LaTeX für Naturwissenschaftler

Ansprechender Text- und Formelsatz von Abschlussarbeiten

2. LaTeX-Grundlagen: Dokumentstruktur, Befehle, Umgebungen

UNI FREIBURG

Albert-Ludwigs-Universität Freiburg

Dr. Till Biskup

Institut für Physikalische Chemie
Albert-Ludwigs-Universität Freiburg
Sommersemester 2018

Zentrale Aspekte

- LaTeX-Dokumente sind reine Textdokumente.
  Sie gliedern sich in Präambel und Dokumentkörper.
- Absätze werden durch Leerzeilen im Dokument markiert,
  Leerraum dient der Strukturierung des Quellcodes.
- LaTeX ist eine Programmiersprache für den Textsatz.
  Es unterscheidet zwischen Befehlen und Umgebungen.
- Fehler und Warnungen von LaTeX können kryptisch sein,
  sollten aber ernst genommen werden.
- Eigene Befehle helfen bei logischer Textauszeichnung
  und erleichtern Schreiben und Lesen von Dokumenten.

Übersicht

UNI FREIBURG

- LaTeX-Dokumente sind reine Textdokumente.
  - lassen sich mit jedem Texteditor bearbeiten
- Kodierung
  - ursprünglich ASCII 7-bit, inzwischen teilweise UTF-8
  - Details zur Eingabe von „Sonderzeichen“ später
- in der Regel überwiegt der Text
  - durchsetzt von Formatierungsanweisungen („Textauszeichnungen“, *markup*) an den Textsetzer
  - In den hier gezeigten Beispielen überwiegen LaTeX-Befehle.
- Dateiendung
  - normalerweise „`.tex`“, selten „`.ltx`“
  - dient im Wesentlichen der Wiedererkennung und erleichtert so die Syntaxhervorhebung durch Editoren

Gliederung eines LaTeX-Dokuments

Aufteilung in Präambel und Dokumentkörper

UNI FREIBURG

- LaTeX-Dokumente bestehen aus zwei Teilen
  - Präambel und Dokumentkörper
- Präambel
  - beginnt mit der Definition einer Dokumentklasse
  - Einbinden von zusätzlichen Paketen
  - Definition von Befehlen
  - Definition von Metadaten (wie Titel etc.)
  - hier darf kein Text stehen, der direkt ausgegeben wird
- Dokumentkörper
  - von entsprechenden Befehlen umschlossen (formen eine Umgebung, später mehr)
  - eigentliches Dokument
  - was nach `\end{document}` steht, wird ignoriert

