausschneiden, Kopieren, Einfügen usw.. Mit der Tastatur oder Maus können Sie einen Textblock hervorheben (oder »markieren«) und dann Aktionen nur in diesem Block oder Textabschnitt ausführen. Beim Emacs wird dieser Textblock »Region« genannt.

2.2.3 Fenster

Okay, das Folgende wird für alle, die irgendwann einmal mit einem GUI Interface gearbeitet haben, etwas verwirrend sein. Erinnern Sie sich einfach daran, dass der Emacs, lange bevor GUI Interfaces und Window Manager populär waren, entwickelt wurde.

Beim Emacs ist ein »Fenster« ein Teil des Bildschirmes, auf dem ein Puffer dargestellt wird. Wenn der Emacs gerade erst gestartet wurde, hat man ein Fenster auf dem Bildschirm. Einige Emacs-Funktionen (wie die Hilfe und die Dokumentation) öffnen oft vorübergehend zusätzliche Fenster in Ihrem Emacs-Bildschirm.

Emacs-Fenster haben im GUI Sinn nichts mit X Window Fenstern gemein. Man kann zusätzliche X Window Fenster öffnen, um Emacs-Puffer darzustellen, vielleicht um zwei Dateien nebeneinander zu vergleichen. Diese neuen X Window Fenster werden im Emacs-Jargon »Rahmen« (Frames) genannt; lesen Sie dazu weiter.

2.2.4 Rahmen

Beim Emacs ist ein »Rahmen« ein separates X Window Fenster, in dem ein Emacs-Puffer dargestellt wird. Beide sind jedoch Teil der gleichen Emacs-Session. Das Verhalten ähnelt etwas - aber nicht allzu sehr - dem, was passiert, wenn Sie »Alt+N« im Netscape Naviagator drücken.

2.3 Grundlagen der Tastaturbedienung

Dieser Abschnitt behandelt die Grundlagen der Tastaturbedienung des Emacs. Wie bei jedem leistungsfähigen Editor ist alles, was man mit dem Emacs machen kann, nur wenige Tastenkombinationen entfernt.

Wenn Sie ein `vi`-Anwender sind, brauchten Sie wahrscheinlich etwas, um sich an die Vorstellung zu gewöhnen, »k«-, »j«-, »l«-, »h«-Tasten zu benutzten, um sich eine Zeile höher, tiefer, einen Buchstaben vor und zurück zu bewegen. Es wird in der Tat wahrscheinlich einige Stunden oder sogar Wochen gedauert haben, bevor Sie sich beim Benutzen der unterschiedlichen Tastenkombinationen, die es beim `vi` gibt, mühelos in einem Dokument bewegen konnten.

Der Emacs ist da nicht anders. Man muss andere Tastenkombinationen und Befehle lernen. Genau wie beim `vi` muss man nur die Grundlagen beherrschen, um viel Arbeit erledigt zu bekommen. Dann, mit der Zeit, kann man langsam sein Wissen erweitern und schnellere Wege finden, wie man Dinge tut.

2.3.1 Befehlstasten (Meta, Esc, Steuerung und Alt)

Wie Sie schnell lernen werden, macht der Emacs viel Gebrauch von Mehrfach-Tastenkombinationen. Da er kein Modal-Editor wie der `vi` ist, muss man nicht darüber nachdenken, ob man sich im »Befehlsmodus« oder »Editiermodus« befindet, bevor man den Cursor bewegen oder einen Befehl ausführen kann. Anstatt dessen drückt man die richtige Tastenkombination und der Emacs tut gewöhnlich, was er soll.

Die Tasten, die man im Emacs am meisten benutzt, werden in der Dokumentation normalerweise mit »C« (für Control, »Ctrl« - auf einer deutschen Tastatur ist »C« durch die Steuerungs-Taste, »Strg«, zu ersetzen) und »M« für (»Meta«) abgekürzt. Während die meisten modernen PC-Tastaturen eine oder mehrere Tasten für »Ctrl« haben, haben wenige eine für »Meta«. Sie müssen in Gedanken entweder »Esc« oder »Alt« durch die »Meta«-Taste ersetzen. Bei den meisten Standardkonfigurationen machen »Esc« und »Alt« im Wesentlichen das Selbe.

Wenn Sie also in irgendeiner Dokumentation, die sich auf den Emacs bezieht, einen Verweis auf »C-x fy sehen, bedeutet dies: drücken Sie »C-x« und dann »f« (»Strg-x f« dann natürlich auf einer deutschen Tastatur). Und wenn Sie einen Hinweis auf etwas wie »M-x shell« sehen, bedeutet dies: drücken Sie »Alt-x« und tippen Sie das Wort »shell«.

Ein sehr nützlicher Befehl für Anfänger ist »M-x apropos« oder »C-h a«. Apropos durchsucht die Emacs Online-Dokumentation nach allen Funktionen und regulären Ausdrücken, die Sie eingeben. Dies ist eine fantastische Möglichkeit, alle Kommandos zu entdecken, die mit Frames zu tun haben. Geben Sie einfach »C-h a« und dann »frame« ein.

2.3.2 Sich in einem Puffer bewegen

Da sie nun wissen, was alle diese tollen Abkürzungen bedeuten, folgt jetzt eine Liste mit den gebräuchlichsten Tastenkombinationen, um sich innerhalb eines Puffers zu bewegen:

```
Tastenkombination  Aktion
---------------------------------------------
 C-p         eine Zeile hoch
 C-n         eine Zeile runter
 C-f         einen Buchstaben vorwärts
 C-b         einen Buchstaben zurück
 C-a         Zeilenanfang
 C-e         Zeilenende
 C-v         eine Seite runter
 M-v         eine Seite hoch
 M-f         ein Wort weiter
 M-b         ein Wort zurück
 M-<         Beginn des Puffers
 M->         Ende des Puffers
 C-g         aktuellen Arbeitsvorgang beenden
---------------------------------------------
```

Und Sie haben es vielleicht erwartet, die Cursor-Tasten (oder Pfeiltasten) funktionieren genau, wie Sie erwartet haben. Ihre `Backspace`-Taste möglicherweise nicht. Das ist eine andere Geschichte. :-(

2.3.3 Wesentliche Befehle

Okay, da Sie nun wissen, wie man sich in einem Puffer bewegt, wie ist es mit dem Öffnen und Speichern von Dateien oder der Suchefunktion? Hier kommen einige grundlegende Befehle.

