Latex

https://iuk.one/1111-1001

Clemens H. **Cap**
ORCID: 0000-0003-3958-6136

Department of Computer Science
University of **Rostock**
Rostock,Germany
clemens.cap@uni-rostock.de

Version 1

Anforderungen wissenschaftlicher Textverarbeitung

- Hohe Qualität, Exaktheit & Ästhetik der Darstellung.
- Reproduzierbare Ergebnisse
- Zuverlässigkeit und Stabilität der Bedieneroberfläche
- Mehrsprachenfähig
- Reicher Satz von Font-Typen
- Aufwendiger Satz: Mathematisch & chemische Formeln
- Abbildungen, Diagramme und Tabellen
- Lange Dokumente: 400 Seiten und mehr
- Verweise: Quer, Formeln, Literatur, Abbildungen, Tabellen
- Verzeichnisse: Inhalt, Abbildungen, Tabellen, Index
- Ausdruck: Unabhängig vom Ausgabegerät
- Versand: Elektronisch möglich bei kompakte Dateien
- Spezifikationssprache statt Clickie-Buntie
- Format-Kompatibilität PS, EPS, TIFF, GIF, JPEG, XFIG, PNG, Mathematica

Lösung

Einzige Lösung: TEX & LATEX & Co.

Vorteile:

- Textsatzsystem das **wirklich** alles kann: Auch Primzahlen berechnen!
- Konzeption von einem der bekanntesten Informatiker, Donald Knuth.
- Basis-System (ziemlich sicher) fehlerfrei.

Nachteile: In manchen Themen sehr steile Lernkurve.

TEX: 5 Bücher von Donald Knuth	je 500 Seiten
Viele Programmier-Primitiva	mehr als 500
Viele Makropakete	rund 6.000
Bsp: tcolorbox-Handbuch:	540 Seiten
Bsp: TikZ und PGF-Handbuch:	1100 Seiten
Bsp: Beamer-Handbuch:	250 Seiten

- Basiskonzept: Informatik der 1970-er Jahre.

Fazit

LaTeX ist kein Textsatzsystem.
LaTeX ist eine Denkweise im Textsatz, die höchste Präzision anstrebt.

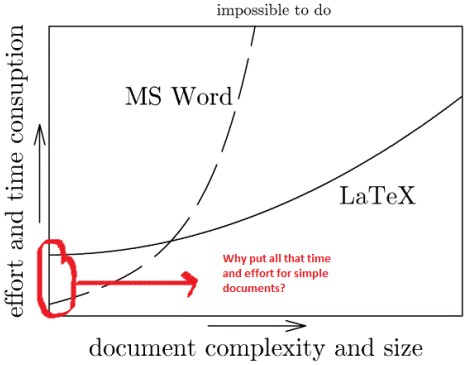

Abb. 1: Vergleich von Latex mit Word Rechte s. Anhang.

Vergleich

WYSIWYG: **W**hat **y**ou **s**ee **i**s **w**hat **y**ou **g**et

- Layout wird interaktiv gestaltet.
- Layout durch Folge von *Mausklicks* und *Mausschubsern* (sehr unklar) definiert.
- **Bsp:** Word, Pages, OpenOffice, ...

WYGIWYM: **W**hat **y**ou **g**et **i**s **w**hat **y**ou **m**ean

- Layout wird über Templates und Makros programmiert.
- Layout als Programm in TEX exakt festgelegt.
- Digitale Form der *Kunst* des Buchdrucks.
- **Bsp:** TEX, LATEX

Ziele dieses Kurses

Was sollen Sie hier lernen?

- Grundkenntnisse in[1] wissenschaftlicher Textverarbeitung.
- Grundkompetenzen im Satzsystem LaTeX.
- Einführung in das Programmsystem TeX.
- Eröffnen weiterer eigener Wege in TeXund LaTeX.

[1] Einzufügen: „in der *Kunst*“

Notwendige Werkzeuge

Text-Editor: Have it your way!