Ein minimales LaTeX-Dokument

„Hello world“ mit LaTeX

UNI FREIBURG

Listing 1: „Hello world“ in LaTeX

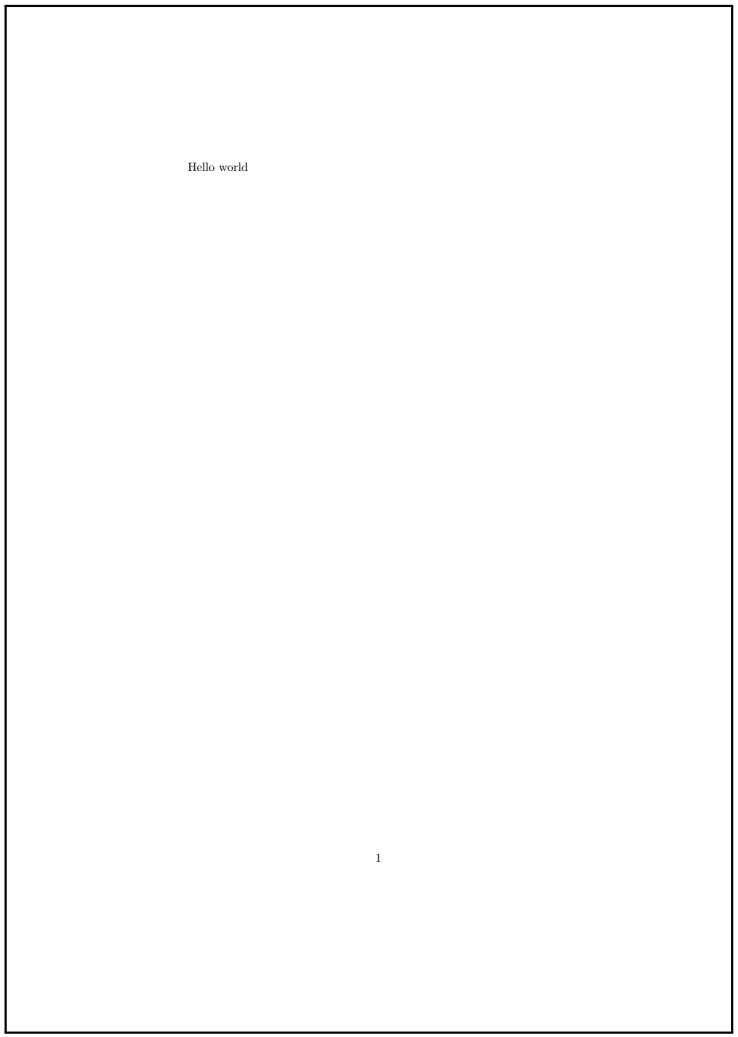
```
\documentclass{article}

\begin{document}
Hello world
\end{document}
```

- Präambel
  - besteht hier nur aus der Definition der Dokumentklasse
- Dokumentkörper
  - wird durch ein Befehlspaar (Umgebung) umschlossen: `\begin{document}...\end{document}`
  - enthält den Text, der ausgegeben wird

Ein minimales LaTeX-Dokument

„Hello world“ mit LaTeX



Anmerkungen

- Papierformat
  - Letter statt DIN A4
- Ränder
  - vergleichsweise groß
  - entsprechen nicht den üblichen Wünschen
- Seitenzahl
  - automatisch eingefügt

☛ LaTeX kommt aus den USA.

☛ Anpassungen an deutsche Gepflogenheiten notwendig

Formatierung des LaTeX-Quellcodes

Grundlegende Aspekte zum Schreiben von Text

UNI FREIBURG

- Absätze werden durch *Leerzeilen* markiert.
  - Innerhalb eines Absatzes sind Zeilenumbrüche erlaubt.
  - Viele Editoren brechen Zeilen automatisch um.
- Mehrere Leerzeilen werden als eine Leerzeile gewertet.
  - Leerzeilen eignen sich zur Formatierung des *Quellcodes*.
- Die Zahl der Leerzeichen zwischen Wörtern ist egal.
  - Horizontale Quellcode-Formatierung ist möglich.
  - Das erste Leerzeichen nach Befehlen wird „verschluckt“.
- geschützte Leerzeichen
  - können z.B. durch die Tilde (~) eingegeben werden
- Worttrennungen erfolgen meistenteils automatisch.
  - auf korrekte Spracheinstellungen achten

Formatierung des LaTeX-Quellcodes

Grundlegende Aspekte zum Schreiben von Text

UNI FREIBURG

> *Irgendwann ist dir der LaTeX-Quellcode lieber als ein normales Word-Dokument.*
>
> – Götz Pilarczyk

- LaTeX erfordert eine logische Textauszeichnung.
  - Die logische Dokumentstruktur ist meist offensichtlich.
  - mitunter immens hilfreich beim Verständnis des Textes
- Quellcode kann und sollte *übersichtlich* gestaltet werden.
  - Für LaTeX gilt das Gleiche wie für's Programmieren: Quellcode wird häufiger gelesen als geschrieben.
  - Vorteil von LaTeX: schön formatierte Ausgabe (PDF-Datei)

☛ Details zur Quellcode-Formatierung gibt's in der Vorlesung „Programmierkonzepte in der Physikalischen Chemie".