Bevor wir uns direkt auf diese Befehle stürzen, muss ich kurz darstellen, wie das funktioniert.

Alle »Befehlstasten« im Emacs (die, die »M-x irgendwas« oder »C irgendwas« sind), sind nur Kürzel für Funktionen, die Teil des Emacs sind. Man kann alle diese Funktionen durch Eintippen von »M-x Funktionsnamen« und Drücken der »Return«-Taste aufrufen. Man kann auch die Tastaturkürzel für diese Funktion benutzen (wenn sie eine hat).

Zum Beispiel heißt die Emacs-Funktion, die einen Puffer auf Festplatte speichert, »save-buffer«. Sie ist standardmäßig auch mit »C-x C-s« belegt. Man kann also entweder das Kürzel benutzen, um den aktuellen Puffer zu speichern oder man könnte »M-x save-buffer« eintippen, um das gleiche Resultat zu erzielen.

Die gebräuchlichsten Funktionen haben standardmäßig zusätzliche Tastenkürzel. Einige von ihnen sind unten aufgelistet.

```
Tastenkürzel  Funktion           Beschreibung
--------------------------------------------------------------------------------
--
C-x C-s     save-buffer        Speichere den aktuellen Puffer auf Festplatte
C-x u       undo               Mache den letzten Arbeitsvorgang rückgängig
C-x C-f     find-file          Öffne eine Datei von Festplatte
C-s         isearch-forward    Inkrementelle Suche vorwärts
C-r         isearch-backward   Inkrementelle Suche rückwärts
            replace-string     Suchen & Ersetzen einer Zeichenkette
            replace-regexp     Suchen & Ersetzen bei regulären Ausdrücken
C-h t       help-with-tutorial Benutze das interaktive Emacs-Tutorial
C-h f       describe-function  Zeige den Hilfetext für eine Funktion an
C-h v       describe-variable  Zeige den Hilfetext für eine Variable an
C-h x       describe-key       Zeige an, was eine Tastenfolge macht
C-h a       apropos            Zeige den Hilfetext für einen (regulären) Aus-
druck an
C-h F       view-emacs-FAQ     Zeige die Emacs-FAQ an
C-h i       info               Lese die Emacs Dokumentation
C-x r m     bookmark-set       Setze ein Lesezeichen. Nützlich bei Suchen
C-x r b     bookmark-jump      Springe zu einem Lesezeichen
--------------------------------------------------------------------------------
--
```

Beim Ausprobieren vieler dieser Funktionen werden Sie feststellen, dass Sie viele zu einer Eingabe auffordern. Das wird immer im Mini-Puffer erfolgen. Dies ähnelt dem Gebrauch der »:«-Kommandos im `vi` oder den meisten Kommandos, die Sie in Ihrer bevorzugten Unix Shell benutzen würden.

Der Emacs verfügt buchstäblich über hunderte von eingebauten Funktionen. Die obige Liste ist ein kleines Beispiel all derer, die ich regelmäßig benutze. Für eine vollständigere Auflistung der verfügbaren Funktionen und eine ausführlichere Dokumentation zu denen, die ich oben erwähne, sehen Sie sich die Online-Hilfe an.

2.3.4 Tab-Ergänzung

Wie viele populäre Unix Shells (`bash`, `csh`, `tcsh` usw.) kann der Emacs Befehle über die »Tab«-Taste ergänzen. Es ist sogar so, dass die Befehlsergänzung in der `bash`, der im Emacs nachgebildet wurde. Wenn Sie dieses Leistungsmerkmal in der `bash` benutzen, werden Sie sich also ganz zu Hause fühlen.

Versuchen Sie zum Beispiel »M-x search« und drücken dann »Tab«. Der Emacs wird einen Bindestrich anfügen, um anzuzeigen, dass es diverse Vervollständigungsmöglichkeiten gibt, die aber alle einen Bindestrich als nächsten Buchstaben haben. Wenn die »Tab«-Taste ein weiteres Mal betätigt wird, wird Emacs Ihnen eine Liste aller Möglichkeiten anzeigen, aus denen Sie auswählen können. Sie werden feststellen, dass er dies in einem »neuen Fenster« tut. Er unterteilt Ihr Display zeitweise in zwei Fenster: eines, das den Puffer enthält, den Sie gerade editieren und ein weiteres, das die Liste aller möglichen Ergänzungen für »search« enthält. Sie können »C-g« tippen, um den Selektionsprozess zu verlassen und das neue Fenster zu schließen.

2.4 Tutorial, Hilfe & Info

Der Emacs hat ein Online-Tutorial, das Sie durch die grundlegenden Editiereigenschaften und -funktionen führt, die jeder kennen sollte. Es erklärt auch, wie die anderen Hilfefunktionen im Emacs genutzt werden können.

Ich empfehle dringend, dass Sie etwas Zeit darauf verwenden, das Tutorial durchzugehen, wenn Sie planen, sich ernsthaft mit dem Emacs zu beschäftigen. Wie in der Tabelle oben dargestellt wurde, können Sie das Tutorial durch »C-h t« aufrufen. Das Tutorial ist selbsterklärend und zielt auf Leute ab, die gerade mit dem Emacs anfangen.

Wenn Sie den Emacs im X laufen haben, werden Sie sehen, dass das Menue ganz rechts auf der Menuezeile »Help« heißt. Wenn Sie das Hilfemenue erforschen, beachten Sie, dass einige Unterpunkte Tastenkürzel haben und diese genau in dem Menue aufgelistet werden.

Und um schließlich das mit dem Emacs verfügbare Volumen an Dokumentation einsehen zu können, sollten Sie »M-x info« oder »C-h i« ausprobieren; dadurch wird Info aufgerufen, der Emacs Dokumentationsbrowser.