TEX Compiler: Übersetzt von TEX in Zielformate.
- Historisch: DVI (**d**e**v**ice **i**ndependent), PS (**P**ost**s**cript)
- Aktuell: PDF (**p**ortable **d**ocument **f**ormat)
- Verfügbar: Alles. Von SVG über PNG bis hin zu HTML und Mediawiki. Erfordert ggf. geeignete Treiber & Konverter.

Viewer: Anzeigeprogramm für das Zielformat; meist: PDF.

Werkzeuge: Rundes Dutzen Hilfsprogramme, meist gut integriert in IDEs.
- BibTEX, biber, makeindex, makeglossary, LuaTEX, eTEX, pdfTEX, ConTEXt, XeTEX

IDE: Integrationsumgebung bindet alles zusammen.

Makropakete: Über 6.000 Stück für jedes TEX-Problem. Zumeist Teil des Installationsbündels; sonst: CTAN.

Latex Umgebungen (1)

Linux: TEXCompiler & viele Werkzeuge, Teil vieler Linux Distris. (Empfehlung Cap)

- Einfach zu installieren, etwa via `apt-get`. Siehe: Debian & TUG
- Nicht immer die *aller*neuesten Makro-Pakete, reicht aber locker.

Linux: Als TEXLive von CTAN (Der Goldstandard. Empfehlung Cap)

- Installation ein wenig aufwendiger.
- Sorgfältige Trennung von ggf. vorhandener Linux- TEX-Distro nötig (Pfade!)
- Neueste Entwicklungen in LATEXMacros verfügbar.

Linux & Mac: IDEs

- Qt-basierte Umgebung: TEXstudio (Empfehlung Cap)
- Atom &Packages: latex language-latex pdf-view (Empfehlung Cap)
- MiKTEX Vergleich Miktex zu Texlive
- Eclipse-basierte Umgebung

Latex Umgebungen (2)

Online: LaTeX as a Service mit Web-Browser-Interface.

- Für 98% der Fälle ok.
- Aber: Nicht alle Werkzeuge angeboten & nicht alle Konfigurationen einstellbar.
- **Overleaf** Free trial, dann ab 9€/ Monat (Empfehlung Cap)
  Hilfreiche Features für Kollaboration
- Papeeria Freie Version und ab $5
- Latexbase Freie Version und ab $4.99

Antiempfehlungen:

MathJaX: Extrem abgespeckt, nur elementarstes LaTeX für den Browser.
Latex.js: Extrem abgespeckt, nur elementarstes LaTeX für den Browser.
LyX: Mäßig geglückter Versuch, WYSIWYG & LaTeXzu mischen.
Katex: Ähnlich wie MathJax

Windows: TeXstudio, MiKTeX, Visual Code
Seit EoL Windows 7 migriere ich weg von Microsoft, wo es geht: Deshalb, deshalb, deshalb...

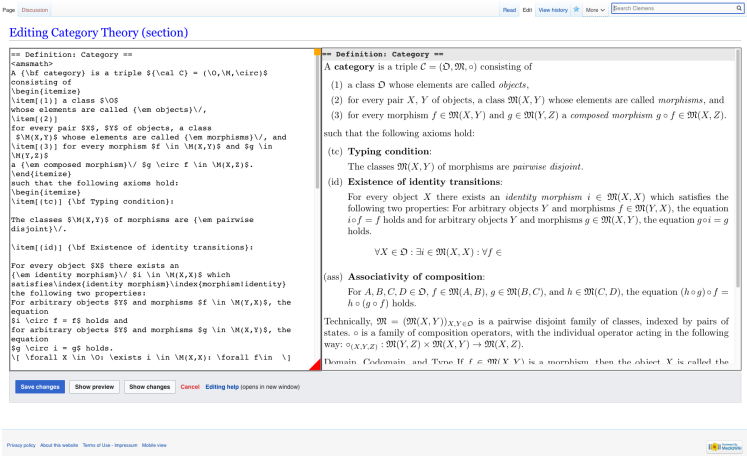

Abb. 2: Parsifal Mediawiki Extension von Clemens Cap. *Realtime live preview* als Mediawiki-Komponente mit Docker-Anbindung; https://github.com/clecap/Parsifal. Rechte s. Anhang.

