GNU Emacs Übungsaufgaben

Emacs Workshop beim WICMP'25

2. März 2025

GNU Emacs[1] erscheint zunächst nur ein Textbearbeitungsprogramm zu sein. Jedoch kann es viel mehr und wird gerne auf einer ähnlichen Art wie eine UNIX-Shell („Schale", „Hülle" oder „Außenhaut") eingesetzt.

Die folgenden Aufgaben sollen einen eine Tour durch diese *fremde* Welt geben, um selbst einschätzen zu können ob einem Emacs gefällt, und selbst wenn nicht welche Denkweisen man als Souvenir mitnehmen könnte.

1 Vorbereitung

Wenn man auf einem (GNU/)Linux System ist, sollte man ein Paket namens „`emacs`" direkt installieren können. Die folgenden Aufgaben sind am besten auf einem GNU/Linux System zu bearbeiten!

Benutzer auf Microsoft Windows können auf `http://ftp.fau.de/gnu/emacs/windows/` einen Emacs-Installer besorgen. Nutzer des macOS Betriebssystems können Brew benutzen, oder direkt auf `https://emacsformacosx.com/` Emacs herunterladen. Auf manchen Versionen von macOS gibt es eine vorinstallierte Version von Emacs, diese ist jedoch sehr alt ($\sim$ 2006 A.D.).

Man kann nun Emacs öffnen, am besten indem man im Startmenu das „GNU Emacs" oder nur „Emacs" Icon aktiviert, oder indem man in einer Shell `emacs` startet. Dieses wird ein neues Fenster öffnen (Wenn man unbedingt Emacs in einem Terminal benutzen *muss*, kann man `emacs -nw` ausführen. Dieses ist jedoch *nicht* zu empfehlen, weil das graphische Emacs mächtiger ist).

2 Allgemeine Hinweise

Bestandteile der Benutzeroberfläche Das neue Fenster ist grob in Drei unterteilt:

1. Die graphischen Knöpfe und Menüs am oberen Ende des Bildschirms,
2. Die Textfläche in der Mitte, welche den größten Bestandteil des Bildschirms ausmacht. Diese kann gespaltet werden um mehrere Text-Puffer gleichzeitig anzuzeigen in sogenannten „Rahmen",
3. Der Eingabebereich („Echo Area") welches die letzte Zeile des Fensters einnimmt.

Tastenkürzel Obwohl Emacs viele Tastenkürzel hat, ist es auch problemlos mit der Maus bedienbar. Es ist jedoch hilfreich die Notation für „key chords" (eine Sequenz von Tastendrücken) lesen zu können:

`C-`<**Zeichen**> Drücke die Steuerung Taste (`Strg`, engl. „Control", `Ctrl`) und danach gleich „<Zeichen>". Sollte man mehrere Tasten mit einem `C-` Präfix drücken müssen, muss man die Taste nicht loslassen.

`M-`<**Zeichen**> Auf den meisten Systemen steht `M-` („Meta") für die `Alt` Taste, auf macOS Systemen sollte es die `Option` Taste sein. Wie zuvor, drücke hier die jeweilige Meta-Taste, und dann „<Zeichen>".

`S-`<**Zeichen**> Analog wie oben für die „Shift" taste.

Kombinationen Es ist nicht unüblich mehr als eine Umschalttasten auf einmal zu benutzen. Die Interpretation ist jedoch analog: `C-M-a` heißt Steuerung und Meta drücken, und dann `a`, `C-x M-:` heißt Steuerung und `x` drücken, alles loslassen und dann Meta und Doppelpunkt.

Ein besonders hilfreiches Kürzel ist `M-x` (im Menu „Edit → Execute Command"). Damit können Befehle abgesetzt werden, die Zugriff zu dem gesamten Potential von Emacs ermöglichen.

Die Häufigsten Text- und UI-Operationen Ein paar primitive Text-Operationen sind anders als gewöhnlich, aus historischen Gründen: `C-/` ist „Undo", Ausschneiden ist auf `C-w`, gebunden, Kopieren auf `M-w` und Einfügen auf `C-y`. Eine Datei wird mit `C-x C-f` geöffnet und mit `C-x C-s` gespeichert. Man kann einen anderen Puffer auswählen oder erstellen mit `C-x b`, und die Liste aller Puffer mit `C-x C-b` anzeigen lassen. Der Inhalt eines Puffers (egal ob es bereits zu einer Datei gehört oder nicht) kann mit `C-x C-w` in eine neue Datei geschrieben werden. In einem Puffer kann mit `C-s` gesucht werden und mit `M-%` kann Text ersetzt werden. Zwischen Rahmen kann man mit `C-x o` wechseln.

