Ein LaTeX-Schnellkurs

erstellt am 25. Mai 2003 und. . .

. . .aktualisiert am 16. August 2005 von

COBRA

Inhaltsverzeichnis

1 Was ist LaTeX (und was nicht)? 1

2 Die LaTeX-Philosophie 1

3 Sinn dieses Schnellkurses 2

4 A brief history of LaTeX 2

5 Distributionen 2

6 Installation 2

6.1 Linux . . . 3
6.2 Windows . . . 3
6.3 Mac OS X . . . 3

7 Dokumentstruktur 3

8 Gliederung und Formatierung 4

8.1 Sprache . . . 4
8.2 Schriften, Formeln und Symbole . . . 5
8.3 Abbildungen . . . 5
8.4 Tabellen . . . 7
8.5 Zitate, Crosslinks und Hyperrefs . . . 7
8.6 Inhaltsverzeichnis und Index . . . 8
8.7 Farbe . . . 8

9 Typografische Feinheiten 8

10 Fazit 9

Anhang: Schritt-für-Schritt 9

Danksagung 11

Literatur 13

1 Was ist LaTeX (und was nicht)?

LaTeX ist ein professionelles Satzprogramm, das insbesondere im Buchdruck und für wissenschaftliche Veröffentlichungen eingesetzt wird. Genausogut eignet es sich aber auch zum Erstellen von Serienbriefen oder Präsentationen, dem Satz von Partituren, der Darstellung von Schachproblemen, und tausend anderen Dingen mehr. Es ist für alle nur denkbaren Plattformen, vom Atari bis zur Cray, frei erhältlich, und ist überdies weitgehend plattformunabhängig: ein auf dem Atari erstelltes LaTeX-Dokument erzeugt auf der Cray identischen Output.

LaTeX ist kein Textverarbeitungsprogramm, und arbeitet nicht nach dem Prinzip des WYSIWYG (**w**hat **y**ou **s**ee **i**s **w**hat **y**ou **g**et). Das Erstellen von Dokumenten mit LaTeX ähnelt vielmehr stark der Tätigkeit des Programmierens, und erfordert daher vom Benutzer einen höheren Grad an Abstraktionsfähigkeit als beim Tippen eines Textes mit einer Textverarbeitung. Diesem Nachteil steht allerdings die buchstäblich druckreife Qualität der mit LaTeX erstellten Dokumente gegenüber. Im Gegensatz zu Textverarbeitungen beherrscht LaTeX z. B. automatisches Kerning, bricht absatzweise um, unterstützt qualitativ hochwertige Postscriptgrafiken, und erlaubt ohne Zusatzprogramme die Ausgabe von PDF-Dateien mit Hyperreferenzen. Ein nicht zu unterschätzender Vorteil ist auch, daß die Facharbeit auf dem Tintendrucker zuhause genauso aussieht wie auf dem Laserdrucker im Kopierstudio oder auf der Belichtungsmaschine einer Offsetdruckerei.

2 Die LaTeX-Philosophie

Die Philosophie von LaTeX ist der einer Textverarbeitung geradezu konträr entgegen gesetzt. Aus diesem Grund haben gerade erfahrene Anwender einer solchen größere Schwierigkeiten mit LaTeX als vollständige Anfänger, die eben noch nicht vorbelastet sind. Ein Anwender von LaTeX muß die logische Struktur eines Dokuments definieren, und kann sich danach auf den Inhalt des Textes konzentrieren. Gestaltung und Satz werden von LaTeX übernommen. Manuelle Korrekturen der Formatierung können erfolgen, sollten aber immer die Ausnahme bleiben. Im Gegensatz dazu sind Nutzer einer Textverarbeitung oft vorwiegend und schon von Anfang an mit der manuellen Formatierung beschäftigt (obwohl hier moderne Textverarbeitungen auch bessere Möglichkeiten anbieten). Diese Gewohnheit muß man unter LaTeX ablegen.

Ein weiterer wichtiger Unterschied ergibt sich aus der Tatsache, daß LaTeX Dokumente absatzweise umbricht

und seitenweise setzt, und sich für mehrere Leerzeichen zwischen Worten nicht interessiert. Textverarbeitungen dagegen können Text nur zeilenweise verarbeiten, und Leerzeichen werden ebenso als Charakter interpretiert wie jeder andere Buchstabe. Das Ergebnis seiner Bemühungen sieht man bei einer Textverarbeitung instantan (WYSIWYG), während das LaTeX-Dokument zunächst in ein Format übersetzt werden muß, das auf dem Bildschirm ausgegeben werden kann. Anfänger neigen dazu, das nach Schreiben von drei Worten zu tun, Fortgeschrittenere benutzen diese Möglichkeit erst, wenn der Text im wesentlichen erstellt wurde.

3 Sinn dieses Schnellkurses

Es gibt ganz ausgezeichnete Bücher über LaTeX, wie z. B. den Kopka[1] oder den sehr aktuellen LaTeX-Companion.[2] Es ist allerdings nicht jedermanns Sache, vor dem Schreiben eines Briefs ein über 1000-seitiges Buch durchzuarbeiten (wofür ich übrigens volles Verständnis habe). Zwar gibt es auch kürzere Bücher, doch haben diese den schwerwiegenden Nachteil, daß tatsächlich nur elementares behandelt wird. LaTeX kennt über 900 Grundbefehle (man vergleiche das mit C!), aber ein Großteil der Funktionalität (wie etwa das Einbinden von Grafiken) wird nicht vom LaTeX-Kernel selbst, sondern von externen Modulen (den sogenannten Packages) bereitgestellt. Genau dieses kommt aber in allen mir bekannten Kurzeinleitungen[3] (die aber ansonsten sehr zu empfehlen sind) zu LaTeX zu kurz...

Ich habe mir daher mit dem vorliegenden Text etwas eigentlich unmögliches vorgenommen: blutigen (aber lernfähigen) Anfängern auf wenigen Seiten zu erklären, wie man LaTeX installiert und ein ansprechendes Dokument mit Abbildungen, Tabellen, Referenzen, Inhaltsverzeichnis und Index erstellt. Notgedrungen bleiben dabei 99% der Funktionalität von LaTeX unerwähnt. Da dieser Text auch als Quelltext erhältlich ist, kann er als direktes Anschauungsmaterial für einige dieser Punkte genutzt werden.

4 A brief history of LaTeX

LaTeX basiert auf dem in der Mitte der 70er-Jahre von DONALD KNUTH entwickelten Programm TeX (das heute auch oft als PlainTeX bezeichnet wird). TeX ist, wie man gewöhnlich zu sagen pflegt, sehr „mächtig“, aber auch entsetzlich kompliziert. Zum Glück für uns alle sprang Mitte der 80er-Jahre LESLIE LAMPORT in die Bresche, und entwickelte das Makropaket LaTeX. Heutzutage benutzen nur noch Gurus TeX, für uns Normalsterbliche gibt es LaTeX, seit 1994 in der Version LaTeX 2_ε. Trotzdem ist der eigentliche Compiler TeX, momentan in der Version 3.141592 (der Versionsstand konvergiert gegen π).