3 Emacs-Modi

Emacs-Modi sind unterschiedliche Verhaltensformen und Charakteristika, die für unterschiedliche Zwecke an- und ausgeschaltet (oder natürlich auch angepasst) werden können. Es sind Modi, die einen Editor (Emacs) gleichermaßen gut verwendbar machen, sowohl Dokumentationen zu schreiben, in einer Vielzahl von Sprachen (C, C++, Perl, Python, Java und viele mehr) zu programmieren, eine Homepage zu erstellen, E-Mails zu verschicken, Usenet News zu lesen, über die eigenen Termine auf dem Laufenden zu bleiben und sogar Spiele zu spielen.

Emacs-Modi sind einfach Lisp-Code-Bibliotheken, die den Emacs auf irgendeine Art und Weise erweitern, verändern oder optimieren.

3.1 Haupt- vs. Untermodi

Es gibt im Wesentlichen zwei Moditypen: Haupt- und Untermodi (Major und Minor Modes). Der Unterschied ist nicht ganz einfach zu begreifen, wenn man noch nicht hin und wieder mit einigen von ihnen gearbeitet hat, aber versuchen wir es mal.

Zu einer bestimmten Zeit kann nur ein Hauptmodus aktiv sein. Es können aber viele Untermodi zu einer bestimmten Zeit aktiv sein. Hauptmodi tendieren dazu, sprach- oder funktionsspezifisch zu sein, während es sich bei Untermodi um kleinere und weniger spezifische Einrichtungen handelt, die funktionsübergreifend sind.

Klingt ein bisschen abstrakt, versuchen wir also ein Beispiel. Es gibt einen Modus, den ich ziemlich oft benutze, um ganz normale Textdateien zu schreiben. Er heißt »text-mode«. Dieser Modus wurde zum Schreiben von freiem, unformatiertem Text entwickelt, wie einer README Datei. Er kann Wörter und Absätze identifizieren und stellt im Allgemeinen sicher, dass das passiert, was ich erwarte, wenn ich die normalen Navigationstasten verwende.

Wenn ich einen Text als Lesefutter für den menschlichen Konsum schreibe, will ich natürlich, dass er gut aussieht. Er sollte einen passenden Zeilenumbruch haben - nach einer angemessenen Anzahl von Zeichen und so weiter. Um Zeilenumbrüche zu ermöglichen, rufe ich einfach den Untermodus »auto-fill« auf. Dieser Untermodus versucht, das Richtige zu tun, während ich vor mich hintippe und das Ende der Zeile erreiche. Die Tatsache, dass es ein Untermodus ist, bedeutet, dass er mit einigen unterschiedlichen Hauptmodi arbeiten kann. Meine Vorstellung davon, was das Richtige ist, wenn ich am Zeilenende bin, ist eine andere, wenn ich in einem »text-mode« bin oder zum Beispiel im »java-mode«. Ich will nicht, dass mein Java-Code umgebrochen wird, als wäre er ein englischer Text. Aber ich will meine Kommentarblöcke in meinem Java-Code umgebrochen haben. Der »auto-fill« Modus ist smart genug, das herauszukriegen

Die Entwickler verschiedener Emacs-Modi haben gute Arbeit geleistet, in dem sie sicher gestellt haben, dass Dinge, die als Untermodi arbeiten sollten, auch Untermodi sind.

Wenn Sie sich noch einmal die ASCII-Skizze eines Emacs-Bildschirmes ansehen, werden Sie feststellen, dass die Moduszeile den/die Modus/Modi anzeigt in dem/denen der Emacs sich befindet. In diesem Fall hieß der Modus »Lisp Interaction«, der der Default-Modus ist. Er ist wirklich nur sinnvoll, wenn man Lisp-Code schreiben will. Aber da fast alles vom Emacs in Lisp geschrieben ist, warum nicht?

3.2 Programmier-Modi

Zuallererst: der Emacs wurde von einem Programmierer für Programmierer entwickelt. Es gibt hochwertige Modi für fast jede populäre Programmiersprache, die man sich denken kann (und sogar einige nicht ganz so populäre). Ich beschreibe nur wenige von ihnen kurz hier.

Die meisten Programmier-Modi teilen einige gemeinsame Charakteristika. Normalerweise machen sie das Folgende ganz oder teilweise:

- stellen eine farbliche Hervorhebung der Syntax für die Sprache zur Verfügung,
- stellen automatischen Texteinzug und Code-Formatierung für die Sprache zur Verfügung,
- verfügen über eine kontext- (sprach-)sensitive Hilfefunktion,
- verfügen über ein automatisches Interface mit Ihrem Debugger,
- ergänzen die Menuezeile durch sprachspezifische Menues.

Zusätzlich gibt es einige nicht-sprachspezifische Modi, die Funktionen unterstützen, die beim Programmieren in vielen Sprachen häufig vorkommen: Dinge wie eine Schnittstelle zu Ihrer Versionskontrollsoftware, automatisches Anfügen von Kommentaren an Ihren Code, Erstellen von Makefiles, Aktualisieren von Change Logs und so weiter.

Wenn man all diese Modi zusammen nimmt und die Reife und Stabilität des Emacs-Codes berücksichtigt, fällt der Vergleich mit kommerziellen Integrated Development Environments (IDEs) für Sprachen wie C++ und Java doch ziemlich gut aus. Und er ist natürlich frei verfügbar.

3.2.1 C/C++/Java

Da die Syntax von C, C++ und Java ziemlich ähnlich ist, gibt es einen Emacs-Modus für alle drei Sprachen (aber auch für Objective-C und IDL). Es ist ein sehr ausgereiftes und vollständiges Paket und in der Emacs-Distribution enthalten. Dieser Modus wird entweder »cc-mode« oder »CC Mode« genannt.

Für weitere Details oder den Download einer neueren Version, besuchen Sie

```
http://www.python.org/emacs/
```

3.2.2 Perl

Es gibt eigentlich zwei Modi, um Perl-Code im Emacs zu editieren. Der erste heißt »perl-mode« (wie wohl zu erwarten war) und der zweite »cperl-mode«. Ich kenne mich in dieser Geschichte nicht gut aus und weiß nicht, warum es zwei Modi gibt (steht nicht in der Doku), aber es scheint so, als ob »perl-mode« der Original-Modus war, um Perl-Code im Emacs zu editieren. Er scheint über weniger Merkmale als »cperl-mode« zu verfügen und es mangelt ihm an der Fähigkeit, einige von Perls ausgefalleneren Sprachkonstrukten zu erkennen.

