

LaTeX leicht gemacht, professioneller Textsatz für Sado-Masochisten?

[Frank Rennemann](#)

Manch einer, der es nicht kennt, hält LaTeX (sprich Latech) vielleicht für eine sehr exotische Klamottenart für noch exotischere Gelegenheiten. Weit gefehlt, hier geht es nicht um schwarzes Lackleder, sondern um eine seit langem ausgereifte Art, komplexe Texte zeit-, ressourcen- und nervensparend professionell zu Papier zu bringen.

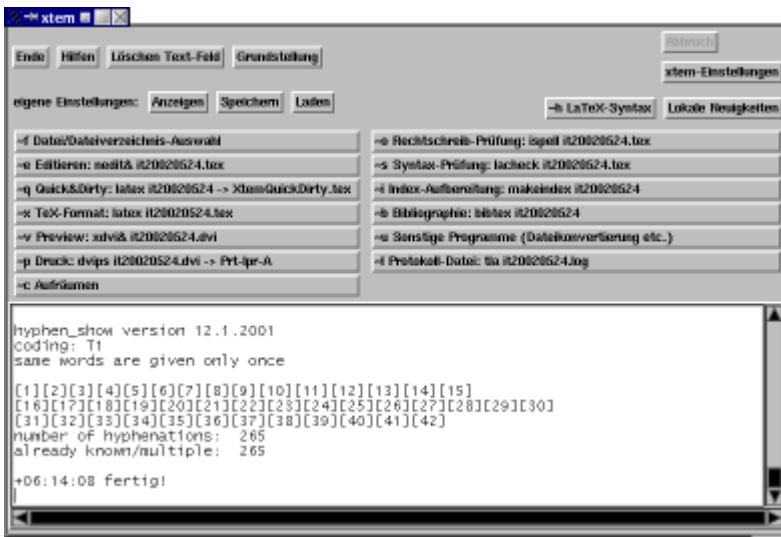
Das Geheimnis dieses Erfolges heißt LaTeX und basiert auf TeX. Was ist TeX? TeX ist ein Textsatzsystem, entwickelt Mitte der 70er Jahre von Donald Knuth, der ein leistungsfähiges System brauchte, mit dem er Artikel, Aufsätze und Bücher setzen konnte. Und was ist nun LaTeX? TeX selbst ist etwas schwierig zu bedienen, was den täglichen Umgang damit nicht eben einfacher machte. Leslie Lamport entwickelte daher ein umfangreiches Makropaket für TeX, das es ermöglicht, die Standard-Dokumente `article`, `report` und `book` sowie `letter` mit einfachen Makros und Vorlagen schnell und effizient zu erstellen. Die aktuelle Version dieses Makropakets ist *latex2e*.

Und warum sollte ich das benutzen?

Schließlich gibt es doch genug Office-Pakete mit Textverarbeitungen, die leicht zu bedienen sind. Nun, natürlich wird niemand gezwungen, LaTeX zu benutzen, aber wer sich beim Verfassen umfangreicher Texte von fünfzig oder mehr Seiten schon einmal mit WinWord und allen ähnlichen Spielarten herumgeschlagen hat, ist vielleicht immer noch auf der Suche nach einem Programm, mit dem er seine Examens-, Diplom- oder Doktorarbeit oder evtl. sogar gleich ein ganzes Buch nervenschonend zu Papier bringen kann. Während normalen Textverarbeitungsprogrammen bei 100 und mehr Seiten langsam die Puste ausgeht, wird LaTeX da erst so richtig munter. Und die verfassten Texte sind zukunftssicher. Meine Diplomarbeit, 1992 auf einem Atari in LaTeX 2.09 geschrieben, ist auch heute noch auf jedem Rechner, auf dem ein TeX/LaTeX-Gespann läuft, mit der kompletten Formatierung lesbar. Und ganz nebenbei ist man gegen Makroviren a la WinWord gefeit.

Wie komme ich da ran?

Hier gibt es nur gute Nachrichten, TeX/LaTeX ist eigentlich bei jeder Linux-Distribution dabei. Und es gibt sie nicht nur für x86-Linux, sondern auch für alle anderen Betriebssysteme und Rechnerarchitekturen, angefangen bei ATARI ST bis zur IBM zSeries. Dazu gibt es je nach Betriebssystem unterschiedliche Shells, sprich Bedienprogramme. Für Linux etwa `texshell`, [kile](#) oder `xtem`, welches seit längerem mein Favorit ist. Und natürlich sind sowohl TeX als auch LaTeX frei verfügbar, die mehreren Fantastillionen Euro, die man sonst in eine Textverarbeitung steckt, kann man bei TeX getrost in eine neue Kaffeemaschine investieren.