5 Distributionen

Man sollte nicht den Fehler machen, die Namen der vielen verfügbaren TeX-Distributionen für andere TeX-Varianten zu halten. Die Namen der im nächsten Abschnitt vorgestellten Distributionen teTeX und MikTeX leiten sich einfach aus den Namen ihrer Ersteller her, basieren natürlich aber beide auf LaTeX 2_ε. Im Grunde sind diese Distributionen ein Analogon zu Linux: der Kernel alleine (TeX/LaTeX) nutzt nicht viel, man benötigt noch alles mögliche andere, um schließlich unter KDE diesen Text zu schreiben. Im Falle der LaTeX-Distributionen sind dies Erweiterungen (z. B. pdfTeX, e-TeX, Omega), Hilfsprogramme (z. B. dvips, epstopdf), Schriftarten, die oben erwähnten Packages, und vieles andere mehr.

6 Installation

Viele Anfänger sind mächtig frustriert, nachdem sie sich eine LaTeX-Distribution installiert haben: sie finden einfach nichts zum draufklicken.☺ Das ist auch kein Wunder, denn LaTeX ist ein Konsolenprogramm. Ich fand jahrelang nichts dabei, meine Texte mit vi zu schreiben, und dann in der shell „latex text.tex“ einzugeben...aber die Zeiten haben sich geändert. Heute nutze ich sowohl unter Linux als auch unter Windows TeX-Editoren, die sich ganz zu recht in Anlehnung an die C- und Pascal-Entwicklungsumgebungen als IDE (Integrated Development Environment) für LaTeX bezeichnen. Beide bieten die wichtigsten LaTeX-Kommandos in hübschen Menüs oder Toolbars an, und integrieren eine ausgezeichnete Hilfe sowohl bezüglich LaTeX als auch des Editors selbst. Nach dem (menügestützten) Eingeben des Texts klickt man lediglich auf ein Icon, und der Text wird kompiliert und in einem der möglichen Ausgabeformate auf dem Bildschirm angezeigt.

Als mögliche Ausgabeformate kann man in jedem Fall zwischen dvi, ps, und pdf wählen. Früher galt das Device-Independent-Format dvi als besonders geeignet für die Bildschirmausgabe, während Postscript (ps) für den Druck vorgezogen wurde. Heute gibt es für alle drei Formate leistungsfähige Viewer, sodaß diese strikte Unterscheidung nicht mehr gilt. Ich persönlich verwende in letzter Zeit fast ausschließlich pdfLaTeX, da die Dateien deutlich kleiner sind und sich so leicht verschicken

lassen.

Im folgenden gehe ich kurz auf die korrekte Installation einer kompletten LaTeX-Umgebung unter Linux, Windows, und Mac OS X ein.

6.1 Linux

Hier ist es am einfachsten: Jede der gängigen Linux-Distributionen enthält die LaTeX-Distribution teTeX von Thomas Esser, die sehr vollständig, einigermaßen aktuell und gut gepflegt ist. Ebenfalls enthalten sind postscript- und pdf-Viewer wie z. B. ghostview und Xpdf. Auch Editoren mit zumindest partieller LaTeX-Unterstützung gibt es zuhauf. Selbst KWrite oder Kate bieten Syntax-Highlighting, und Emacs ist in Hinsicht LaTeX sicher nicht zu übertreffen. Für Anfänger (und nicht nur diese) besser geeignet ist allerdings Kile.[4] Alle wichtigen Compilerbefehle sowie Viewer sind über die Iconleiste, das Menü „Tools“ oder über Shortcuts (F1–F12) abrufbar. Ferner gibt es Menüs für die wichtigsten Befehle und Symbole, was bei der Vielfalt derselben eine nicht zu unterschätzende Hilfe darstellt. Schließlich lassen sich Projekte definieren und verwalten, also komplexe Dokumente, die aus mehreren Einzeldateien bestehen. Dies ist nützlich bei langen Texten wie etwa Diplomarbeiten und Dissertationen.

Die Einbindung der Compiler geschieht bei der Installation vollautomatisch. Nur eine Kleinigkeit sollte man ergänzen: in Options/Configure Kile sollte man bei der pdfLaTeX-Zeile folgendes hinzufügen: `-shell-escape`. Eine Erklärung hierzu folgt in Kürze. Es soll aber schon hier erwähnt werden, daß dieser Parameter prinzipiell eine Sicherheitslücke darstellt, da er pdfLaTeX erlaubt, beliebige Programme zu starten.

teTeX bietet kein automatisiertes Update oder ein menügeführtes Einspielen weiterer Packages, dies muß manuell erfolgen. Alle verfügbaren Packages gibt es auf CTAN,[5] man folge nach dem Download dem ReadMe der Packages. Nach dem Einspielen neuer Packages muß die Datenbank erneuert werden: in einer Shell (als root) texconfig eingeben, dann rehash auswählen.

6.2 Windows

Nicht ganz so einfach wie bei Linux, aber einfach genug. Man installiere (in exakt dieser Reihenfolge!):

1. Adobe Reader (hat wohl jeder)
2. Ghostscript[6]
3. Ghostview[7]
4. MiKTeX[8]
5. TeXnicCenter[9] oder TeXMaker[10]

MiKTeX von Christian Schenk ist in Hinsicht Aktualität, Komplettheit und Pflege teTeX ebenbürtig (als einzige LaTeX-Distribution für Windows). Bei der Installation werden drei verschiedene Installationsgrößen angeboten: minimum, large, total. Wenn man den Platz hat (500 MB), sollte man total wählen, ansonsten large.

Beim anschließenden Starten von TeXnicCenter übernimmt ein Wizard die Einbindung von MiKTeX. Man sollte nur den Pfad des DVI-Viewers angeben, der sich in `\miktex\installdir\miktex\bin` unter dem Namen yap (**Y**et **A**nother **P**reviewer) verbirgt.

TeXnicCenter bietet unter Windows ähnlichen Komfort wie Kile unter Linux. Ebenso wie mit Kile kann man Projekte definieren und verwalten. Eine Alternative ist TeXMaker, der der Vorgänger von Kile ist und sowohl unter Linux als auch unter Windows läuft.

Wie bei Kile sollte man auch hier eine Kleinigkeit ergänzen: unter „Ausgabe/Ausgabeprofile definieren bei LaTeX→PDF“als Compilerargument ein `-shell-escape` hinzufügen. Dort sollte also stehen: `-interaction=nonstopmode -shell-escape "%pm"`

MiKTeX bietet ein automatisches Update an (Upgrade Wizard, im Programmenü unter MiKTeX), das wunderbar funktioniert. Da von vielen Packages regelmäßig neue Versionen erscheinen, sollte man davon Gebrauch machen. Das Updaten der Datenbank erfolgt hierbei automatisch. Falls man doch einmal ein Package manuell installiert, führt man danach in MiKTeX-Options ein refresh database durch.