Ich persönlich benutze und empfehle »cperl-mode«, der anscheinend ziemlich aktiv gepflegt wird und genau all die Charakteristika hat, die ich mir überhaupt nur wünschen kann. Die neueste Version findet man hier:

```
ftp://ftp.math.ohio-state.edu/pub/users/ilya/emacs
```

Aber nehmen Sie mich nicht beim Wort. Probieren Sie beide aus und nehmen Sie den, der am meisten Ihren Bedürfnissen entspricht.

3.2.3 Python

Für Python (eine weitere sehr populäre Skript-Sprache) gibt es auch einen Emacs-Modus. So weit ich weiß, ist er nicht in der GNU Emacs, sondern in der XEmacs-Distribution enthalten. Er funktioniert aber recht gut in beiden Editoren.

Sie können den »python-mode« von der offiziellen Python-Website beziehen:

```
http://www.python.org/emacs/python-mode/
```

3.2.4 Weitere

Es gibt viele, viele andere Editier-Modi zur Unterstützung von Programmierern. Diese Modi helfen bei Dingen wie:

- Shell Skripte (`bash`, `sh`, `ksh`, `csh` usw)
- `awk`, `sed`, `tcl` usw.
- Makefiles
- Change Logs
- Dokumentation
- Debugging

und vielen weiteren Dingen. Wenn Sie mehr Informationen dazu möchten, wie man an Modi und Add-Ins kommt, gehen Sie zum letzten Abschnitt dieses Dokuments.

3.3 Arbeiten als Autor

Stellen Sie sich vor, Emacs-Modi sind *nicht* nur auf die beschränkt, die Code schreiben. Leute die Dokumentationen jeglicher Art schreiben, können auch von einer breiten Auswahl an Emacs-Modi profitieren.

3.3.1 Rechtschreibprüfung (»ispell« Modus)

Autoren vieler Arten von Dokumenten brauchen immer wieder mal eine Hilfe bei der Rechtschreibprüfung. Wenn sie GNU `ispell` installiert haben, können Sie »M-x ispell« eintippen und den aktuellen Puffer auf seine Rechtschreibung hin überprüfen lassen. Wenn `ispell` Wörter findet, die er nicht kennt, bekommt man eine Liste möglicher Varianten und kann sich eine oder keine aussuchen. Die Funktion entspricht in etwa der Rechtschreibkorrektur vieler anderer nicht kostenloser Softwarepakete.

3.3.2 HTML (»html-helper-mode«)

Wenn Sie immer wieder mal oder sogar oft HTML-Dateien schreiben, möchten Sie vielleicht den `html-helper-mode` ausprobieren. Man bekommt ihn unter

```
http://www.santafe.edu/~nelson/tools/
```

wie auch die Dokumentation und was damit zu tun hat.

Wie der Namen schon andeutet, kann der `html-helper-mode` viel für die Leute tun, die noch HTML mit der Hand schreiben, auf die altmodische Art.

3.3.3 TeX (»tex-mode«)

Wenn man Dokumente in TeX schreibt, ist es oft hilfreich, sich den Emacs zu besorgen, um etwas Farbe hinzuzufügen, um die Backslashes, Klammern und andere Buchstaben hervorzuheben. Der `tex-mode` macht das für Sie.

Obwohl ich nicht mehr viel direkt in TeX schreibe, hat er sich, als ich es noch tat, als ziemlich hilfreich dabei erwiesen, meinen TeX-Code etwas lesbarer zu machen.

3.3.4 SGML (»sgml-mode«)

Das Dokument, das sie gerade lesen, wurde in SGML geschrieben und in das Format konvertiert, in dem sie es lesen. Der `sgml-mode` hat alle Grundlagen für SGML Dokumente: Validierung, Hervorhebung, Forward-Tag, Backward-Tag, und vieles mehr. Er ist ein Standardbestandteil des Emacs.

3.4 Andere Modi

Natürlich gibt es noch viele andere praktische Modi, die das Leben einfacher machen. Hier folgt nur eine kleine Auswahl der populären:

3.4.1 Versionskontrolle (»vc« Modus)

Der vc Modus hat Schnittstellen zu den meisten der populären Versionskontrollsysteme (RCS, SCCS, CVS). Dies macht es sehr leicht, Dateien ein- und auszuchecken, Releases zu managen etc. Er ist ein Standardbestandteil des Emacs und wird in der Emacs-Dokumentation beschrieben.

3.4.2 Shell Modus (»shell«)

Warum zu einem anderen X Window Fenster oder einer virtuellen Konsole wechseln, nur um ein paar Shell-Kommandos auszuführen? Machen Sie es vom Emacs aus und ersparen Sie sich die Mühe :-).

»M-x shell« ruft eine Shell innerhalb eines Emacs-Puffers auf. Mit diesem Puffer kann man die meisten Sachen machen, die man auch mit einem normalen Shell tun könnte (außer dem Betrieb von Programmen, die den ganzen Bildschirm einnehmen, wie vi oder pine), weil der Emacs hinter den Kulissen mit der wirkichen Shell kommuniziert.

Auch dies ist ein Standardbestandteil des Emacs, so dass man ihn in der Emacs-Dokumentation dokumentiert findet.

3.4.3 Telnet und FTP (»telnet« und »ftp« Modi)

Warum zu einem anderen X Window Fenster oder einer virtuellen Konsole wechseln, nur um Telnet und FTP auszuführen? Machen Sie es vom Emacs aus und ersparen Sie sich die Mühe.(Erkennen Sie schon das Muster ;-)?

Genau wie eine Shell im Emacs zu betreiben, kann man von ihm aus mit Telnet und FTP arbeiten. Versuchen Sie »M-x telnet« oder »M-x ftp«, um es selbst auszuprobieren. Lesen Sie die Dokumentation wegen all der blutrünstigen Details.

