

EINFÜHRUNG IN GNU EMACS

MATTHIAS M. HÖLZL

1. KONZEPTE UND BEZEICHNUNGEN

1.1. **Tastatureingaben.** In der Dokumentation von Emacs werden folgende Konventionen für Tastatureingaben verwendet: **C-x** bedeutet, dass *Control*-Taste gehalten wird und dazu die Taste *x* gedrückt wird. Auf deutschen Tastaturen ist die *Control*-Taste oft mit *Strg* beschriftet. **M-x** bedeutet, dass die *Meta*-Taste gehalten wird und dazu die Taste *x* gedrückt wird. Auf deutschen Tastaturen ist die *Meta*-Taste oft mit *Alt* beschriftet.

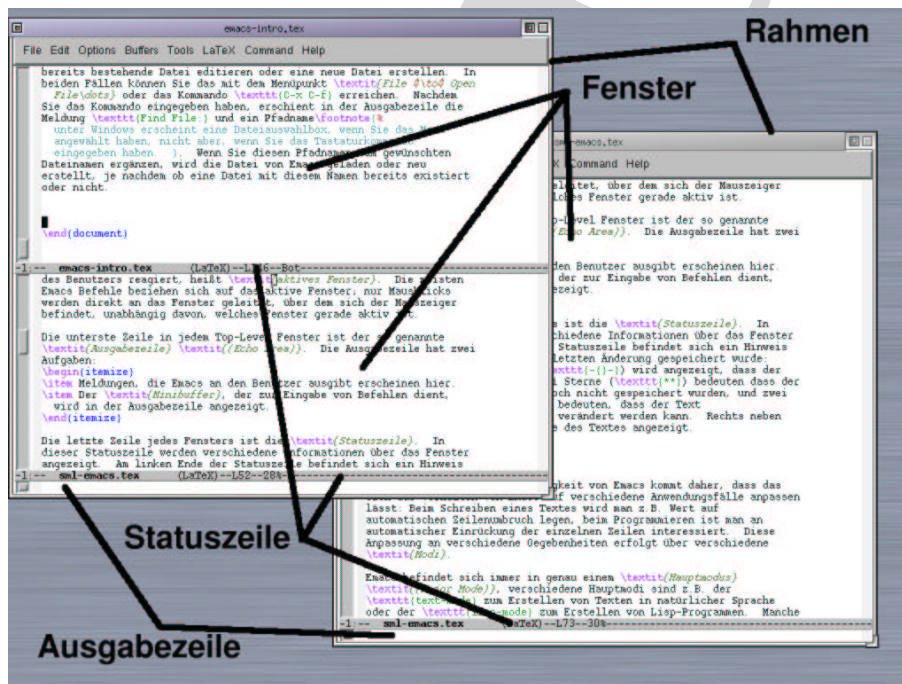


ABBILDUNG 1. Veranschaulichung der Konzepte

1.2. **Rahmen und Fenster.** Leider gibt es einige Unterschiede zwischen den Bezeichnungen, die in der Dokumentation von Emacs verwendet werden und den üblichen Bedeutungen mancher Begriffe. Da dieses Dokument Ihnen das selbstständige Studium der Emacs-Dokumentation erleichtern soll, verwende ich im Folgenden die „Emacs-typischen“ Begriffe. Ein Emacs-Prozess zeigt ein oder mehrere Fenster auf dem Bildschirm an. Diese

Fenster werden als *Top-Level Fenster* oder *Rahmen* bezeichnet. Jedes dieser *Top-Level Fenster* kann selber wieder in mehrere Teile aufgeteilt sein; jeder solcher Teil wird in der Emacs Dokumentation als *Fenster (Window)* bezeichnet. Das Fenster, das gerade auf Eingaben des Benutzers reagiert, heißt *aktives Fenster*. Die meisten Emacs Befehle beziehen sich auf das aktive Fenster; nur Mausklicks werden direkt an das Fenster geleitet, über dem sich der Mauszeiger befindet, unabhängig davon, welches Fenster gerade aktiv ist.

Die unterste Zeile in jedem Top-Level Fenster ist die so genannte *Ausgabezeile (Echo Area)*. Die Ausgabezeile hat zwei Aufgaben:

- Meldungen, die Emacs an den Benutzer ausgibt erscheinen hier.
- Der *Minibuffer*, der zur Eingabe von Befehlen dient, wird in der Ausgabezeile angezeigt.