6.3 Mac OS X

Zunächst eine Warnung: Finger weg von ozTeX! Diese beliebte Distribution für den Mac ist veraltet, und ist nicht TDS-konform (TeX Directory Structure, ein Standard, der erstmals 1994! auf einer LaTeX-Entwicklerkonferenz festgelegt wurde).[11]

Da ich schon Jahre nicht mehr mit Macs arbeite, würdees mir schwerfallen, eine Anleitung für teTeX auf dem Mac zu schreiben, aber glücklicherweise gibt es diese bereits.[12] Dieser Link ist übrigens auch für Nutzer anderer Betriebssysteme interessant.

7 Dokumentstruktur

So, jetzt geht's endlich los. Wie ist ein LaTeX-Dokument strukturiert? Folgendermaßen:

```
\documentclass[optionen]{klasse}
\usepackage[optionen]{paketname}
```

```
\begin{document}

\end{document}
```

- `\documentclass[optionen]{klasse}`
  Dieses Kommando deklariert den Typ des Dokuments und legt dessen globale Formatierung fest. Die `optionen` in eckigen Klammern sind optional. Die `klasse` in geschweiften Klammern ist dagegen zwingend (das ist ein genereller Standard bei LaTeX!).

  Fangen wir mit den Klassen an. Standardmäßig gibt es hier book, report, article, letter. Der vorliegende Text z. B. benutzt die Klasse article. Allerdings hat jeder Verlag, der etwas auf sich hält, seine eigene Klasse, d. h. es gibt für Veröffentlichungen bei Springer eine springer-Klasse. Das soll uns aber nicht stören, denn normalerweise sind die verfügbaren Klassen vollkommen ausreichend. Die Standardklassen sind allerdings eher auf amerikanische Verhältnisse zugeschnitten. Für europäische und insbesondere deutsche Anwender eignen sich die Klassen des KOMA-Script-Packages (scrbook, scrreprt, scrartcl, scrlettr) eher, da sie den Regeln der europäischen Typographie folgen. Vor dem Einsatz dieses komplexen Packages sollte man aber unbedingt einen Blick in die Anleitung werfen, die unter `\doc\latex\koma-script` zu finden ist.

  Nun die Optionen. Diese sind normalerweise weniger wichtig. Dervorliegende Text nutzt zwei: a4paper, aus verständlichen Gründen, und twocolumn, um einen zweispaltigen Text zu erhalten. Erwähnenswert ist auch twoside für den Druck von zweiseitigen Texten mit asymmetrischen Rändern. Wichtig die Standardschriftgröße, die man auf 10pt, 11pt, oder 12pt festlegen kann. Läßt man diese Option weg, wird 10pt verwendet.

- `\usepackage[optionen]{paketname}`

  Die schon mehrfach erwähnten Packages, von denen es wohl tausende gibt. Ich werde in den nächsten Abschnitten einige der wichtigsten davon besprechen.

- `\begin{document}...\end{document}`

  Beinhaltet den Text des Dokuments, sowie Tabellen und Abbildungen. Tippt aber bitte jetzt nicht gleich drauflos, es kommt noch einiges wissenswertes... ☺ Für Ungeduldige habe ich im Anhang eine Schritt-für-Schritt-Anleitung für das Erstellen eines einfachen Dokumentes zusammengestellt. Man beachte allerdings: Da die wichtigsten Formatierbefehle in den Menüs sowohl von Kile als auch von TeXnicCenter zu finden sind, erspare ich mir deren detaillierte Erklärung. Bei Unklarheiten bezüglich der Funktion eines Kommandos gibt die Hilfe dieser Editoren normalerweise reichlich Auskunft. Unter einer Lesephobie darf man freilich nicht leiden...

8 Gliederung und Formatierung

Ganz elementar: eine Leerzeile interpretiert LaTeX als neuen Absatz, mehrere Lehrzeichen werden als ein einzelnes betrachtet! Eine neue Zeile kann man mit `\\` erzwingen, was man aber generell verneiden sollte. Horizontale und vertikale Abstände kann man mit `\vspace{xcm}` und `\hspace{xcm}` erzeugen, wobei das x natürlich für eine x-beliebige Zahl steht. Beides sollte man sparsam (wenn überhaupt) verwenden, denn:

LaTeX bietet viele automatisierte Methoden der Gliederung und Formatierung eines Dokuments an. Es gibt `\title`- und `\author`-Kommandos, und zahlreiche Gliederungsbefehle (von der Klasse abhängig). Für article z. B. section, subsection, subsubsection, paragraph, subparagraph. Alle diese Kommandos lassen sich mit `\commandname{Text}` benutzen. Dieser Abschnitt ist z. B. mit `\section{Gliederung}` entstanden.

Eine weitere und wichtige Klasse der Gliederung wird über die Environments eingeführt.

```
\begin{enumerate} \item bla \end{enumerate}
\begin{itemize} \item bla \end{itemize}
```

sind hierfür Beispiele.

Weitere dieser Kommandos folgen in den nächsten Abschnitten. Und um es noch einmal zu sagen: all diese Kommandos (und viele mehr) finden sich in den Menüs der hier verwendeten Editoren, und werden in deren Hilfe auch erklärt.

8.1 Sprache

Die meisten von uns wollen einen Text in Deutsch schreiben. Allerdings ist LaTeX inherent ein englischsprachiges Programm. Genau dafür gibt es Babel.

```
\usepackage[german]{babel}
```

lädt Babel mit der Option german. Wer die neue Rechtschreibung verwenden will, verwendet die Option ngerman. Schließlich sorgt einer der folgenden Kommandos

```
\usepackage[latin9]{inputenc} (Linux)
\usepackage[ansinew]{inputenc} (Windows)
\usepackage[applemac]{inputenc} (Mac)
```

dafür, daß man auch Umlaute so eingeben kann, wie es einem das Keyboard erlaubt.

Mehrsprachige Texte gelingen leicht über

```
\usepackage[german,french,english]{babel}
```

und später

```
\selectlanguage{french}
```

Alle, die schon immer Dokumente in Chinesisch, Japanisch, Koreanisch, Tibetanisch, Arabisch, Hindi, und Khmer schreiben wollten, und das möglichst in einem Text, sollten sich Omega und Lambda anschauen,[13] die Unicode-Varianten von TeX und LaTeX. Sowohl teTeX als auch MikTeX beinhalten diese Pakete.

8.2 Schriften, Formeln und Symbole

Die Default-Schriftart unter LaTeX ist die Bitmap-Schrift cm (Computer Modern). Erstellt man zunächst eine Postscriptdatei und dann aus dieser eine pdf-Datei (etwa mit dem Acrobat-Distiller), sieht das Endergebnis im Acrobat Reader entsetzlich aus. Um das zu vermeiden, verwendet man besser gleich pdfLaTeX, das automatisch die Postscriptschrift cm einsetzt. Eine Alternative ist, gleich auf Postscript-Schriften zu setzen. Wie im gegenwärtigen Text:

```
\usepackage{mathpazo}
\usepackage[scaled]{helvet}
\usepackage{courier}
```

selektiert Palatino für den Haupttext, sowie Helvetica und Courier für die davon abgehobenen Textteile. Andere Möglichkeiten sind mathptmx (entspricht Times) oder bookman und viele andere mehr.