3.4.4 Handbuch (»man« Modus)

Warum zu einem anderen X Window Fenster oder einer virtuellen Konsole wechseln, nur um eine Manual Page zu lesen? Machen Sie es vom Emacs aus und ersparen Sie sich die Mühe. Ich verspreche, ich höre jetzt auf.

Genau wie beim Betreiben der Shell innerhalb des Emacs, kann man vom Emacs aus Manual Pages lesen. Versuchen Sie »M-x man« um es selbst auszuprobieren. Lesen Sie die Dokumentation, wenn Sie weitere Informationen möchten.

3.4.5 Ange-FTP

Um die ange-ftp Dokumentation zu zitieren:

> Dieses Paket will den Zugang zu Dateien und Verzeichnissen für die Nutzung von FTP vom GNU Emacs aus so einfach und transparent wie möglich machen. Ein Teil der gebräuchlichen Dateiarbeitsvorgänge ist für die Interaktion mit FTP erweitert worden.

Dies bedeutet, dass man Dateien auf weit entfernten Rechnern behandeln kann, als wären sie lokal. Will man also eine Datei auf einem anderen Computer editieren, muss man nur den Emacs anweisen, sie zu öffnen (in dem man eine etwas andere Pfadsyntax benutzt) und er kümmert sich um alle Details beim Login und dem Holen der Datei. Dann, wenn die Datei mit »C-x C-s« gespeichert wird, nimmt ange-ftp die Speicherung vor und schreibt die Datei zurück auf den weit entfernten Rechner.

Die etwas andere Pfadsyntax geht so: Eine `myfile` benannte Datei in einem `user` Verzeichnis auf einem `my.host.org` benannten Rechner kann durch »C-x f« geöffnet werden:

```
/user@my.host.org:user/myfile
```

Auch dies ist ein Standardbestandteil der Emacs-Distribution, sodass man es in der Emacs Dokumentation dokumentiert findet.

Meinen Dank an Etienne Grossman (`etienne@anonimo.isr.ist.utl.pt`) für das obige Beispiel.

4 Den Emacs anpassen

Praktisch die gesamte Emacs-Anpassung geschieht über den Lisp-Code. Man kann Variablen verändern, die Einfluss darauf nehmen, wie der Emacs arbeitet oder man kann dem Emacs neue Funktionen hinzufügen oder bereits existierende Funktionen aufheben und durch eigene ersetzen.

4.1 Temporäre Anpassung

Das Experimentieren mit der Emacs-Anpassung möchten Sie wahrscheinlich so gestalten, dass sie temporär bleibt. Wenn man was ganz schrecklich Falsches macht, braucht man nur »C-x C-c« einzugeben, um den Emacs zu schließen und wieder zu starten. Sobald Sie herausgefunden haben, welche der Veränderungen Sie gerne für immer hätten, können Sie sie zur eigenen `.emacs`-Datei hinzufügen, so dass sie bei jedem Start von Emacs geladen werden. Dies wird im nächsten Abschnitt besprochen.

4.1.1 Variablen-Zuweisung

Die einfachsten Anpassungen erreicht man durch das Ändern des Wertes einer Variablen im Emacs. Der Lisp-Code, um dies zu tun, sieht so aus:

```
(setq variable-name new-value)
```

Wobei »variable-name« der Name der Variablen ist und »new-value« der Wert, den Sie der Variablen übergeben möchten. Im Lisp-Jargon bindet man eine Variable an einen Wert. Die »setq« Funktion in Lisp entspricht den Bestimmungsoperatoren (meist »=«) in anderen Programmiersprachen.

Beachten Sie: Ich gehe hier - um der Einfachheit willen - über viele Details hinweg. Es kann auch sein, dass man mich oder andere beim Benutzen der Lisp-Funktionen »set« und sogar »setq-default« sieht. Wer wirklich neugierig ist, sollte sie einfach in einer Emacs Lisp Referenz nachsehen.

Betrachten wir eine Zeile aus meiner `.emacs`-Datei:

```
(setq-default transient-mark-mode t)
```

Die Variable »transient-mark-mode« steuert, ob eine Region hervorgehoben wird, wenn ich sie markiere oder nicht. Bei vielen GUI Applikationen wird sie invertiert oder einer anderen Farbe hervorgehoben, wenn man klickt und die Maus zieht, um einen Textbereich auszuwählen. Der Emacs macht das gleiche, wenn die Variable des »transient-mark-mode« auf einen nicht-»nil« Wert gesetzt wurde.

Was für einen Wert?

Okay. Kurzer Exkurs. Die meisten Programmiersprachen unterscheiden zwischen wahr/falsch Werten. In C/C++ wird ein Wert als wahr betrachtet, wenn er ein Wert ungleich Null ist. In Perl ist ein non-null oder non-zero Wert wahr. In Lisp, existiert die gleiche Idee, aber die Namen und Symbole sind andere.

Wahr wird gewöhnlich als »t« geschrieben und falsch (oder Null) als »nil«. Wie in anderen Sprachen, wird jedoch jeder nicht-»nil« Wert als wahr betrachtet.

Für die vollständige Beschreibung dessen, was »transient-mark-mode« tut, kann man die Online-Hilfe benutzen. Tippen Sie »C-h v« oder »M-x describe-variable« und dann »transient-mark-mode«. Wenn man faul ist wie ich, kann man sich durch Gebrauch der »Tab«-Taste die Variablennamen-Ergänzung zu Nutze machen. Tippen Sie einen Teil des Variablennamens ein und schlagen Sie die »Tab«-Taste an. Wenn genug eingetippt wurde, sodass der Emacs es eindeutig identifizieren kann, wird der vollständige Name für Sie ergänzt.