Die letzte Zeile jedes Fensters ist die *Statuszeile*. In dieser Statuszeile werden verschiedene Informationen über das Fenster angezeigt. Am linken Ende der Statuszeile befindet sich ein Hinweis darauf, ob die Datei seit der letzten Änderung gespeichert wurde: Durch zwei Gedankenstriche (--) wird angezeigt, dass der Text nicht geändert wurde, zwei Sterne (**) bedeuten dass der Text Änderungen enthält, die noch nicht gespeichert wurden, und zwei Prozentzeichen (%) bedeuten, dass der Text schreibgeschützt ist und nicht verändert werden kann. Rechts neben dieser Markierung wird der Name des Textes angezeigt.

1.3. Modi. Ein Großteil der Leistungsfähigkeit von Emacs kommt daher, dass das sich das Verhalten von Emacs auf verschiedene Anwendungsfälle anpassen lässt: Beim Schreiben eines Textes wird man z.B. Wert auf automatischen Zeilenumbruch legen, beim Programmieren ist man an automatischer Einrückung der einzelnen Zeilen interessiert. Diese Anpassung an verschiedene Gegebenheiten erfolgt über verschiedene *Modi*.

Emacs befindet sich immer in genau einem *Hauptmodus (Major Mode)*, verschiedene Hauptmodi sind z.B. der *text-mode* zum Erstellen von Texten in natürlicher Sprache oder der *lisp-mode* zum Erstellen von Lisp-Programmen. Manche Hauptmodi verändern die Arbeitsweise auf noch deutlichere Weise: Im *info-mode* wird Emacs zu einem Hypertext-Browser und Text der im Info-Modus angezeigt wird, kann nicht verändert werden. In einem späteren Abschnitt werden wir den *SML-Modus* besprechen, der die Erstellung von SML-Programmen erleichtert.

Zusätzlich können mehrere *Untermodi* aktiv sein, z.B. der *line-number-mode*, der festlegt ob die Zeile in der sich die Eingabemarkierung befindet, in der Statuszeile angezeigt wird.

2. WICHTIGE KOMMANDOS (ODER: HILFE, ICH WILL RAUS HIER)

Gerade wenn Sie noch keine große Erfahrung mit Emacs haben, kann es vorkommen, dass Sie versehentlich in eine Situation geraten, in der Sie nicht mehr wissen, auf welche Eingabe Emacs wartet. In solchen Fällen können Sie durch dreimaliges Drücken der *Esc*-Taste Emacs wieder in einen definierten Grundzustand zurücksetzen ohne Daten zu verlieren.

Falls Sie eine Aktion begonnen haben, diese Aktion aber nicht beenden wollen, können Sie mit *C-g* den gerade aktiven Bearbeitungsvorgang abbrechen.

Ein Kommando ist wichtig, um den Tippaufwand zu reduzieren: Wenn Emacs nach einem Namen fragt, so lässt sich ein eingetipptes Anfangsstück des Namens mit der Tabulator-Taste vervollständigen.

3. EMACS STARTEN UND BEENDEN

Auf Rechnern mit dem Betriebssystem Unix oder Linux und einer graphischen Oberfläche lässt sich GNU Emacs in der Regel durch Eingabe von `emacs&` in ein Terminalfenster oder durch einen Eintrag im Startmenü starten. Im CIP-Pool können Sie eine aktuelle Version von GNU Emacs mit dem Befehl `emacs-21.2&` aufrufen, das Kommando `emacs&` startet hier eine ältere Version von Emacs.

Unter Windows existiert entweder ein Icon auf dem Desktop oder ein Eintrag im Startmenü. Wenn Sie Emacs beenden wollen, können Sie entweder mit der Maus den Menüpunkt *File* → *Exit Emacs* anwählen oder das Kommando `C-x C-c` eingeben. Falls Sie Dateien geändert aber nicht gespeichert haben erfolgt eine Abfrage ob Sie sicher sind, dass Sie Emacs verlassen wollen, andernfalls wird GNU Emacs ohne weitere Abfrage beendet.

4. ARBEITEN MIT DATEIEN UND BUFFER

Wenn Sie Emacs gestartet haben, wollen Sie in der Regel entweder eine bereits bestehende Datei editieren oder eine neue Datei erstellen. In beiden Fällen können Sie das mit dem Menüpunkt *File* → *Open File...* oder das Kommando `C-x C-f` erreichen. Nachdem Sie das Kommando eingegeben haben, erscheint in der Ausgabezeile die Meldung `Find File:` und ein Pfadname¹. Wenn Sie diesen Pfadnamen zum gewünschten Dateinamen ergänzen, wird die Datei von Emacs geladen oder neu erstellt, je nachdem ob eine Datei mit diesem Namen bereits existiert oder nicht.