Hinweis

LaTeX ist nur bedingt eine Programmiersprache im herkömmlichen Sinn, eher mit Makros in C vergleichbar.

nachfolgend angesprochene Aspekte

- Befehle
- Umgebungen
- Zeichen mit besonderer Bedeutung

☛ Der zweistufige Prozess der Dokumenterzeugung – Quellcode, kompilieren – erinnert ans Programmieren.

Offensichtlich ist TeX aber Turing-vollständig.

Listing 2: Allgemeine Struktur eines Befehls in LaTeX

```
\Befehlsname[optionales Argument]{Argument}
```

- Befehle fangen mit einem „\“ an.
- Befehlsnamen
  - müssen mit einem Buchstaben anfangen
  - dürfen keine Sonderzeichen enthalten
- Befehle können, müssen aber keine Argumente haben.
- Argumente können obligatorisch oder optional sein.
  - obligatorische Argumente in geschweiften Klammern
  - optionale Argumente in eckigen Klammern

Listing 3: Beispiele für Befehle in LaTeX

```
\today

\emph{hervorgehoben}

\parbox{Breite}{Inhalt}

\item[Markierung]

\rule[Offset]{Breite}{Hoehe}
```

- Hinweis zu Argumenten
  - Reihenfolge mehrerer Argumente meist nicht intuitiv
  - je weniger Argumente, desto besser
  - bei Befehlen ohne Argument wird das erste Leerzeichen als Befehlsende interpretiert und „verschluckt“. Abhilfe: `{}`

Listing 4: Allgemeine Struktur einer Umgebung in LaTeX

```
\begin{Umgebung}
...
\end{Umgebung}
```

Umgebungen können ebenso wie Befehle obligatorische und optionale Argumente haben:

Listing 5: Allgemeine Struktur einer Umgebung mit Argumenten in LaTeX

```
\begin{Umgebung}[optionales Argument]{Argument}
...
\end{Umgebung}
```

Listing 6: Beispiele für Umgebungen in LaTeX

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

\begin{itemize}
\item ...
\end{itemize}

\begin{minipage}{Breite}
...
\end{minipage}

\begin{minipage}[Anordnung]{Breite}
...
\end{minipage}
```

Sonderzeichen mit spezieller Bedeutung

Eine unvollständige Liste

UNI FREIBURG

- \
  - Kennzeichnung von Befehlen
  - steht vor allen Befehls- und Umgebungsnamen
- &
  - Trenner von Tabellenspalten
  - dient dem Ausrichten mathematischer Formeln
- \\
  - manueller Zeilenumbruch (*sparsam* einsetzen!)
  - kann optional eine Länge übergeben bekommen
- %
  - Kommentarzeichen
  - alles bis zum Zeilenende wird ignoriert

Sonderzeichen mit spezieller Bedeutung

Eine unvollständige Liste

UNI FREIBURG

- $
  - Umschaltung in den mathematischen Modus
  - zur Erzeugung mathematischer Formeln im Text
- #
  - Ersetzungszeichen zur Parameterübergabe bei (eigenen) Befehlsdefinitionen
- ~
  - geschütztes Leerzeichen
  - verhindert Zeilenumbrüche
- { }
  - zur Gruppierung von Bereichen
  - werden normalerweise nicht ausgegeben

Sonderzeichen mit spezieller Bedeutung

TEX-Fehlermeldungen bei der unerlaubten Verwendung

UNI FREIBURG

Die unerlaubte Verwendung mancher Sonderzeichen führt zu entsprechenden Fehlermeldungen von TEX:

Listing 7: Fehlermeldung bei der Verwendung des Zeichens „#“

```
! You can't use `macro parameter character #' in vertical
    mode.
```

Listing 8: Fehlermeldung bei der Verwendung des Zeichens „&“