Getting Started...

1. Werkzeug besorgen und einrichten.
2. Sich mit dem Werkzeug vertraut machen.
3. Hello World Dokument schreiben.
4. Dokument auf PDF Form bringen.

```
1 \documentclass{article}  % Kommentare beginnen beim Prozentzeichen
2 \begin{document}
3 Hello World!
4 \end{document}
```

Weitere Beispiele: Github: clecap/latex-examples

Weitere Übungen: iuk.one

Typischer Aufbau eines Dokuments

```
%%% Beginn der Präambel

<... Spezifikationen, die vor der Dokumentenklasse kommen müssen ...>
\documentclass[<Optionen>]{<classname>}  % Deklaration der Dokumentenklasse
\usepackage{<Paketname1>}
\usepackage[<Optionen>]{<Paketname2>}
<... eigene Makrodefinitionen...>

%%% Ende der Präambel und Beginn des eigentlichen Dokuments

\begin{document}                         % Anfang des eigentlichen Dokuments

<... eigentliches Dokument ...>

\end{document}                           % Ende des eigentlichen Dokuments
```

Src. 1: Typisches LaTeXDokument

Tipps: Werkzeug Einstellungen (1)

Build Directory: Separates Verzeichnis für Zwischen- & Ergebnis-Files.
- Im Werkzeug geeignet aktivieren.

SyncTEX: Nahtloses Springen zwischen TEX-Text und PDF-Resultat
- Im Werkzeug geeignet aktivieren.

Shell Escape: Erlaubt TEXZugang zur Shell zum Starten von Hilfsprogrammen.
- Macht Workflow „runder“, daher: Im Werkzeug geeignet aktivieren.
- Achtung: Unbekannte TEX-Programme können `rm -rF *` aufrufen.
- Infos zu Risiken und Nebenwirkungen.

Suchpfade: Zum Einbinden der Makro-Pakete und Fonts.
- Sind im Werkzeug (meist) geeignet aktiviert.
- Hilfreich: Eigene Werkzeichnis `~user/tex` in den Suchpfad setzen.
- Wenn nicht richtig gesetzt kann es Ärger geben; Debugging via kpathsea

Tipps: Werkzeug Einstellungen (2)

Tastatur-Anpassung: Ist notwendig.

- TEX-Arbeit benötigt häufig: `\{ } [ ] $ ~ _`
- Sollten auf Tastatur-Belegung leicht erreichbar sein: Bei Bedarf anpassen.
- **Mac**: Karabiner-Elements mit „German mapping for programming“ nutzen.
- **Linux**: xmodmap nutzen.
- **Windows**: kwT

Short-Cut-Einrichtung: Kann helfen.

```
1 \begin{itemize}
2 \item
3 \end{itemize}
```

- oder einfach `alt-i`
- Die meisten Werkzeuge erlauben Short-Cuts und Makros.

Fehlermeldungen

Sehr beruhigend...

Am besten betrachten Sie die Fehlermeldungen als eine Art Psycho-Test, mit dem herausgefunden werden soll, wie belastbar Sie sind.

Aus: R. Wonneberger, Kompaktführer LaTeX, Addison-Wesley, 1988.

Ursache von Fehlern

In **fallender Häufigkeit**:

1. Schreibfehler in Befehlswörtern oder Makros
2. Nicht-balancierte Klammern
3. Fehlende Argumente
4. Falsches mentales Modell von einem Konstrukt
5. Fehler in einem Makropaket
6. ~~Fehler in TEX~~ sind sehr sehr selten.

Fehler beheben

Gefühlte Statistik:

- In 95% der Fälle ist die Fehlermeldung selbsterklärend.
- In 5% der Fälle: Siehe Zitat von zuerst.