Die Schriftgröße selbst sollte man normalerweise nur in Ausnahmefällen ändern, und sich lieber auf die eingebaute Buchsatzintelligenz von LaTeX verlassen. Falls es doch nötig ist: die Größen rangieren von tiny bis Huge, und man findet die entsprechenden Kommandos in den Menüs von Kile und TeXnicCenter.

Auch die Schriftattribute findet man dort (also fett, italics, small caps, etc.), die man aber im Normalfall sparsam anwenden sollte. Ausnahmen bestätigen die Regel: ich mache in diesem Text exzessiv Gebrauch von `\textsf{}`, das Kommando, um auf eine serifenlose Schrift (hier Helvetica) umzuschalten.

Noch ein Hinweis: bei erstmaliger Verwendung einer Schrift muß diese erst erzeugt werden, was entweder beim Kompilieren des Dokuments oder bei der Anzeige im dvi-Viewer einige Sekunden dauert. Nach einiger Zeit der Verwendung hat man aber alle üblicherweise verwendeten Schriften generiert, und alles geht flott.

Formeln sind die Paradedisziplin von LaTeX. Ein kleines Beispiel:

$$\mathbf{E}_y \sum_0^{t_\varepsilon} L_{x,y^x(s)} \varphi(x)\, ds \in \Gamma_C \tag{1}$$

Nein, ich sage jetzt nicht, wie das geht...wer sich dafür interessiert, kann im Quelltext nachschlagen. Wichtig ist: in den mathematischen Modus, der benötigt wird, griechische Buchstaben und Formeln aller Art darzustellen, wechselt man entweder mit $...$ (beenden nicht vergessen!), oder benutzt für größere Gleichungen das Gleichungsenvironment: `\begin{equation}...\end{equation}`.

Wichtig für Otto Normalanwender sind Brüche: $\frac{3}{4}$, hochstellen: $e^{i\pi}$, tiefstellen: $f_{x,y}$, Grad Celsius: 30°C. Das schreibt sich dann so: `$\frac{3}{4}$, $e^{i\pi}$, $f_{x,y}$,` und `$30^{\circ}$C`. Kompliziertere Konstruktionen, die man ständig benötigt, kann man übrigens abkürzen: einfach vor `\begin{document}` z. B. ein `\newcommand{\grad}{$^{\circ}$C}` definieren, und überall im Text dann `30\grad` schreiben.

Symbole sind eine weitere Paradedisziplin von LaTeX. Die wichtigsten werden in den Menüs von Kile und TeXnicCenter angeboten. Die komplette Liste[14] umfaßt über 2000 Symbole, und führt auch die Packages auf, die man zusätzlich laden muß.

8.3 Abbildungen

Das wird oft gefragt. Es ist ganz einfach (zum Beweis siehe Abb. 1):

```
\begin{figure}[htb]
\includegraphics[clip,width=7.0cm]{bild}
\caption{Die Bildunterschrift.}
\label{bild1}
\end{figure}
```

`bild` ohne Endung, das ist so gewollt (siehe unten)!

Hier schaltet `\begin{figure}[htb]` in das Figure-Environment, das man mit `\end{figure}` wieder verläßt. Man kann das auch weglassen, beraubt sich aber dadurch der automatischen Positionierung der Abbildung (TeX nennt solche Objekte Floats). Im vorliegenden Beispiel wäre die Positionierung ganz LaTeX überlassen, da sämtliche Möglichkeiten als Option übergeben wurden: [htb] steht für here, top, bottom. Normalerweise sieht es

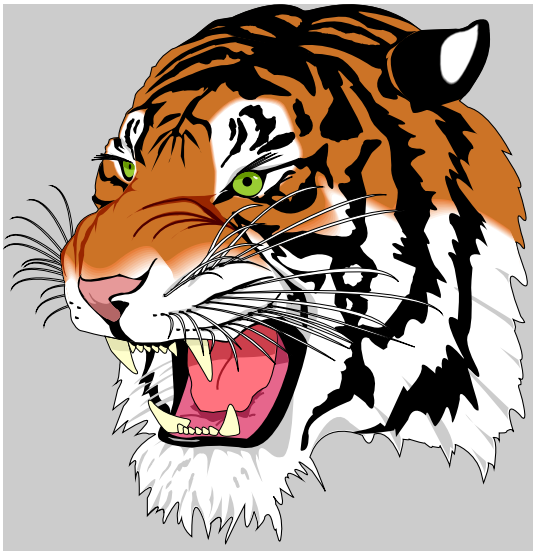

Abbildung 1: Ein Tiger. Man beachte die Schnurrhaare und die gelben Zähne.

besser aus, wenn Abbildungen entweder oben oder unten auf der Seite angeordnet sind, was man mit einem [t!] oder [b!] erzwingen kann. Dies sollte man allerdings erst vornehmen, nachdem der Text abgeschlossen ist.

Mit

```
\includegraphics[clip,width=7.0cm]{bild}
```

wird die Abbildung eingebunden (falls sich die Datei in einem anderen Verzeichnis befindet, gibt man den Pfad an). Die Option `clip` sorgt dafür, daß die eventuell vorhandenen Ränder der Abbildung keinen Text verdecken (man kann sie normalerweise weglassen). Die Option width sollte man so wählen, daß die Abbildung deutlich zu sehen ist, ohne über die Spalte oder gar Seite zu ragen. Hierbei können auch relative Angaben verwendet werden, wie z. B. `0.8\columnwidth`. Statt width kann man auch height verwenden, oder beide zusammen, wodurch man die Abbildung lustig verzerren kann. angle ist eine der vielen anderen Optionen.

Die Abbildung bild sollte eine Encapsulated Postscript-Datei (*.eps) sein, die man leicht unter jedem Betriebssystem durch das Drucken in eine Datei über einen postscriptfähigen Druckertreiber erstellen kann. Bei Vektorgrafiken versteht sich das von selbst, bei Bitmaps wundert sich vielleicht der eine oder andere. Nun, pdfLaTeX kann auch mit png-Dateien umgehen, LaTeX selbst dagegen nicht. Es gibt Möglichkeiten, auch andere Bitmaps einzubinden, aber glaubt mir, das wollt ihr gar nicht wissen. eps-Dateien sind die beste Wahl, und bereiten nie Probleme.

Mit einer Ausnahme: leider unterstützen die wenigsten Programme (CorelDraw ist eines) einen echten eps-Export. Die oben erwähnte Funktion „Drucken in eine Datei“ druckt leider die ganze Seite, das heißt, die Abbildung hat womöglich riesige Ränder. Der width-Befehl oben bezieht sich aber auf die Seite mit Rändern, wodurch die eigentliche Abbildung viel zu klein dargestellt wird.