Das Hauptfenster von xtem

Und wie funktioniert's?

Zuerst die schlechte Nachricht (wenn's denn eine ist), anders als bei Office-Paketen wie OpenOffice schreibt man den Text nicht in einem WYSIWYG-Fenster (What You See Is What You Get). Man sieht also nicht schon während des Schreibens, wie der Text später ausgedruckt aussehen wird. Jetzt die gute Nachricht, LaTeX-Dokumente sind reiner ASCII-Text, den man mit einem beliebigen Editor erstellen kann, von vi bis emacs. Ich persönlich nehme nedit, das hat zwar keine LaTeX-relevanten Zusatzfunktionen, beherrscht aber LaTeX Syntax-Highlighting. Neuerdings bietet kile eine komfortable Möglichkeit, TeX-Editor und Shell in einem zu nutzen.

Wer sich schon einmal mit HTML befasst hat, dem wird ein LaTeX-Dokument bekannt vorkommen. Alle Formatierungsbefehle sind direkt im Text enthalten. Im Kopf der Datei stehen die Formatierungsbefehle, die für das gesamte Dokument anzuwenden sind, etwa, welches Makropaket verwendet werden soll, um welche Art Dokument es sich handelt, wie groß die Standard-Schriftgröße sein soll etc.

Zusätzlich lassen sich die eingebundenen Makropakete noch modifizieren, etwa, um andere Zeilenabstände zu bekommen oder den bedruckbaren Seitenbereich zu verändern. Grafiken lassen sich entweder direkt in LaTeX erstellen (einfache Schemazeichnungen etwa), oder als EPS dazuladen. Um eine druckbare Datei zu erzeugen, startet man TeX (für den Einsteiger am besten mittels einer der oben genannten Shells), wobei aus der .tex-Datei mitsamt etwaiger weiterer Dateien wie z.B. den Grafiken eine DVI-Datei (DeVice Independent = Geräteunabhängig) erzeugt wird, die mittels TeX-Druckprogramm auf jeden x-beliebigen Drucker ausgegeben werden kann, vom alten Neunadeldrucker bis zur Linotype-Textsatzmaschine. Und anders als bei Officepaketen, wo sich je nach Druckertreiber auch schon mal einzelne Zeilen auf die nächste Druckseite verschieben, kommen DVI-Dateien auf allen Ausgabemedien in der selben Formatierung heraus, nur eben in unterschiedlicher Qualität :-). Passend zu diesem Konzept ist das Programm, mit dem man sich DVI's am Bildschirm ansehen kann, eigentlich auch nichts anderes als ein Druckprogramm, nur das es eben auf den Bildschirm bzw. in ein Fenster druckt und noch ein paar Extrafunktionen wie Zoom oder eine Lupe anbietet.

Das Konzept von Master- und Filial-Dokumenten kam mit LaTeX erstmals auf, d.h. bei größeren Projekten wie Büchern kann man ein Hauptdokument mit den globalen Formatierungen und den Verweisen auf Teildokumente erstellen. Den eigentlichen Inhalt kann man dann z.B. kapitelweise in Unterdokumente auslagern und einzeln bearbeiten:

```
\include{<Unterverzeichnis>/<Dateiname ohne .tex>}
```

direkt an der Stelle im Text, wo später das Unterdokument eingefügt werden soll. Durch

Auskommentieren einzelner Filialdokumente läßt sich so viel Zeit sparen, denn zum Ansehen und Kontrollieren der aktuellen Arbeit muß nicht jedes Mal das komplette Buch oder die Diplomarbeit durch den TeX-Compiler.

Eine Stärke von LaTeX ist der Formelsatz, weshalb es besonders im naturwissenschaftlichen und mathematischen Umfeld sehr geschätzt wird. Wer schon einmal versucht hat, eine komplexe Summe mit Mehrfachbrüchen oder ein umfangreiches Integral mit dem WinWord-Formeleditor zu setzen, wird LaTeX nicht mehr hergeben wollen. Die andere Stärke ist die automatische Erstellung diverser Listen wie Inhaltsverzeichnisse, Abbildungs- und Tabellenverzeichnisse.

Ein LaTeX-Beispiel sagt mehr als 1000 Words

Genug der Theorie, gehen wir mal *in medias res* und erzeugen aus diesem Artikel mittels `xtem` und dem persönlichen Lieblingseditor ein LaTeX-Dokument. Zuerst wird der ASCII-Text mal in `<Dateiname>.tex` umbenannt. Nach dem ersten Start von `xtem` sollte man den bevorzugten Editor in die Programmvoreinstellungen eintragen, wenn man mit dem Standard (`xemacs`) nicht zufrieden sein sollte. Dazu klickt man mit der mittleren Maustaste auf die "Editieren" Schaltfläche und ändert den Editor-Eintrag. Nachdem man wieder im Hauptfenster ist, kann man die geänderten Einstellungen speichern, damit man sich diese Arbeit nicht immer wieder machen muß.