Schreibfehler:

- In 95% der Fälle ist der Fehler ein banaler Schreibfehler.
- IDE macht auf Schreib- und Strukturfehler aufmerksam.
- IDE kann mit *autocomplete* Schreibfehler vermeiden.
- Ggf. muß man IDE anpassen: Siehe Anleitung der jeweiligen IDE!

TeX Stackexchange mit rund 1/4 Million Antworten.
Praktisch jedes Problem trat bereits mehrfach auf und wurde dort mehrfach gelöst.
Hauptaufgabe: Gute Beschreibung bei der Suche.

Hartnäckige Fehler beheben

Fehler isolieren

- Bestimmte Zeilen auskommentieren mit %-Zeichen.
- Dokument früher beenden: `\end{document}` einfügen.
- Kopie anlegen und auf Fehlerstelle reduzieren.

Für jedes heikle Konstrukt: Ein **MWE** Minimal Working Example bauen.

Für jeden Fehler: Ein Minimal **Non**-Working Example bauen.

Liste häufiger Fehler

Fehlersuche

Ziele dieses Kurses

Mein erstes und zweites LaTeXProgramm

Werkzeuge

Schriften: Größen, Varianten, Arten, Hervorhebung, Farben

Zeilen: Abstände, Bindestriche, Akzente, Trennung

Paragraphen: Hurenkind, Schusterjunge

Satzelemente: Listen, Tabelle, Abbildungen

Dokumente: Arten, Teile

Verzeichnisse: Inhalt, Index, Literatur, Glossar

Mathematischer Formelsatz: Formeln, Gleichungen Symbole, Formelnummern, Matrizen

Informatik-Satz: Code, Diagramme, Automaten

Vortragsfolien

Diagramme, Abbildungen, Code-Einbindung

Codierung von Texten

Codierungen von Dateien: ASCII, windows-1252, ISO-8859-1, UTF-8.
Verschiedene Einstellungen von Editor und Darstellungsprogramm bewirken:

W◆rter, ◆berpr◆fung, Schaltfl◆che, ◆nderung, Dateigr◆◆e, ◆kosystem, Fu◆ball, K◆nstler, ...

W?rter, ?berpr?fung, Schaltfl?che, ?nderung, Dateigr??e, ?kosystem, Fu?ball, K?nstler, ...

WÃ¶rter, Ãœberprüfung, SchaltflÃ¤che, Ã„nderung, DateigrÃ¶ße, Ã–kosystem, FuÃŸball, KÃ¼nstler, ...

W□rter, □berpr□fung, Schaltfl□che, □nderung, Dateigr□□e, □kosystem, Fu□ball, K□nstler, ...

Abb. 3: Typische Fehler bei fehlerhaften Codierungen auf Webseiten Rechte s. Anhang.

Ein „deutsches kleines Umlaut a“, also ein ä, wird dargestellt:

ASCII	Nicht vorhanden	Codepage 437	84_{hex}
Windows-1252	$E4_{\text{hex}}$	EBCDIC 273	$C0_{\text{hex}}$
ISO 8859-1	$E4_{\text{hex}}$	HTML-Entity	`&auml;`
UTF-8	$C3A4_{\text{hex}}$	UTF-16	$00E4_{\text{hex}}$

Codierung von Texten in TEX: Das Problem

Situation:

- TEX war ursprünglich ein Programm mit 7-bit Textcodierung.
- Donald E. Knuth entwickelte 1977 (!) die Codierungen OT1, OML und OMS.
- Viele heutige gebräuchliche Codierungen und Fonts waren noch gar nicht existent!

Konfligierende Ziele

Ziel 1: Reproduzierbare Ergebnisse.

- TEX soll nicht versionsabhängig unterschiedliche Resultate erzeugen.
- TEX muß daher im Kern 7-bit bleiben.