Man kann das mit allerhand Getrickse hinbiegen, das sauberste und einfachste ist jedoch, diese Ränder zu beschneiden. Dazu öffnet man die betreffende eps-Datei mit einem Editor (z. B. Kile oder TeXnicCenter). Ziemlich weit oben steht `%%BoundingBox: 22 171 567 738`, wobei das Zahlenquadrupel die x-y-Koordinaten der eigentlichen Abbildung darstellt (und zwar in der Reihenfolge: $x_{\min}$, $y_{\min}$, $x_{\max}$, $y_{\max}$). Nun öffnet man die Datei mit Ghostview (nicht mit KGhostview!), das als Standardmauszeiger ein Fadenkreuz darstellt. Fährt man an die jeweiligen Ränder der Grafik, werden die Koordinaten angezeigt. Diese notiert man oder merkt man sich, ändert sie im Editor, speichert, und hat eine schöne Abbildung ohne Ränder. User, die Perl installiert haben, können dies übrigens mittels eines kleinen Scripts automatisieren, und so sehr bequem Abbildungen ohne jegliche Ränder erhalten.[15]

Daß der Dateityp nicht explizit übergeben wird, hat einen wichtigen Grund. pdfLaTeX kann mit eps-Dateien nicht umgehen, und träfe es auf eine, würde es umgehend den Dienst einstellen. Hier kommt das Hilfsprogramm epstopdf zum Zuge, daß in der Präambel geladen wird. Damit es gestartet werden kann, muß ein -shell-escape an pdfLaTeX übergeben werden. . .darum also!

Wird dies übergeben, startet epstopdf im Hintergrund und übersetzt blitzschnell alle eps-Dateien in pdf-Dateien. Eine feine Sache. . .allerdings mit dem Haken der oben erwähnten Sicherheitslücke. Wem es nicht gefällt, LaTeX die Ausführung von Programmen zu erlauben, muß epstopdf manuell starten.

Was verwunderlicherweise auch langjährige LaTeX-User oft nicht wissen: natürlich kann man auch Abbildungen nebeneinander oder untereinander darstellen. Ersteres geht mit

```
\includegraphics[width=3cm]{bild} \hfill
\includegraphics[width=3cm]{bild},
```

letzteres mit

```
\includegraphics[width=3cm]{bild}\\
\includegraphics[width=3cm]{bild}.
```

Eine leicht modifizierte Variante der ersten Möglichkeit zeigt Abb. 2.


Abbildung 2: Links ein Tiger. Rechts ein breiter Tiger.☺

Schließlich kann man auch Text um eine Abbildung herumfließen lassen. Das macht in einem zweispaltigem Text wie diesem natürlich überhaupt keinen Sinn, trotzdem zeigt Abb. 3 ein Beispiel. Um das zu tun, muß zunächst die floatflt-Package geladen werden, also

```
\usepackage{floatflt}.
```

Die Grafik selbst erstellt man mit

```
\begin{floatingfigure}{3cm}
\centerline{
\includegraphics[width=3cm]{golfer}
}
\caption{Ein nettes Golfmädel.}
\label{weib}
\end{floatingfigure}
```

Eigentlich klar, oder? Auf fortgeschrittenere Konzepte wie die der minipages gehe ich hier jetzt nicht ein.

8.4 Tabellen

Auch Tabellen sind einfach (siehe Tab. 1 und Tab. 2):

```
\begin{table}[t!]
\begin{center}
\caption{Eine sehr einfache Tabelle.}
\begin{tabular}{lcc}\\
\hline
Bit & A & B\\
\hline
on  & 1 & 0\\
off & 0 & 1\\
\hline
\end{tabular}
\label{Tab1}
\end{center}
\end{table}
```

Nehmen wir das doch mal im Detail auseinander. `\begin{table}[t!]` startet das Table-Environment, das vollkommen analog zum Figure-Environment von oben ist und ebenso wie dieses ein Float ist. Mit `\end{table}` verläßt man es wieder. Die Befehle `\begin{center}...\end{center}` zentrieren das alles, was hier besser aussieht, da unsere Tabelle so klein ist. Die `\caption` erklärt sich von selbst. Die eigentliche Tabelle steht zwischen `\begin{tabular}{lcc}` und `\end{tabular}`. Das Argument `{lcc}` definiert drei Spalten, die erste linksbündig (left), die zwei letzten zentriert (centered). `\\` bewirkt einen Zeilenumbruch, `\hline` eine horizontale Linie. Nun kommen die Einträge der Tabelle, wobei der Ampersand & als Tabulator oder Spaltenoperator dient.

Tabelle 1: Eine sehr einfache Tabelle.

Bit	A	B
on	1	0
off	0	1

Mit diesem simplen Rezept kann man auch etwas realistischere Tabellen erstellen, wie in Tab. 2 gezeigt (Interessierte schauen bitte im Quelltext nach). Mit den Packages supertab und longtab lassen sich darüberhinaus absolut monströse, mehrseitige Tabellen und ineinander verschachtelte Tabellen erstellen. Absolut empfehlenswert für alle, die Schachrätsel und Intelligenztests mögen...

Abbildung 3: Ein nettes Golfmädel.

8.5 Zitate, Crosslinks und Hyperrefs

Zitate sind unerläßlich, um Quellen zu belegen. Die Nummerierung sollte aber nicht manuell erfolgen, sondern von LaTeX durchgeführt werden. Das ist leicht. Man erstelle ganz am Ende des Texts folgende Zeilen:

Tabelle 2: Eine einfache Tabelle. Man beachte die vielen hübschen Zahlen.

	$(11\bar{2}4)$ FWHM (deg)		α (deg)		
	$o \cdot \kappa = 0$	$o \cdot \kappa \neq 0$	XRD	TEM	GIXD
#A	0.072	0.24	0.36	—	—
#B	0.047	0.367	0.6	—	—
#C	—	0.344	0.56	0.63	—
#D	—	0.109	0.16	0.15	—
#E	—	0.249	0.38	—	0.31
#F	—	0.37	0.57	—	0.55

```
\begin{thebibliography}{99}
\bibitem{dottie} www.n_dot_force.com
\bibitem{piet} Ha ick vom piet jehört...
\bibitem{doc} Ich will kochen!
\end{thebibliography}
```

und verweise darauf im Text mittels `\cite{dottie}` etc. Anschließend lasse man LaTeX dreimal durchlaufen (damit diese Referenzen aufgelöst werden können), et voilà!

Eine professionellere Variante gelingt mit dem Zusatzprogramm BibTeX, das auf eine (vom Benutzer erstellte) Literaturdatenbank zugreift. Für jeden, der häufig die gleichen Arbeiten zitiert, ist dies absolut empfehlenswert, da man einfach auf die Datenbank zurückgreifen kann, anstatt die gleiche Referenz wieder einmal eintippen zu müssen.