Wenn Sie eine Datei in Emacs (oder in fast jedem anderen Texteditor) bearbeiten, so werden Ihre Änderungen nicht direkt auf die Festplatte geschrieben, sondern es wird eine Kopie der Datei im Hauptspeicher bearbeitet. Diese Kopie bezeichnet man auch als *Buffer*. Sie bearbeiten in Emacs also nie eine Datei direkt, sondern immer einen Buffer, in den der Inhalt der Datei kopiert wurde. Damit die Änderungen, die Sie im Buffer durchführen, nicht nach dem Beenden von Emacs wieder verloren gehen, müssen Sie den Buffer speichern. Das Kommando zum Speichern eines einzelnen Buffers ist `C-x C-s`.

Wenn Sie mehrere Dateien laden, so legt Emacs mehrere Buffer an. In jedem Fenster ist einer dieser Buffer sichtbar (es können auch mehrere Fenster den gleichen Buffer anzeigen). Um den vom aktuellen Fenster angezeigten Buffer zu wechseln können Sie den Befehl `C-x b` verwenden: Wenn Sie diesen Befehl eingeben, fragt Emacs in der Ausgabezeile nach dem Namen des Buffers, der im aktuellen Fenster angezeigt werden soll. Wenn Sie den Namen eines existierenden Buffers im Minibuffer (in der Ausgabezeile) eintippen und die Eingabe mit der Eingabetaste beenden, wechselt der angezeigte Buffer. (Verwenden Sie die Tabulator-Taste, um den Namen zu vervollständigen!) Eine Auflistung aller Buffer

¹unter Windows erscheint eine Dateiauswahlbox, wenn Sie das Menü angewählt haben, nicht aber, wenn Sie das Tastaturkommando eingeben haben.

erhalten Sie mit dem Befehl `C-x C-b`. Diese Liste ist Maussensitiv, Sie können also einen Buffer auswählen, indem Sie ihn mit der mittleren Maustaste anklicken.

Das Kommando zum Speichern aller geänderten Buffer ist `C-x s`.

Insbesondere beim Programmieren ist es oft zweckmäßig zwei Buffer gleichzeitig anzuzeigen. Dazu können Sie ein Fenster mit dem Befehl `C-x 2` in zwei Fenster aufteilen. Beide Fenster zeigen den gleichen Buffer an, mit den oben beschriebenen Kommandos können Sie die angezeigten Buffer wechseln.

Mit dem Befehl `C-x o` können Sie zwischen den angezeigten Fenstern wechseln. Mit `C-x 1` schließen Sie alle Fenster außer dem gerade aktiven.

5. ERSTELLEN VON SML-PROGRAMMEN

In diesem Abschnitt soll an einem Beispiel erläutert werden, wie Sie SML-Programme mit Emacs erstellen können. Dabei wird davon ausgegangen, dass der SML-Modus für Emacs installiert ist und ein lauffähiger SML-Compiler zur Verfügung steht.

5.1. Vorbereitende Schritte im CIP-Pool. Im CIP-Pool müssen Sie Emacs so konfigurieren, dass der SML-Modus zur Verfügung gestellt wird. Erzeugen Sie dazu eine Datei `~/.emacs2` mit folgendem Inhalt:

```
(autoload 'sml-mode "sml-mode" "Major mode for editing SML." t)
(setq auto-mode-alist
  (append '(("\.sml$" . sml-mode)
            ("\.sig$" . sml-mode)
            ("\.ML$" . sml-mode)) auto-mode-alist))
```

Diese Datei kann auch von der Homepage der Vorlesung (die unter der URL <http://www.pst.informatik.uni-muenchen.de/lehre/WS0203/infoI/> erreichbar ist) heruntergeladen werden.

5.2. Erstellen eines SML-Programms. Mit dem interaktiven SML-Compiler können Sie ein Programm direkt in den Compiler einzutippen und ausführen. Allerdings ist das Programm verloren, sobald Sie den Compiler beenden. Um das Programm dauerhaft verfügbar zu haben müssen Sie es in einer Datei speichern, und diese Datei in den Compiler laden.

Emacs klassifiziert Dateien nach ihrer Dateiendung. Alle Dateien, die mit `.sml` (oder `.sig` oder `.ML`) enden, werden als SML-Datien erkannt. Daher sollten Sie SML-Programme immer in Dateien mit der Endung `.sml` speichern.