Eine weitere Variable, die oft gesetzt wird, ist »fill-column«. Sie sagt dem Emacs, wie weit der Bildschirm beim Zeilenumbruch (und der »auto-fill-mode« respektiert diesen Wert) sein sollte. Um mal einen absurden Wert einzugeben, könnte man eintippen:

```
(setq fill-column 20)
```

Aber dann passiert eigentlich nichts. Man muss den Emacs anweisen, den Ausdruck, den man eingetippt hat, auszuwerten. Um das zu tun, stellen Sie den Cursor an das Ende des Ausdrucks und tippen »C-x C-e«, dies ruft die Funktion »eval-last-sexp« auf (für den Fall, dass es Sie interessiert). Wenn Sie das tun, werden sie feststellen, dass »20« (oder welchen Wert auch immer sie genommen haben) im Mini-Puffer am unteren Rand des Bildschirms wiedergegeben wird. Dies ist nur der Rückgabewert des Ausdrucks, den Sie ausgewertet haben.

Nur um zu zeigen, dass es funktioniert - tippen Sie einen oder zwei Sätze ein. Wenn Sie den »auto-fill-mode« eingeschaltet haben (haben sie wahrscheinlich nicht), werden Sie den Zeilenumbruch nach dem 20. Zeichen erleben. Oder, wenn Sie irgend etwas eingegeben haben, tippen Sie »M-q«, es ruft die Funktion »fill-paragraph« auf. Sie wird dann den Zeilenumbruch ausführen.

4.1.2 Zuordnung von Dateinamen

Sie können den Emacs so konfigurieren, dass er automatisch etwas tut, wenn Sie eine Datei eines bestimmten Typs öffnen (so wie einige GUIs automatisch eine bestimmte Applikation starten, wenn man auf das Icon klickt). Zum Beispiel, möchte ich vielleicht, dass der Emacs jedes Mal automatisch zum Textmodus wechselt, wenn ich eine Datei mit einer `.txt`-Endung öffne. Nun, das passiert bereits :-). Sagen wir also dem Emacs, dass er in den Textmodus geht, wenn Sie eine Datei namens `README` öffnen.

```
(setq auto-mode-alist (cons '("README" . text-mode) auto-mode-alist))
```

Huh?

Ohne tief in die Lisp-Programmierung einzutauchen, die Sie wirklich nicht kennen müssen (es würde Ihnen aber nicht weh tun, sie zu lernen), lassen Sie mich einfach sagen, dass die Variable »auto-mode-alist« eine Liste von Paaren enthält. Jedes Paar enthält einen regulären Ausdruck und einen Emacs-Modus-Namen. Wenn eine Datei, die Sie öffnen, dem regulären Ausdruck entspricht (in diesem Fall, die Zeichenkette »README«), startet der Emacs den Modus, den Sie festgelegt haben.

Die merkwürdige Syntax oben ergibt sich deshalb, weil man im Grunde genommen ein weiteres Paar an die Modusliste anfügt. Sie würden nicht wollen, etwas der »auto-mode-alist« zuzuordnen, ohne sicher zu stellen, dass die Werte, die sie bereit hält, nicht verloren gehen.

Und wenn ich wollte, dass der Emacs jedes Mal automatisch zum (`html-helper-mode` wechselt, wenn ich eine Datei öffne, die auf `.html` oder `.htm` endet, würde ich folgendes meiner `.emacs` Datei hinzufügen:

```
(setq auto-mode-alist (cons '("\\.html$" . html-helper-mode) auto-mode-alist))
(setq auto-mode-alist (cons '("\\.htm$" . html-helper-mode) auto-mode-alist))
```

Die Möglichkeiten sind wirklich unbegrenzt.

4.2 Der Umgang mit der ».emacs«-Datei

Wenn man etwas Zeit mit dem Emacs verbracht hat und eine grundlegende Vorstellung davon hat, welche Vorteile eine Anpassung hätte, möchte man wahrscheinlich ein paar Dinge dauerhaft anpassen (oder mindestens so lange, bis man seine Meinung geändert hat). Wenn Sie täglich mit dem Emacs arbeiten, werden Sie auch feststellen, das die `.emacs`-Datei mit der Zeit immer größer wird. Das ist eine gute Sache, denn es bedeutet, das Sie herausgefunden haben, wie Sie den Emacs dazu bringen so zu arbeiten, wie Sie wollen. Es ist eine Schande, dass viele Softwareprodukte Sie das nicht tun lassen.

Für den Fall, das Sie es noch nicht erraten haben, jedes Mal, wenn der Emacs gestartet wird, sucht er eine `.emacs` benannte Datei in Ihrem Homeverzeichnis. Sie sollten Lisp-Code, den Sie automatisch laufen lassen möchten, in Ihre `.emacs`-Datei schreiben und das schließt die Art von Anpassung ein, die hier gerade behandelt wurde.

Ein weiteres Beispiel aus meiner `.emacs`-Datei:

```
(setq inhibit-startup-message t)
```

Die Variable »inhibit-startup-message« steuert, ob der Emacs die Willkommensmeldung anzeigt, wenn er startet. Nach einer Weile war ich es leid, sie zu betrachten (da ich wußte, wie ich Hilfe finden konnte und was weiß ich). Deshalb suchte ich nach einem Weg, sie auszuschalten.

Zur Übung versuchen Sie eine eigene `.emacs`-Datei zu erstellen und fügen Sie diese Zeile an. Dann schließen Sie den Emacs und starten ihn wieder. Die Willkommensmeldung sollte nicht mehr erscheinen.

Oft, wenn Sie etwas über einen Emacs-Modus (oder ein Paket) lesen, wird in der Dokumentation vorgeschlagen, einen Code an die eigene `.emacs`-Datei anzuhängen, um den Modus oder das Paket auf eine bestimmte Art und Weise arbeiten zu lassen.

Die GNU Emacs FAQ (»C-h F«) enthält einige Themen, die sich auf `.emacs`-Dateien beziehen und die Sie möglicherweise nützlich finden.

4.3 Das customize-Paket

Als der Emacs populärer wurde und sich kontinuierlich entwickelt hat, hat vielleicht irgendwer einmal gesagt: »es muß für Einsteiger einen besseren Weg geben, um ihren Emacs anzupassen«. Und `customize` wurde geboren.

Mit `customize` gibt es eine intuitivere Möglichkeit, Teile des Emacs anzupassen. Um es auszuprobieren, sehen Sie sich entweder das `Customize`-Untermenue in Ihrem `Help` Menue an, oder tippen »M-x customize«.