Crosslinks wurden weiter oben schon erwähnt: man kann Abbildungen, Tabellen, Gleichungen, und auch Zitate mit einem Label versehen (`\label{name}`), und dann dieses im Text mittels `\ref{namen}` ansprechen. So bleiben die Bezüge in einem Dokument immer korrekt, auch wenn man neue Objekte einfügt.

Hyperrefs werden von pdfLaTeX bereitgestellt. Mit

```
\usepackage[pdftex,bookmarks]{hyperref}
```

aktiviert man diese. `colormarks` ist eine zusätzliche Option, die die Hyperrefs einfärbt (es gibt noch ca. 70 andere). Hyperref-fähige Viewer wie der Acrobat Reader und Xpdf (nicht KGhostview!) gehen dann beim Mausklick auf den Hyperref im Inhaltsverzeichnis oder im Text (Zitate) an die betreffende Stelle im Text. Bei vorliegendem Text sind Inhaltsverzeichnis, Zitate und URLs als Hyperrefs implementiert.

8.6 Inhaltsverzeichnis und Index

Das Inhaltsverzeichnis ist kinderleicht: Ein

```
\tableofcontents
```

nach `\begin{document}` genügt vollkommen. Nichtnummerierte Teile des Textes wie die Danksagung (`\section*{Danksagung}`) werden mit

```
\addcontentsline{toc}{section}{Danksagung}
```

hinzugefügt. Das war's.☺

Die Erstellung des Indexes dagegen ist schon ein recht fortgeschrittenes Thema, etwa vergleichbar mit der Zitierung mittels BibTeX. Zunächst einmal muß man den Index manuell erzeugen, indem man im Text indexwürdige Begriffe mit `\index{}` umrahmt. Ein `\makeindex` im Vorspann des Dokuments erstellt dann eine Datei, die für die Indexerzeugung benutzt wird, und zwar von dem Hilfsprogramm makeindex. Die Default-Profiles von TeXnicCenter enthalten sowohl BibTeX als auch makeindex, bei Kile muß man beide manuell starten.

Danach hat man einen Index.

8.7 Farbe

Um farbig schreiben zu können, muß man...na was wohl? Richtig! Die Package laden, und zwar mit

```
\usepackage{xcolor}.
```

Wer hätte das gedacht?

Jetzt können wir in Farbe tippen...und wenn schon, denn schon, also z. B.:

MSOffice MSOffice

Dabei stehen die Grundfarben schon zur Verfügung, und ein `\textcolor{blue}{C}` erzeugt ein blaues C, ebenso wie ein `\colorbox{red}{text}` ein rotes Feld erzeugt. Eigene Farben kann man sich zusammenmixen:

```
\definecolor{mygray}{gray}{0.7}
\definecolor{myyellow}{rgb}{1,0.7,0}
\definecolor{orange}{rgb}{1,0.5,0}
```

vor `\begin{document}` einfügen, und schon kann man seine eigenen Farben benutzen (ich habe keine Ahnung, ob mein Orange wirklich orange ist).

9 Typografische Feinheiten

Schriften hin, Farbe her: elementare typografische Sünden sieht man fast in jedem LaTeX-Text. Die zwei häufigsten sind die der Abkürzungen und der Bindestriche.

- Eine häufige Abkürzung ist z. B. „z. B.“. Die Punkte werden von LaTeX als Satzende interpretiert, und es fügt daher dort einen größeren Abstand ein als zwischen Worten (was bei tatsächlichen Satzenden auch sinnvoll ist und das Lesen ungemein erleichtert). Um LaTeX mitzuteilen, daß ein Punkt nicht das Ende der Welt ... äh, des Satzes bedeutet, schreibt man `.\`. Auch sollte man „z. B.“und ähnliche Dinge nicht trennen, das sieht greußlich aus. Um Trennung zu vermeiden, verbindet man zwei Buchstaben normalerweise mit einer Tilde, also ~. Bei Abkürzungen wie dieser möchte man aber zusätzlich noch einen geringeren Abstand, denn man mittels `z.\,B.\` erhält.
- Mit Bindestrichen wird auch viel Schindluder getrieben. Eineinfacher Bindestrich (-) wird zum Verbinden von Worten benutzt, wie in diesem Beispiel: Word-Fetischist. Ein doppelter Bindstrich (--),

auch Halbgeviertstrich (englisch: en-dash) dagegen wird zum Verbinden von Sätzen verwendet: „Der Word-Fetischist zweifelte an der Welt – die ohnehin schon schlecht genug war – nicht wegen Word – weit gefehlt!“ Der dreifache Bindstrich (`---`), auch Geviertstrich (englisch: em-dash) findet im Deutschen keine Anwendung, wohl aber im Englischen:„the Word fetishist doubted the world—which was bad enough anyway—not because of Word—far from it!“ Ein Minus-Zeichen ist wieder etwas ganz anderes, und läßt sich unter LaTeX so setzen: `$-20^{\circ}$C`, also -20°C.

Noch einmal nebeneinander: -, –, —, −.

10 Fazit

8 Seiten . . .mehr, als ich erhoffte, und weniger, als ich befürchtete. Deutlich weniger zumindest als die 1000 Seiten, die sonst üblich sind. Trotzdem steckt in diesem kurzen Text alles, was zum Schreiben z. B. einer Veröffentlichung oder einer Diplomarbeit notwendig ist. Natürlich verlangt die verkürzte Darstellung dem Leser viel ab. Ich hoffe trotzdem, daß diese Schnellanleitung verständlich ist, und dem einen oder anderen LaTeX näherbringt. Wäre dies so, würde ich mich freuen.

Anhang: Schritt-für-Schritt

Im folgenden möchte ich für beide der hier verwendeten Editoren erklären, wie man Schritt für Schritt einen einfachen Text schreibt, diesen kompiliert und auf dem Bildschirm bewundert. Der erste Abschnitt sollte auch von Windows-Usern gelesen werden, da ich die Erklärung einiger Befehle im nächsten Abschnitt nicht wiederhole.