Die linke Schaltflächenleiste von `xtem`, sozusagen die Kommandozentrale für LaTeX

Nun wählt man über die Schaltfläche "Datei/Dateiverzeichnis-Auswahl" (per Linksklick) bzw. mittels **ALT f** die zu bearbeitende `.tex`-Datei. Mit **ALT e** wird der Editor gestartet und man kann sich ans Bearbeiten des Textes machen. Der Anfang sollte in unserem Fall ungefähr wie folgt aussehen:

<code>\documentclass[12pt,a4paper]{report}</code>	grundlegende Formatierungen, Art des Dokuments, Schriftgröße
<code>\usepackage{german,a4,fontenc,graphicx}</code>	zusätzliche Makros z.B. für deutsche Umlaute, Grafikeinbindung und das Papierformat
<code>\pagestyle{headings}</code>	auf jeder Seite wird oben die Seitenzahl und die Kapitelüberschrift mitgedruckt
<code>%\setlength{\topmargin}{-1cm}</code> <code>%\setlength{\textheight}{25cm}</code> <code>%\setlength{\textwidth}{17cm}</code> <code>%\setlength{\oddsidemargin}{-0.5cm}</code> <code>%\setlength{\evensidemargin}{-0.5cm}</code>	hiermit kann der bedruckbare Bereich der Seite beeinflusst werden
<code>\newcommand{\linux}{\textsf{Linux}}</code>	ein eigenes Makro, um das Wort Linux besonders zu formatieren
<code>\begin{document}</code>	hier beginnt das eigentliche Dokument
<code>\begin{titlepage}</code> <code>\title{\LaTeX, Textsatz für Sado-</code> <code>Masochisten?}</code> <code>\author{Frank Rennemann}</code> <code>\date{Irgendwo in Europa, \today}</code> <code>\end{titlepage}</code>	alle Angaben, die man im Titel haben will, stehen hier
<code>\maketitle</code>	und hiermit wird der Titel dann erzeugt
<code>\tableofcontents</code>	erzeugt das Inhaltsverzeichnis
<code>\listoffigures</code>	erzeugt ein Abbildungsverzeichnis
<code>\chapter{Ein paar Worte vorweg...}</code> Manch einer, der es nicht kennt, h"alt <code>\LaTeX{}</code> vielleicht für eine sehr exotische Klamottenart f"ur noch exotischere Gelegenheiten...	der eigentliche Text mit der ersten Überschrift
<code>\end{document}</code>	und hiermit ist der Text beendet

Tja, eigentlich ganz einfach, oder? Ganz oben werden mit `\documentclass` grundlegende Formatierungsoptionen gesetzt. `12pt` setzt die Schriftgröße, `a4paper` die Größe der Seite und `{report}` bedeutet, das es etwas umfangreicher als nur ein `{article}` wird. Dadurch wird eine zusätzliche Gliederungsebene möglich, bei `{article}` geht es mit `\section` los, bei `{report}` mit `\chapter`. Außerdem werden Kapitel natürlich immer auf einer neuen Seite begonnen, `\section` dagegen erzwingt keinen Umbruch.

Der nächste Befehl, `\usepackage` lädt noch einige zusätzliche Makros. Durch den Schalter `german` vereinfacht sich der Umgang mit Umlauten. Trotzdem ist es zu Anfang etwas lästig, statt eines ä ein "a zu schreiben. Aber wozu gibt es die Suchen/Ersetzen-Funktion bei Editoren :-). Einige Editoren ersetzen im LaTeX-Modus schon während des Schreibens die Umlaute. Mit dem `ß` ist es ähnlich, hier hilft einem

"s weiter.

Das `graphics`-Paket erlaubt es, Encapsulated Postscript Grafiken in den Text einzubinden. Diese EPS-Dateien lassen sich aus allen herkömmlichen Bitmap-Formaten (png, gif, jpeg etc.) z.B. mit gimp erzeugen. Eingebunden werden sie so:

```
\begin{figure}[h]
\includegraphics[height=5cm]{<Dateiname>.eps}
\caption{<Bildunterschrift>}
\end{figure}
```

Die `\caption` Anweisung erzeugt nicht nur die Bildunterschrift sondern wird auch für das Abbildungsverzeichnis ausgewertet.