Customize unterteilt die Anpassung in sinnvolle Gruppen wie »Editieren«, »Programmieren«, »Dateien« und so weiter. Einige Gruppen enthalten Untergruppen.

Wenn Sie Veränderungen vornehmen, in dem Sie `customize` benutzen, speichert der Emacs diese Veränderungen in Ihrer `.emacs`-Datei. Das ist ziemlich praktisch, weil Sie leicht die Veränderungen einsehen und verändern können, die er für Sie gemacht hat.

Ich benutze das Customize Interface nicht, deshalb kann ich nicht viel mehr darüber sagen.

4.4 X Window

Wie jede gut erzogene X Applikation respektiert der Emacs Ihre X Ressourcen. Das bedeutet, Sie können die ursprünglichen Farben, Geometrie und andere X-spezifischen Dinge steuern, genau wie bei `xterm`, `nxterm` oder was auch immer.

Hier ist der relevante Teil meiner `~/.Xdefaults` Datei:

```
emacs*Background: DarkSlateGray
emacs*Foreground: Wheat
emacs*pointerColor: Orchid
emacs*cursorColor: Orchid
emacs*bitmapIcon: on
emacs*font: fixed
emacs.geometry: 80x25
```

Lesen Sie die Manual Page zu `X`, um mehr Einzelheiten über X Ressourcen zu erfahren.

Chris Gray (`cgray4@po-box.mcgill.ca`) stellt auch fest:

> Debian benutzt anscheinend die `~/.Xdefaults` nicht. Wie dem auch sei, Debian-Leute können das, was sie gerade eingegeben haben, in `/etc/X11/Xresources/emacs` schreiben und bekommen dann die gleichen schönen Farben, die sie bekommen hätten, wenn Sie RedHat benutzen würden.

5 Populäre Pakete

Zusätzlich zu den vielen unterschiedlichen Modi für den Emacs gibt es auch viele Zusatzpakete. Ich nenne sie *Pakete*, weil sie mehr als nur neue Modi sind. Sie beinhalten oft zusätzliche Utilities oder sind so groß, dass sie Modi zu nennen, ihnen einfach nicht gerecht wird. In anderen Fällen handelt es sich um Software, die andere Emacs-Modi und Pakete erweitert oder integriert. Der Unterschied ist nicht ganz klar, aber das ist okay.

5.1 VM (Mail)

Um die *VM FAQ* zu zitieren:

> VM (View Mail) ist ein Emacs Untersystem, das es ermöglicht, im Emacs Mails zu lesen und zu verschicken. Mit den bestehenden Befehlen kann man die normalen Dinge tun, die man von einem Mail User Agent erwartet, wie etwa Antworten schreiben, Mails in Ordnern speichern, Mails löschen und so weiter. Es gibt andere komplexere Befehle, die Aufgaben ausführen, wie das Erstellen von Digests, Mail Forwarding und der Darstellung von Mails nach verschiedenen Kriterien.

Als ich mit dem Emacs begann, habe ich mit VM eine Weile herumexperimentiert. Ich hielt es für einen tollen Ersatz für Pine, Elm oder die meisten anderen Mail-Programme. Aber ich wollte nicht unterschiedliche Programme benutzen, um Mail und News zu lesen. VM wird ständig weiterentwickelt und gut gepflegt.

Man bekommt es hier:

```
http://www.wonderworks.com/vm/
```

5.2 Gnus (Mail und News)

Um das GNUS-Handbuch zu zitieren:

> Gnus ist ein Mail-Lese-Labor. Es lässt Sie praktisch alles ansehen, als wäre es eine Newsgroup. Sie können damit Mail lesen, Sie können Verzeichnisse durchsehen, Sie können damit FTP betreiben - sie können sogar News damit lesen!
>
> Gnus versucht Menschen, die News lesen auf die gleiche Art und Weise mit Vollmachten auszustatten, wie es der Emacs mit Leuten versucht, die Text editieren. Gnus setzt dem Anwender, in dem was er tun darf, keine Grenzen. Anwender werden ermutigt, Gnus zu erweitern, damit es so funktioniert, wie sie es möchten. Ein Programm sollte nicht Menschen kontrollieren; die Menschen sollten tun können, was sie möchten, wenn sie das Programm benutzen (oder missbrauchen).

Ich benutze z.Zt. GNUS für meine Mail und News (wie oben angedeutet). GNUS wird auch aktiv weiterentwickelt und gut gepflegt.

Man bekommt es hier:

```
http://www.gnus.org/
```

5.3 BBDB (eine Art Rolodex)

BBDB ist eine heimtückische Big Brother Datenbank, ein Rolodex-artiges Programm für den Emacs, das mit den meisten der populären Emacs-Mail-Pakete (VM und GNUS eingeschlossen) arbeitet.

Man bekommt es hier:

```
http://bbdb.sourceforge.net
```

5.4 AucTeX (ein weiterer TeX Modus)

AucTeX ist ein weiterer Modus zum Editieren von TeX-Dateien.

Um die AucTeX Website zu zitieren:

> AucTeX ist ein erweiterbares Paket, das das Schreiben und Formatieren von TeX-Dateien für die meisten GNU Emacs Varianten unterstützt. Viele unterschiedliche Makro-Pakete werden unterstützt, einschließlich AMSTeX, LaTeX und TeXinfo.

Man bekommt es hier:

```
http://sunsite.auc.dk/auctex/
```

6 Andere Quellen

Dieser Abschnitt behandelt Bücher, Websites, Newsgruppen, Mailinglisten und andere Orte, an denen man mehr Informationen über den Emacs findet.

6.1 Bücher

Es gibt einige wenige wirklich gute Bücher zum Erlernen des Emacs. Zusätzlich zu diesen werden Sie feststellen, dass viele Linux- und Unix-Bücher auch ein oder zwei Kapitel über den Emacs (und `vi`) beinhalten.

6.1.1 Learning GNU Emacs

Autoren: Debra Cameron, Bill Rosenblatt, Eric S. Raymond

Verlag: O'Reilly & Associates

`http://www.ora.com/`

Kommentar: Dieses Buch ist wahrscheinlich das beste, wenn man anfängt. Nachdem Sie die HOWTO gelesen und die FAQ durchgesehen haben, dient dieses Buch als ein umfassendes und sehr leicht zugängliches Tutorial.