Ziel 2: TEX soll neue, aktuelle Codierungen verwenden können.

Ziel 3: TEX soll neue, aktuelle Fonts verwenden können.

Codierung von Texten in TEX: Die Lösung

inputenc zur Auswahl der Zeichen-Codierung in TEX

- Bsp: `\usepackage[utf8]{inputenc}` für UTF8 Codierung.
- Bsp: `\usepackage[latin1]{inputenc}` für ISO Latin-1 Codierung.
- In LATEX seit 2018 automatisch `\usepackage[utf8]{inputenc}` geladen.

fontenc zur Auswahl der Font-Codierung in TEX

- `\usepackage[T1]{fontenc}` wählt neuere T1 Codierung.
- Ohne diese Spezifikation wird die alte OT1 Codierung geladen.
- OT1 hat keine Zeichen mit Akzent und emuliert Akzente daher nur. Manche Worte werden dann nicht automatisch getrennt.
- T1 hat Zeichen mit Akzent und trennt korrekt.

inputenc muß vor **fontenc** geladen werden.

Codierung von Texten in TEX: Das Rezept

Optimal: Zu Beginn immer:

```
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
```

Wenn **inputenc** vergessen: Nicht schlimm, LATEX lädt es selber.

Wenn **fontenc** vergessen: Nicht schlimm, einige Worte nicht automatisch getrennt.

Beim Entwurf eigener Makros kann das Thema aber ein wenig nerven...

Sonderzeichen und Akzente

Gravis	ò	`\`{o}`	Akut	ó	`\'{o}`
Zirkumflex	ô	`\^{o}`	Trema / Umlaut	ö	`\"{o}`
Doppelakut	ő	`\H {o}`	Tilde	õ	`\~{o}`
Cedille	ç	`\c {c}`	Ogonek	ą	`\k {a}`
Makron	ō	`\={o}`	Unterstrich	o̲	`\b {o}`
Oberpunkt	ȯ	`\.{o}`	Unterpunkt	ụ	`\d {u}`
Ring	å	`\r {a}`	Brevis	ŏ	`\u {o}`
Hatschek	Č	`\v {C}`	Verbindung	š	`\v {s}`
Querstrich l	ł	`\l`	Scharfes s	ß	`\ss`
i ohne Punkt	ı	`\i`	j ohne Punkt	ȷ	`\j`
i mit Trema	ï	`\"{\i }`	j mit Doppelakut	ȷ̋	`\H {\j }`

Einfache Sonderzeichen (deutsche Umlaute) funktionieren auch über UTF-8 Codierung.

LaTeX kann übrigens auch Arabisch, Chinesisch, Hebräisch, Japanisch, Koreanisch uvam.

Hier aber bitte keine Beispiele von mir erwarten ;-)

Mathematische Akzente

Zirkumflex	$\hat{o}$	`$\hat {o}$`
Wide Hat	$\widehat{pqrs}$	`$\widehat {pqrs}$`
Check	$\check{o}$	`$\check {o}$`
Tilde	$\tilde{o}$	`$\tilde {o}$`
Wide Tilde	$\widetilde{pqrs}$	`$\widetilde {pqrs}$`
Akut	$\acute{o}$	`$\acute {o}$`
Gravis	$\grave{o}$	`$\grave {o}$`
Dot	$\dot{o}$	`$\dot {o}$`
Double Dot	$\ddot{o}$	`$\ddot {o}$`
Brevis	$\breve{o}$	`$\breve {o}$`
Makron	$\bar{o}$	`$\bar {o}$`
Vektor	$\vec{o}$	`$\vec {o}$`

Lernen auf Vorrat?

Lernen auf Vorrat erscheint wenig sinnvoll!

Es gibt einfach zu viel.

Besser:

1. Learning by doing.
2. Lernen, die jeweils spezifische eigene Anforderung umzusetzen.
3. **Quellen:** Sehr sehr viele.
4. **Hinweis 1:** Reader
5. **Hinweis 2:** Overleaf