```
! Misplaced alignment tab character &.
```

- Gilt nicht für alle Sonderzeichen.
- Manche Sonderzeichen liefern „nur“ seltsame Ergebnisse.

Sonderzeichen mit spezieller Bedeutung

Varianten, sie im Text zu verwenden

UNI FREIBURG

- vorangestellter Backslash
  - Beispiele: `\&`, `\%`, `\$`, `\#`, `\{`, `\}`
  - in vielen Programmiersprachen übliche Variante
  - funktioniert nicht für alle Sonderzeichen in LaTeX
- explizite Befehle
  - benötigen ggf. ein zusätzliches Paket
  - Beispiele: `\textbackslash`, `\textasciitilde`

Tipp

Es gibt eine ausführliche Liste mit in LaTeX zur Verfügung stehenden Symbolen. Details auf der Webseite zum Kurs.

„Backslash“ ist gemäß Duden mittlerweile ein gültiges deutsches Wort...

Praktischer Umgang mit Sonderzeichen

Nur wenige der Zeichen tauchen im normalen Text auf.

UNI FREIBURG

Häufig auftretende Sonderzeichen

- %
  - Kommentarzeichen: alles Folgende wird ignoriert
- &
  - wirft einen TEX-Fehler bei der unerlaubten Verwendung
- { }
  - im mathematischen Modus gerne als Klammern verwendet
- _
  - zur Tieferstellung im mathematischen Modus verwendet
  - nur im mathematischen Modus gültig

☛ alle durch vorangestellten Backslash verwendbar

Fehlermeldungen und Warnungen

Grundsätzliches

UNI FREIBURG

- TEX und LATEX haben eine eingebaute Fehlerbehandlung
  - LATEX-Durchlauf liefert Ausgabe auf der Kommandozeile
  - Ausgabe in Editoren/IDEs oft in Konsolenfenster
- Unterscheidung zwischen Fehlern und Warnungen
  - Fehler führen zur Unterbrechung des LATEX-Compilerlaufs
  - Warnungen werden nur ausgegeben

Grundregel

Fehler des LATEX-Compilers müssen, Warnungen sollten immer ernst genommen werden.

Umgang mit Fehlern

Tipps aus der eigenen Erfahrung

UNI FREIBURG

- Fehlermeldungen sind mitunter kryptisch.
  - nicht abschrecken lassen
  - relevante Informationen oft einfach herausfilterbar
- meist Angabe der problematischen Zeile
  - hilfreich beim Auffinden/Einkreisen des Fehlers
  - kann je nach Situation irreführend sein
- manche Fehler(meldungen) sind offensichtlich
  - falsch geschriebene Befehlsnamen
  - vergessene Klammern
- pragmatischer Ansatz
  - Löweneinfangmethode: Fehler systematisch einkreisen
  - betroffene Codeteile ggf. auskommentieren

Warnungen, die die Typografie betreffen

LaTeX kann sehr pingelig sein.

UNI FREIBURG

Listing 9: Warnung wegen zu langer Zeile(n)

```
Overfull \hbox (2.59183pt too wide) in paragraph at lines
    63--86
```

Listing 10: Warnung wegen zu großer Wortabstände in einer Zeile

```
Underfull \hbox (badness 10000) has occurred while \output
    is active
```

- in jedem Fall ernst nehmen
  - insbesondere bei größeren Werten für „*overfull hbox*“
- typische Abhilfe
  - Silbentrennung manuell verbessern

Warnungen zu Marken und Referenzen darauf

LaTeX weist den Autor hilfreich auf Probleme hin.

UNI FREIBURG

Listing 11: Warnung wegen undefinierter Referenz

```
LaTeX Warning: Reference `ch:code' on page 11 undefined on
    input line 120.
```

Listing 12: Warnung wegen einer mehrfach definierten Marke (Labels)