Populäre Pakete Mit `M-x package-install` können weitere „Pakete" heruntergeladen und installiert werden. Diese können Emacs um verschiedenste Funktionalitäten erweitern. Hier ist eine kleine Übersicht:

`evil`	Ein Emulations-Layer für Vim-Benutzereingaben.
`magit`	Eine mächtige Git Benutzeroberfläche.
`avy`	„Springe" zu beliebigen Positionen auf dem Bildschirm.
`yasnippet`	Expandiere häufige „Snippets" per Kürzel.
foo`-mode`	Editorunterstützung für eine Sprache *foo* deiner Wahl.

Mit `C-h a` kannst du nach Befehlen aus den Paketen suchen, und mit `M-x customize-group` werden alle Optionen angezeigt.

[1] `https://gnu.org/s/emacs`

3 Aufgaben zur Selbstübung

Bevor man anfängt: Führe den Befehl `emacs-version` mit `M-x` aus um zu bestimmen welche Version von Emacs man gerade benutzt. Diese Information kann beim Bearbeiten der nachfolgenden Aufgaben relevant sein!

Die Aufgaben-Abschnitte müssen *nicht* in der vorgegebenen Reihenfolge bearbeitet werden.

3.1 Emacs Einrichten

Emacs kann personalisiert indem man eine Initialisierungs-Datei schreibt. Auf einem UNIX-artigem System ist dieses meistens unter `/.emacs` oder `/.emacs.d/init.el` zu finden. Allgemein kann man

```
M-: (find-file user-init-file) RET %
```

ausführen, und dass sollte die richtige Datei öffnen. Für viele Einstellungen gibt es aber auch ein TUI-Interface.

- Mit `M-x customize-themes` kann man ein Farbschema auswählen und speichern. Such etwas heraus und schaue wie man es speichern kann!
- Mit `M-x customize-apropos` kann man nach Optionen suchen. Gebe `electric pair` ein und suche die Option um das Automatische schließen von Klammern zu aktivieren (endet in „Mode“).
- Finde deine Initialisations-Datei, und schreibe den folgenden Lisp code:
  `(keymap-global-set "C-c w" #'whitespace-mode)`
  um ein neues Kommando zu binden. Man kann mit `M-x load-file` die Datei direkt neue Laden (ohne Emacs zu schließen). Schaue was der Befehl macht.

3.2 Strukturelle Textbearbeitung

Man kann zwar die Maus und die Pfeiltasten benutzen um den Cursor (in Emacs-Sprach „Point“) zu bewegen, alternativ gibt es auch die Befehle `C-f`/`C-b` um sich ein Zeichen nach vorne (forward) oder hinten (backward) zu bewegen, wie auch `C-n`/`C-p` um eine Zeile nach unten (next) oder hoch (previous) zu springen.

- Wenn man anstatt `C-` die gleichen Tasten mit `M-` benutzt, dann bewegt man sich Wort-Weise. Versuche zu verstehen was Emacs unter einem „Wort“ versteht. Probiere verschiedene Dateien aus.
- Drückt man beide Umschalttasten (bspw. `C-M-f`) bewegt man sich um Strukturelle-Einheiten in einer Programmiersprache, bspw. um Klammerungen oder Blöcke, was von der Programmiersprache abhängt. Spiele damit, und versuche die bisherigen Befehle zu benutzen um sich möglichst effizient zu einer spezifischen Stelle in einer Datei zu navigieren.
- Über die vorherigen Befehle hinaus, welche nur Bewegung anbieten, kann man auch Text strukturell manipulieren. Hier ein paar Befehle um damit zu Spielen:

`C-M-u`	Springe *aus* einer Klammerung *heruas*
`C-M-d`	Springe *in* einer Klammerung *hinein*
`C-M-k`	Löschen die nächste Klammerung

- Mit einem „Präfix“-Befehl kann man vielen Befehlen einen numerischen Wert mitgeben. Diese gibt man ein indem Ziffern zusammen mit `C-` eingibt. Mit einem „`-`“ kann man einen negativen Präfix angeben. Schaue wie ein Präfix die vorherigen Befehle verändert.