6.1.2 Writing GNU Emacs Extensions

Autor: Bob Glickstein

Verlag: O'Reilly & Associates

`http://www.ora.com/`

Kommentar: Wenn Sie mit dem Emacs eine Zeit gearbeitet und beschlossen haben, dass Sie gern Ihren eigenen Modus schreiben oder vielleicht einige kompliziertere Anpassungen ausprobieren würden, ist dies Ihr Buch. Es versucht zwar nicht Lisp zu lehren, doch enthält es eine kurze Einführung in die Sprache.

6.1.3 Programming in Emacs Lisp: An Introduction

Autor: Robert J. Chassell

Aus der README-Datei:

> Dies ist eine grundlegende Einführung in die Programmierung in Emacs Lisp für Menschen, die keine Programmierer sind und nicht unbedingt am Programmieren Interesse haben, aber ihre Computerumgebung anpassen oder erweitern wollen.

Man bekommt das vollständige Handbuch über Anonymous-FTP vom GNU FTP Server:

`ftp://prep.ai.mit.edu/gnu/emacs/`

Kommentar: Dies ist ein gutes Einführungshandbuch für Emacs Lisp - selbst wenn man kein Hochleistungsprogrammierer ist.

6.1.4 The GNU Emacs Lisp Reference Manual

Autor: Richard Stallman

Verlag: The Free Software Foundation

`http://www.fsf.org/`

Man bekommt das vollständige Handbuch über Anonymous-FTP vom GNU FTP Server:

`ftp://prep.ai.mit.edu/gnu/emacs/`

Kommentar: Dies ist das ultimative Handbuch für das Programmieren in Emacs Lisp.

6.2 Websites

6.2.1 EMACSulation

EMACSulation ist eine von Eric Marsden geschriebene Kolumne, die im Online-Magazin Linux Gazette erscheint unter:

`http://www.linuxgazette.com/`

Die aktuellste Kolumne erscheint unter:

`http://www.linuxgazette.com/issue39/marsden.html`

Links zu den vorher erschienenen Artikeln finden Sie, wenn Sie zum Ende des Artikels blättern.

6.3 Newsgruppen

Wenn Sie Ihren lokalen News Feed nach Newsgruppen mit der Zeichenkette »emacs« durchsuchen, finden Sie wahrscheinlich viele. Auf meinem Server sind es:

- `comp.emacs`
- `comp.emacs.sources`
- `gnu.emacs`
- `gnu.emacs.bug`
- `gnu.emacs.help`
- `gnu.emacs.sources`

Zwei Beispiele für deutsche Newsgruppen sind:

- `de.comp.gnu`
- `de.comp.editoren`

6.4 Mailinglisten

Es gibt eine Mailingliste für den GNU Emacs bei der Free Software Foundation. Sehen Sie sich die Website an, wenn Sie weitere Informationen möchten:

`http://mail.gnu.org/mailman/listinfo/help-gnu-emacs`

Die einzige dem Emacs gewidmete Mailingliste, die ich im Moment kenne, ist die NT-Emacs-Liste. Es ist eine Liste für Leute, die die Emacs Microsoft Windows Version benutzen. Wenn Sie mehr Informationen dazu wollen, sehen Sie sich die *NT-Emacs-FAQ* an:

`http://www.cs.washington.edu/homes/voelker/ntemacs.html`

6.5 Das Emacs-Lisp-Archiv

Aus dem Emacs-Lisp-Archiv-README:

> Die Emacs-Lisp-Archive bei `ftp.cis.ohio-state.edu` enthalten diverse Teile und Pakete des Emacs Lisp Code. Mit der Sprache Emacs Lisp kann man den von der Free Software Foundation herausgegebenen Editor GNU Emacs erweitern. Obwohl in der GNU Emacs Distribution viel Emacs Lisp Code enthalten ist, haben viele Menschen Pakete geschrieben, um mit anderen Systemen interagieren zu können, um die Programmiersprache, die sie benutzen beim Editieren besser zu unterstützen, um neue Funktionen hinzuzufügen, um das Standardverhalten des Emacs zu verändern. Der größte Teil des Archivs ist von Einzelnen geschrieben worden und über das Internet durch die `info-emacs` oder die `info-gnu-emacs` Mailingliste oder den `comp.emacs`, `gnu.emacs` oder `gnu.emacs.sources` Newsgruppen öffentlich verbreitet worden.

Auf die Archive kann über Anonymous-FTP von

`ftp://ftp.cis.ohio-state.edu/pub/emacs-lisp/`

zugegriffen werden.

Beachten Sie: Soweit ich sagen kann, verliert das Emacs-Lisp-Archiv langsam an Aktualität. Ich sehe, dass dort wenige neue (oder aktualisierte) Pakete erscheinen, obwohl ich weiß, dass es sie gibt. Sie werden an die `comp.emacs.sources` Newsgruppe gepostet. (Zögern Sie nicht, mich zu korrigieren, falls das nicht stimmt).

7 Danksagungen

Die folgenden Menschen haben zum Erfolg des Dokumentes beigetragen.

- Craig Lyons (`Craig.Lyons@compaq.com`)
- Robert Vollmert (`rvollmer@gmx.net`)
- Larry Brasfield (`larrybr@seanet.com`)
- Etienne Grossmann (`etienne@anonimo.isr.ist.utl.pt`)
- Thomas Weinell (`kf6mli@amsat.org`)

- Adam C. Finnefrock (`adam@bigbro.biophys.cornell.edu`)
- Chris Gray (`cgray4@po-box.mcgill.ca`)
- Robert J. Chassell (`bob@rattlesnake.com`)
- Isaac To (`kkto@csis.hku.hk`)
- Matteo Valsasna (`valsasna@elet.polimi.it`)
- Tijs van Bakel (`smoke@casema.net`)

Bei der deutschen Übersetzung hat mich unterstützt:

- Michael Weyrauch (`michael.weyrauch@gmx.net`)