Erzeugen Sie in Emacs eine Datei `test.sml` in Ihrem Home-Verzeichnis (durch Eingabe von `C-x C-f test.sml`). Fügen Sie die folgende Zeile in die Datei `test.sml` ein:

```
fun fact n = if n = 0 then 0 else n * fact (n - 1);;
```

²Das ist eine Datei mit Namen `.emacs` in Ihrem Home-Verzeichnis, d.h., der Name dieser Datei beginnt mit einem Punkt. Derartige Dateien werden vom normalen `ls`-Kommando nicht angezeigt. Deshalb werden solche Dateinamen oft für Konfigurationsdateien verwendet. Mit dem Befehl `ls -A` werden auch derartige „versteckte“ Dateien angezeigt.

Diese Zeile ist ein (fehlerhafter) Versuch eine Fakultätsfunktion zu implementieren. Beachten Sie, dass die Funktionsdefinition mit zwei Strichpunkten abgeschlossen wird. Das ist eine Konvention, die es Emacs erleichtert, Programme korrekt einzurücken. Der SML-Compiler ignoriert den zweiten Strichpunkt. Speichern Sie den Buffer (**C-x C-s**).

Um die Fakultätsfunktion zu testen müssen Sie sie in den SML-Compiler laden und mit einigen Werten überprüfen. Dazu können Sie in Emacs einen *Subprozess* starten: Geben Sie **M-x run-sml** ein. Emacs fragt dann in der Ausgabezeile nach dem SML-Compiler. Hier ist der Default-Wert `sml` bereits richtig, Sie können also einfach die Eingabetaste drücken. Emacs teilt den Top-Level-Frame in zwei Fenster auf: Im oberen Fenster sehen Sie Ihr Programm, im unteren Fenster können Sie mit dem SML-Compiler interagieren.

Um Ihr Programm in den SML-Compiler zu laden, können Sie das Kommando **C-c C-l** eingeben. Daraufhin fragt Emacs nach dem Namen der Datei, die geladen werden soll, und lädt diese Datei in das SML-System. (Sie können natürlich auch hier wieder den Dateinamen mit der Tabulator-Taste vervollständigen). Nachdem die Datei geladen wurde, wird das untere Fenster zum aktiven Fenster, so dass Sie die Funktion `fact` unmittelbar testen können. Geben Sie z.B. `fact 3`; ein und drücken Sie die Eingabetaste. Der SML-Compiler gibt Ihnen daraufhin

```
val it = 0 : int
```

zurück. Das ist natürlich nicht der korrekte Wert für die Fakultät von 3.

Wir müssen jetzt also den Fehler im Programm beseitigen. Wechseln Sie dazu (mit **C-x o** wieder in den Buffer `test.sml`, gehen Sie (mit den Cursorstasten) auf das zweite Vorkommen der 0 (nach dem Schlüsselwort `then`), löschen Sie die 0 und fügen Sie eine 1 ein. Ihre Datei sieht jetzt also folgendermaßen aus:

```
fun fact n = if n = 0 then 1 else n * fact (n - 1);;
```

Speichern Sie die Datei (mit **C-x C-s**) und laden Sie die geänderte Datei wieder in SML (mit **C-c C-l**). Um jetzt wieder `fact 3`; zu evaluieren, können Sie mit der Taste **M-p** die bisher getätigten Eingaben durchlaufen.

Wechseln Sie jetzt wieder in den Buffer `test.sml`. Um Ihnen die automatische Einrückung des SML-Modus zu demonstrieren, definieren wir die folgende Funktion:

```
fun fact_iter n acc = if n = 0
  then acc
  else fact_iter (n-1) (n*acc);;
```

Geben Sie dazu die erste Zeile ein (bis `n = 0`) und drücken Sie die Eingabetaste. Sie sind jetzt am Anfang der zweiten Zeile. Geben Sie `then` ein und drücken Sie die Tabulatortaste. Die Zeile wird so eingerückt, dass das `then` unter dem `if` steht. Wenn Sie die zweite Zeile eingetippt haben, können Sie mit der gleichen Methode die dritte Zeile einrücken.

Wenn die Auswertung eines Programms zu lange dauert, können Sie es beenden, indem Sie in den `*sml*`-Buffer wechseln und das Kommando **C-c C-c** eingeben. Wenn Sie z.B. das folgende (nicht terminierende) Programm eingeben

```
fun f x:int = f x;;
```

und versuchen `f 1` auszuwerten, so reagiert der SML-Compiler nicht mehr auf weitere Eingaben, da er sich in einer Endlosschleife befindet. Mit `C-c` `C-c` können Sie diese Schleife unterbrechen. Der Compiler gibt die Meldung `Interrupt` aus, und Sie erhalten wieder eine Eingabeaufforderung.