Kile

Zunächst einmal starten wir Kile (was. . .das wußtet ihr schon?). Die Menübefehle schreibe ich fett (beim Auswählen erscheint eine kurze Erklärung des Befehls im Messages/Log File-Fenster). Also:

1. **File/New**: ein neues Dokument wird geöffnet.
2. **File/Save As**: speichert den Text unter jedem gewünschten Namen ab. Die Extension allerdings muß *.tex sein, *nicht* *.TeX oder irgendetwas anderes!
3. **LaTeX/documentclass**: damit erhält man `\documentclass[10pt]{}`. Damit dieser Befehl vollständig ist und unseren Bedürfnissen entspricht, ändern und ergänzen wir ihn zu `\documentclass[12pt,a4paper]{scrartcl}`
4. **LaTeX/usepackage**: man erhält `\usepackage{}`. Auch diesen Befehl müssen wir natürlich ergänzen, denn woher soll Kile wissen, welche Packages wir laden wollen? Wir laden zunächst einmal die Unterstützung der deutschen Sprache mit `\usepackage[german]{babel}`. Wir wiederholen diesen Schritt noch zweimal und laden mit `\usepackage[latin9]{inputenc}` die direkte Eingabe der Umlaute und des ß sowie mit `\usepackage{mathpazo}` die Postscriptschrift Palatino, die auch im pdf-Viewer gut ausschaut.
5. **LaTeX/begin{document}**: man erhält ein `\begin{document} ... \end{document}`. Der Cursor steht zwischen diesen beiden Befehlen und lädt zum Tippen ein. Ja, tatsächlich: wir könnten jetzt loslegen und hier jeden x-beliebigen Unsinn schreiben, wie z. B. Hallo, LaTeX! Aber ein paar Sachen wollen wir noch ausprobieren.
6. **LaTeX/author**: man erhält `\author{}`. In die geschweiften Klammern schreibt ihr. . .na was wohl, euren Namen rein!
7. **LaTeX/title**: man erhält `\title{}`. In die geschweiften Klammern schreiben wir einen sinnigen Titel, wie etwa „Konformität als Maß gesellschaftlicher Akzeptanz“.
8. **LaTeX/maketitle**: man erhält `\maketitle`. Es gibt keine geschweiften Klammern, also gibt es auch nichts einzutragen. Dieser Befehl generiert den Titel mit Datum. Wem das nicht gefällt, der schreibe gleich vor `\maketitle` ein `\date{}` (dann entfällt das Datum), oder auch ein `\date{1061 BC}`, um anzudeuten, wie alt der Text ist.
9. **LaTeX/Sectioning/section**: hui, da geht eine Dialogbox auf! Da schreiben wir jetzt was rein, meinethalben „Gesellschaftliche Akzeptanz“. Die Nummerierung lassen wir so, sonst wird diese Sektion nicht nummeriert. Das ganze machen wir gleich nochmal, und betiteln die zweite Sektion z. B. mit „Konformität“. Das gleiche Spiel könnt ihr, wenn ihr wollt, mit den übrigen Section-Befehlen wiederholen. Diese sind im Menü hierarchisch angeordnet, also versucht bitte nicht, zunächst einen Paragraphen zu definieren, und in diesem dann eine Sektion (Kapitel gibt es übrigens nur in der Klasse book, nicht in der von uns gewählten Klasse article).

10. Jetzt tippen wir mal unter dem ersten der \section-Befehle einen Text ein. Ruhig mehrere Zeilen! Kile bricht die Zeilen automatisch um, also untersteht euch, auf die Return-Taste zu hauen! Wenn ihr das Gefühl habt, es sei genug, macht zwei mal Return, so daß eine Leerzeile entsteht. Das erzeugt euren ersten Absatz. Tippt nochmal was ein, damit auch ein zweiter entsteht.

11. Jetzt versuchen wir mal was ganz anderes. Positioniert den Cursor unter dem zweiten der \section-Befehle, und wählt **LaTeX/List Environment/begin{itemize}**. Das erzeugt eine von allen Managern geschätzte Bullet-Liste:

    ```
    \begin{itemize}
    \item
    \end{itemize}
    ```

    Hinter dem Kommando \item schreibt man etwas möglichst sinniges. Man kann dann so viele\item-Befehle hinzufügen, wie man mag. Genauso geht das auch mit den übrigen Befehlen in diesem Menü.

12. So, das sollte uns für den Anfang erst einmal genügen. Kompilieren wir das ganze mal! Dazu klickt ihr entweder auf das linke blaue Zahnrad im Iconbar, oder verwendet **Tools/LaTeX**, oder drückt F2. LaTeX rast jetzt durch den Text und zeigt im Messages-Fenster unzählige Informationen an. Bei einer Fehlermeldung: man gehe an die betreffende Zeile, und schaue, ob man etwa eine geschweifte Klammer vergessen hat. Das macht 70% aller Fehler aus (die restlichen 30% sind ein vergessenes $ im mathematischen Modus, was aber erst im Haupttext besprochen wird).

    Zum Anschauen nehmen wir entweder den bebrillten Löwen rechts vom blauen Zahnrad, oder **Tools/View DVI**, oder F3. Analog geht das auch mit pdfLaTeX: das rechte blaue Zahnrad, uswusf. Wenn ihr Euch für ein Lieblingsformat entschieden habt, könnt ihr unter **Options/Configure Kile** auch ein Quickbuild definieren, das man fortan über den Blitz im Iconbar erreicht.

13. Jetzt kommen die Extrawürste. Sind euch die Ränder zu groß? Dann definiert eigene:

    ```
    \usepackage[
    ,textwidth=18cm
    ,textheight=22cm
    ,top=38.1mm
    ,left=11mm
    ,right=11mm
    ,verbose
    ,dvips
    ]{geometry}
    ```

    vor \begin{document}. Die Zahlen sind natürlich nur Beispiele.

14. Euch gefällt der Titel nicht? Dann löscht die Zeilen

    ```
    \author{}
    \title{}
    \maketitle
    ```

    und fügt statt dessen z. B. ein:

    ```
    {\LARGE\textbf{Konformität ...}}\\

    \vspace*{0.3cm}
    \noindent
    {\large erstellt am \today\ von}\\

    \vspace*{0.3cm}
    \noindent
    {\large\textsc{Otto Normalanwender}}\\
    ```

    Der Befehl \\ erzwingt einen Zeilenumbruch. Im Fließtext sollte man ihn nie oder zumindest sehr selten verwenden.

15. Ihr wollt Worte groß schreiben oder fett, oder in einer anderen Weise hervorheben? Nun, oben stehen einige Beispiele. All dies läßt sich im Iconbar und unter **LaTeX/Font Styles** ganz zwanglos auswählen.

So, das war's. Den ersten Schritt haben wir hiermit getan, und wer mehr wissen will, steige wieder in den Haupttext ein.

TeXnicCenter

Zunächst einmal starten wir TeXnicCenter (sagt jetzt nicht, das ahntet ihr schon...). Die Menübefehle im folgenden schreibe ich fett. Also:

1. **Datei/Neu**: ein neues Dokument wird geöffnet.
2. **Datei/Speichern unter**: speichert den Text unter jedem gewünschten Namen ab. Als Extension empfiehlt sich *.tex, aber auch anderes ist möglich.
3. **Datei/Neu von Vorlage**: dies ist eine Spezialität von TeXnicCenter, die ermöglicht, Vorlagen (Templates) für Dokumente zu verwenden und

zu erstellen. Der Befehl erzeugt ein neues Dokument gemäß einer zu spezifizierenden Vorlage. Unter **Extras/Optionen/Verzeichnisse/Dokumentenvorlageverzeichnisse** gibt man ein Verzeichnis an, oder benutzt den vorgegebenen Pfad, um seine Vorlagen an dieser Stelle zu verwalten. In diesem Verzeichnis legt man sich für seine Vorlagen entsprechende Unterverzeichnisse an (das ist zwingend!). Für den Schriftverkehr also z. B. einen Ordner namens *Artikel*.