3.3 Emacs als (Meta)Shell

(Die Folgenden aufgaben nehmen an, dass man bereits Erfahrung mit einer Shell hat und Befehle daraus kennt.) Zwar ist Emacs selbst eine Art „Shell“, aber als Bestandteil eines GNU bzw. UNIX-Systems kann man auch so Befehle absetzen.

- Man kann direkt einen Befehl ausführen indem man `M-!` benutzt. In Minibuffer wird man nach einem Befehl gefragt und kann dieses eingeben. Experimentiere damit um zu schauen wie sich die Befehle verhalten, bspw. auch wenn man ein `&` hinzufügt.
- Man kann Befehle mit dem Text in Puffern füttern mit dem `M-|` Befehl. Man kann davor Text auswählen um nur auf diesem zu operieren.

  Mit `C-u` („universal prefix“) kann man Befehlen ein *alternatives* Verhalten anfordern. Was macht `M-|` damit?
- Eine traditionelle Shell kann man mit `M-x shell` oder `M-x term` starten. Wie unterschieden sich die beiden?
- Beim Programmieren kann man oft ein Befehl mehrfach ausführen. Dazu ist `M-x compile` (und `M-x recompile`) hilfreich. Wie `M-!` startet es ein Befehl relativ zum Pfad der jetzigen Datei, schreibt aber die Ausgabe in ein `*compilation*` Puffer. Schaue was die `g` Taste in dem Puffer macht. Was passiert wenn du mit `C-x C-w` den Puffer speicherst, schließt (`C-x k`) und dann wieder öffnest (`C-x C-f`)?

3.4 Nutzen des Hillfssystems

Emacs ist ein „self-documenting“ System, dass viel Wert darauf legt sich erklären zu können. Es ist daher ein wichtiger Schritt dieses Hillfssystem kennenzulernen, damit man selbstständig an Informationen kommen kann.

- (Optional, etwas Zeitaufwendig) Mit `C-h t` startet man ein Tutorial, dass langsam und interaktiv viele Konzepte langsam einführt. Probiere es bei Gelegenheit aus.
- Mit `C-h f` kann man den Namen eines Befehls eingeben um mehr darüber herauszufinden. Finde etwas über Befehle heraus, die wir bisher benutzt haben.
- Mit `C-h k` kann man nachschlagen welcher Befehl auf welche Taste gebunden ist. Schaue auch hier bereits erwähnte Befehle nach, aber schaue auch was die Taste `a`, Pfeiltasten oder ein Mausklick macht.
- Emacs sollte mit einem Nachschlagewerk kommen, dem „Emacs Manual“. Diese sind im Info-Format geschrieben. Benutze `C-h i` um ein Index von Handbüchern aufzuschlagen, und suche das Emacs Manual.

Eine Übersicht von weiteren Hilf-Befehlen kann mit `C-h C-h` generiert werden.

3.5 Arbeiten mit Verzeichnissen

Mit einem „Major-Mode“ bestimmt Emacs wie es eine Datei interpretiert, bspw. ob man eine C oder Python Datei hat. Diese können die Anzeige auch viel mehr verändern, wie bei Verzeichnissen: Mit „Dired“ (directory editor) kann man ein Verzeichnis Anzeigen und mit diesen arbeiten.

- Öffne ein Verzeichnis deiner Wahl mit `C-x C-f`. Versuche eine Datei mit `C` zu kopieren, mit `S` ein Symlink zu erstellen und dann mit `D` zu löschen.
- Mit dem `C-x d` Befehl kann man nur Dateien anzeigen, die einem Glob-Ausdruck entsprechen. Kannst du damit nur alle Versteckten Dateien in deinem Home-Verzeichnis anzeigen?
- Mit `C-x C-q` springt man in den „Writable Dired“ mode. Jetzt kann mit Datei-Namen wie eine Text-Datei manipulieren, und die Änderungen mit `C-c C-c` umsetzen. Kannst du damit auch den Namen zweier Dateien vertauschen?