Die im Beispiel auskommentierten `\setlength`-Anweisungen verändern den bedruckten Seitenbereich, LaTeX erzeugt rundherum ungefähr einen Zoll Rand. Bevor man seine Traktate zu Papier bringt, sollte man anhand eines Testausdrucks ausprobieren, wohin der eigene Drucker den Text denn zaubert. Mit Hilfe der `\setlength`-Anweisungen kann man dann die Feinjustage vornehmen. Das muß dann bis zum nächsten Druckerwechsel nicht mehr wiederholt werden.

Während des Schreibens wird einem wahrscheinlich auffallen, das man bestimmte Worte oder Phrasen des öfteren benutzt und evtl. sogar besonders formatiert haben möchte. Damit man das dann nicht immer wieder tun muß, kann man mit `\newcommand{<Aufruf>}{<Das hier soll später im Text stehen>}` eigene Textbausteine deklarieren, die danach im ganzen Text zur Verfügung stehen.

Nach den Deklarationen im Kopfbereich steht alles, was hinterher auch auf dem Drucker erscheinen soll, zwischen `\begin{document}` und `\end{document}`. Wie bei einem mehrseitigen Dokument üblich, fängt man natürlich mit dem Titel an. Die `\titlepage` wird innerhalb der `document`-Umgebung deklariert und danach mittels `\maketitle` erzeugt. `\tableofcontents` und `\listoffigures` tun genau, was man von ihnen erwartet und erzeugen ein Inhalts- und ein Abbildungsverzeichnis. Ein Tabellenverzeichnis erzeugt man analog mit `\listoftables`.

Dem aufmerksamen Leser werden die geschweiften Klammern hinter `\LaTeX` nicht entgangen sein sowie die Tatsache, daß sich nirgendwo ein `\newcommand` dafür befindet. Dankenswerterweise weiß LaTeX wie es heißt und bringt das Makro gleich mit, genau wie das für `\TeX`. Und die geschweiften Klammern sind immer dann nötig, wenn hinter dem Textmakro ein Leerzeichen kommen soll.

Genug herumeditiert, jetzt wollen wir uns das Ergebnis mal ansehen. Vorher müssen wir mit **ALT x** allerdings mindestens einen TeX-Lauf durchführen. Treten dabei Syntaxfehler zu Tage, werden die Fehlermeldungen mit Angabe der Zeile, in der der Fehler auftrat im unteren Teil von `xtem` ausgegeben. Ist alles glatt durchgelaufen, kommt man mit **ALT v** in die Vorschau der frisch erstellten DVI-Datei. Gefällt einem, was man sieht, kann man das Ergebnis mittels **ALT p** auch ausdrucken.

So, das war's schon für dieses Mal, viel Spaß beim Ausprobieren und nur nicht entmutigen lassen. Hat man den Dreh erstmal raus, will man von Textverarbeitungen wie StarOffice oder gar MS Word nichts mehr wissen.

Noch ein paar Informationsquellen

Wenn LaTeX installiert ist, findet sich irgendwo im Filesystem höchstwahrscheinlich auch die LaTeX-Kurzeinführung, `l2kurz.dvi` (unter SUSE Linux 9.1 in [/usr/share/texmf/doc/latex/general/l2kurz.dvi](http://usr/share/texmf/doc/latex/general/l2kurz.dvi)). Wem das nicht genügt, der sollte einfach auf dem Server der Dante-Vereinigung vorbeischaun, <http://www.dante.de>. Wer sich gern ausführlich mit dem Thema beschäftigen möchte, sollte den Erwerb eines Buches ins Auge fassen. Hier

war für mich immer Helmut Kopka's **LaTeX, Einführung Band 1, 3.Auflage, Pearson Studium** eine große Hilfe, sowohl als Referenz zum Nachschlagen, als auch einfach mal zum Stöbern, was eigentlich noch so alles mit LaTeX möglich ist.

Über die Arbeit mit `k1le` ist mittlerweile ein eigener Artikel auf diesem Portal erschienen, [LaTeX Textsatz mit k1le](#).

Viel Spaß beim Ausprobieren.

Nachtrag:

Zum Artikel erreichte mich ein Tipp von Stefan Wollny:

Solltest Du eine Ergänzung dazu planen: Füge doch bitte einen kurzen Hinweis auf das Komascript-Paket ein. Dies wurde von Markus Kohm erstellt und ist deutlich stärker auf die im deutsche Sprachraum vorherrschenden Konventionen abgestimmt. Hier noch zwei Links dazu:

- <http://people.freenet.de/kohm/markus/latex.html.de>
- <http://komascript.de/> 

LinuxKP.org 30.06.2004