4. **Vorlagen erzeugen**: Über Schritt 1. und 2. legt man ein neues Dokument an, das man im gewählten Template-(Unter)Verzeichnis abspeichert. Für die jeweilige Aufgabe stellt man nun die entsprechende Dokumentklasse und die benötigten Packages zusammen. Im Gegensatz zu Kile muß man hierzu aber selbst zum Keyboard greifen. Tippt also:

```
\documentclass[12pt,a4paper]{scrartcl}
\usepackage[latin9]{inputenc}
\usepackage[german]{babel}
\usepackage{mathpazo}
\begin{document}

\end{document}
```

Die Erklärungen zu diesen Befehlen finden sich in der obigen Beschreibung von Kile.

Dieses Template speichert man im Unterverzeichnis *Artikel* ab. Will man nun später ein neues Dokument erstellen, kann man die Vorlage über **Datei/Neu von Vorlage** unter dem Karteireiter **Artikel** wieder benutzen. So lassen sich bequem für jede Aufgabe Vorlagen erstellen, die man den jeweiligen Anforderungen an das zu erstellende Dokument anpassen kann. Um eine gewisse Ordnung zu erhalten, sollte man unter **Extras/Optionen/Verzeichnisse/Arbeitsverzeichnis** einen Pfad angeben, in dem die Dokumente abgespeichert werden. Legt man an dieser Stelle keinen Pfad an, werden die Dokumente per Default in `C:\Eigene Dateien` abgespeichert.

Will man sich die Mühe ersparen, eigene Vorlagen zu erstellen, so gibt es auch die Möglichkeit, vorgefertigte Templates zu benutzen. Dazu schaue man z. B. auf der TeXnicCenter-Seite nach[9] oder bemühe Google.

Nun geht es weiter, analog zum Abschnitt **Kile** ab Punkt 6.

Einfügen/Dokumenttitel/Titeleigenschaften : man erhält `\title{} \author{} \date{}`. Jetzt könnt ihr einen Titel wie etwa „Die Inversion als Stilmittel der Rhetorik“ angeben, euren Namen eintragen und das jeweils aktuelle Datum generieren lassen (mit `\date{\today}`, s. o.!).

Einfügen/Dokumenttitel/Titel : generiert den Titel durch das Kommando `\maketitle`.

Einfügen/Überschrift : Hier können Abschnitte, Unterabschnitte etc. zur Gliederung des Dokumentes erzeugt werden, ganz analog zu Punkt 9. im Abschnitt **Kile**.

Einfügen/Aufzählungen : erzeugt Listen. Mit **Nummerierte Aufzählung** z. B. eine nummerierte Liste:

```
\begin{enumerate}
\item
\end{enumerate}
```

Diese Liste z. B. ist so entstanden. . .also tippt ruhig was ein, und erstellt noch ein paar `\item`s mehr!

5. Wollen wir das Dokument jetzt kompilieren, benutzen wir die Tastenkombination **[Strg]+[F7]** oder **Ausgabe/Aktives Dokument/Compilieren**. Ein Button für diese Funktion ist selbstverständlich auch vorhanden. . . Über das aus den meisten Windows-Anwendungen bekannte Symbol „Ansicht“ (Blatt mit Lupe) oder der Taste **[F5]** können wir unser Dokument im jeweils benutzten Viewer (z. B. Ghostview oder Acrobat) betrachten.

6. **Format**: hier könnt Ihr bequem die Schrifteigenschaften, Textausrichtung, und Absätze euren Wünschen anpassen.

7. **Mathe**: alles rund um das Erstellen von Formeln, Sonderzeichen, Symbolen etc. findet ihr hier.

8. **Extrawürste**: Wie oben bei Kile ab Punkt 13!

9. **Projekte**: TeXnicCenter erlaubt auch das Erstellen von Projekten, also Dokumenten, die aus mehreren Dateien bestehen. Dies ist bei sehr großen Dokumenten (Diplom- oder Doktorarbeiten) sehr nützlich. In der Hilfe findet ihr eine genaue Beschreibung dieser Funktion.

So, das war's. Den ersten Schritt haben wir hiermit getan, und wer mehr wissen will, steige wieder in den Haupttext ein.

Danksagung

Ich danke n.force, dem trotz (oder wegen?) völliger Unkenntnis von LaTeX viel Nützliches zu diesem Text einfiel. Dank auch an Lucky, der (als totaler Anfänger) nicht nur innerhalb kurzer Zeit MiKTeX installiert hatte, sondern mir auch anschließend Anregungen zukommen ließ. OllerKater schickte mir sogar sein erstes LaTeX-Dokument, was ich ganz besonders nett fand.

Mein besonderer Dank aber an piet. Viele Ideen zu diesem Text stammen von ihm. Er hat mir auch die Arbeit abgenommen, die Schritt-für-Schritt-Anleitung für TeXnicCenter zu erstellen, und auch die Mühe, diesen Text zu lesen und zu überprüfen, hat piet übernommen.

Schließlich gilt mein Dank Rolf Niepraschk, der diesen Text nicht nur inhaltlich, sondern auch nach formalen Kriterien durchkämmte, und etliche obsolete Konstrukte entdeckte. Ferner erinnerte er mich an das im deutschen Sprachraum unverzichtbare KOMA Script. □

Literatur

[1] Helmut Kopka, *LaTeX: eine Einführung*, 1. Auflage, Addison-Wesley (Bonn, Paris) 1994 (Vorzuziehen ist die dritte, inzwischen dreibändige Auflage.)

[2] M. Goosens, F. Mittelbach, und A. Samarin, *The LaTeX Companion*, 1st edition, Addison-Wesley (Berkeley) 1994 (Ist auch auf Deutsch erhältlich. Vorzuziehen ist aber die zweite Auflage, die momentan allerdings nur auf Englisch erhältlich ist.)

[3] www.piware.de/docs/latex-1ststeps.pdf, www.ctan.org/tex-archive/info/lshort/german/l2kurz2.pdf

[4] soliton.science.uva.nl/ wijnhout/Kite oder auch rpmfind.net

[5] www.dante.de Man beachte CTAN-search: www.dante.de/cgi-bin/ctan-index

[6] www.cs.wisc.edu/ ghost/doc/AFPL/get814.htm

[7] www.cs.wisc.edu/ ghost/gsview/get46.htm

[8] www.miktex.org

[9] www.texniccenter.org. Man beachte auch die User-Group von TeXnicCenter, die Templates zum Download anbietet (nur für registrierte Leser): groups.yahoo.com/group/TeXnicCenter-Users/files/Templates-TeX

[10] www.xm1math.net/texmaker/

[11] www.tug.org/tds

[12] unimac.switch.ch/students/latex.de.html

[13] omega.cse.unsw.edu.au:8080/index.html

[14] ftp.dante.de/tex-archive/info/symbols/comprehensive/symbols-a4.pdf

[15] www.cs.washington.edu/homes/zasha/latexexample/TightBoundingBox.pl