Sie können SML beenden, indem Sie in den `*sml*`-Buffer wechseln und `C-d` `C-d` eingeben.

ANHANG A. INSTALLATION VON SML UNTER LINUX

A.1. Installation von SML of New Jersey. Zur Installation von `smlnj` müssen Sie die Pakete zur Programmentwicklung mit C (z.B. `gcc` und `binutils`) installiert haben.

- Erzeugen Sie ein Verzeichnis `/prog/sml/smlnj` und wechseln Sie in dieses Verzeichnis

```
$ mkdir -p ~/prog/sml/smlnj
$ cd ~/prog/sml/smlnj
```

- Laden Sie von `ftp://ftp.research.bell-labs.com/dist/smlnj/working/110.42` die Dateien `config.tgz`, `boot.x86-unix.tgz`, `runtime.tgz`, `ml-lex.tgz`, `ml-yacc.tgz`, `smlnj-lib.tgz` und `cml.tgz` in das Verzeichnis `prog/sml/smlnj` herunter

```
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/config.tgz
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/boot.x86-unix.tgz
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/runtime.tgz
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/ml-lex.tgz
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/ml-yacc.tgz
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/smlnj-lib.tgz
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
dist/smlnj/working/110.42/cml.tgz
```

(Geben Sie die zwei nach einem Dollarzeichen stehenden Zeilen als eine lange Zeile, ohne Zeilenumbruch oder Leerzeichen ein.)

- Entpacken Sie die Datei `config.tgz`

```
$ tar xvzf config.tgz
```

- Starten Sie das Installationskript

```
$ ./config/install.sh
```

Das Installationskript erzeugt den SML-Compiler. Das dauert einige Zeit und es werden einige Warnungen ausgegeben; es sollten aber keine Fehlermeldungen auftreten. Wenn das Installationskript erfolgreich beendet wurde, sind die letzten Zeilen der Ausgabe vermutlich

```
./config/install.sh: Libraries compiled successfully.
./config/install.sh: Moving libraries to /home/tc/prog/sml/smlnj/lib.
```

- Erstellen Sie einen symbolischen Link auf das neu erstellte sml-Programm

```
$ su
```

Geben Sie das root-Passwort ein.

```
# ln -s 'pwd' /bin/sml /usr/local/bin/sml
# exit
```

Die „Anführungszeichen“ in dem vorhergehenden Befehl erhalten Sie auf einer deutschen Tastatur, indem Sie die Taste rechts neben dem schafen S bei gehaltener Umschalttaste (Shift-Taste) drücken.

- Jetzt sollten Sie mit

```
$ sml
```

den SML-Compiler starten können. Damit kann auch Emacs im SML-Modus mit dem SML-Compiler kommunizieren.

A.2. Installation des SML-Modus für Emacs.

- Erzeugen Sie ein Verzeichnis `~/prog/sml/downloads`:

```
$ mkdir -p ~/prog/sml/downloads
```

- Laden Sie von `ftp://ftp.research.bell-labs.com/dist/smlnj/contrib/emacs` die Datei `sml-mode-3.9.5.tar.gz` herunter und speichern Sie sie in dem Verzeichnis `/prog/sml/downloads`. (Sie können dazu Mozilla oder Konqueror verwenden, oder z.B. die Befehle

```
$ cd ~/prog/sml/downloads
$ wget --passive-ftp ftp://ftp.research.bell-labs.com/
  dist/smlnj/contrib/emacs/sml-mode-3.9.5.tar.gz
```

(ohne Zeilenumbruch)

- Entpacken Sie die heruntergeladene Datei

```
$ cd ~/prog/sml/
$ tar xvzf downloads/sml-mode-3.9.5.tar.gz
```

- Wechseln Sie in das erstellte Verzeichnis, und führen Sie `make install` aus

```
$ cd cd sml-mode-3.9.5
$ make install
```

- Fügen Sie den folgenden Text zur Datei `~/.emacs` hinzu (oder erzeugen Sie die Datei mit diesem Inhalt neu, falls sie nicht existiert):

```
(autoload 'sml-mode "sml-mode" "Major mode for editing SML." t)
(setq auto-mode-alist
  (append '(("\\.sml$" . sml-mode)
            ("\\.sig$" . sml-mode)
            ("\\.ML$" . sml-mode)) auto-mode-alist))
```