```
LaTeX Warning: Label `fig:orbitale-zustaende' multiply
    defined.
```

- in jedem Fall beheben
  - führen zu fehlerhaften oder mit „?“ versehenen Referenzen
  - Warnungen sind eindeutig
- typische Ursache: copy&paste

Zusammenfassung von Warnungen

Am Ende der Ausgabe und der Log-Datei

UNI FREIBURG

- Warnungen hindern LaTeX nicht an der Übersetzung.
  - gehen schnell in der Ausgabe auf der Konsole unter
  - deshalb Zusammenfassung am Ende der Ausgabe
- Log-Datei
  - Endung: `log`
  - liegt neben der LaTeX-Datei
  - wird bei jedem LaTeX-Durchlauf erzeugt
  - enthält die komplette Ausgabe des LaTeX-Durchlaufs

- IDEs bereiten Warnungen meist übersichtlich auf.
- Unix-Werkzeuge auf der Kommandozeile sind hilfreich.
  - (komplexes) Durchsuchen der Logdatei nach Mustern
- ggf. hilft ein weiterer LaTeX-Durchlauf

IDE – integrierte Entwicklungsumgebung (*integrated development environment*)

Zusammenfassung von Warnungen

Beispiele

UNI FREIBURG

Listing 13: Globale Warnung wegen undefinierter Referenzen

```
LaTeX Warning: There were undefined references.
```

Listing 14: Globale Warnung wegen mehrfach definierter Marken (Labels)

```
LaTeX Warning: There were multiply-defined labels.
```

Listing 15: Hinweis, dass sich Marken geändert haben könnten

```
LaTeX Warning: Label(s) may have changed.  Rerun to get
    cross-references right.
```

- ähnliche Warnung für nicht aufgelöste Literaturverweise

Umgang mit Fehlern und Warnungen

Ein pragmatischer Ansatz

UNI FREIBURG

- Fehler
  - müssen immer behoben werden
  - sind so gut wie immer Anwenderfehler
    (TEX und LATEX enthalten quasi keine Fehler mehr)
- Warnungen zur Typografie
  - sollten erstgenommen werden
  - können im Einzelfall ignoriert werden
- Warnungen zu Marken und Referenzen
  - sollten immer ernst genommen werden
  - führen zu falschen oder mit „?“ versehenen Referenzen
- Warnungen zu Literaturverweisen
  - sollten immer ernst genommen werden
  - führen zu mit „?“ versehenen Verweisen

Gründe für die Definition eigener Befehle

Wieder einmal eine persönliche Sicht...

UNI FREIBURG

> *Die Grenzen meiner Sprache bedeuten die Grenzen meiner Welt.*
>
> – Ludwig Wittgenstein

- semantische Textauszeichnung
  - das Wesen ist wichtig, nicht das Aussehen
  - kann (LaTeX)-Code sehr viel lesbarer machen
- Faulheit
  - Wiederkehrende Dinge lassen sich als Befehl definieren.
- Konsistenz
  - Formatierung wird zentral festgelegt
  - in der Präambel – oder in (eigenen) Paketen

Wittgenstein: *Tractatus logico-philosophicus*. Suhrkamp, Frankfurt am Main 1963, Nr. 5.6

Beispiele für den Einsatz eigener Befehle

Erfahrungen aus der eigenen langjährigen Praxis

UNI FREIBURG

- Differentialoperator, imaginäre Einheit, Eulersche Zahl
  - sollten immer aufrecht gesetzt werden
- fremdsprachige Begriffe
  - werden normalerweise kursiv gesetzt
- Variablennamen etc. bei der Programmierung
  - werden oft in dicktengleicher Schrift gesetzt
- Name von Programmen
  - werden oft serifenlos gesetzt
- Vektoren fett und kursiv statt mit Pfeil
  - Beispiel für die Umdefinition eines Befehls

☛ Fokus auf *semantischer* Textauszeichnung

Befehle für die Definition eigener Befehle

Allgemeine Aspekte

UNI FREIBURG

- Möglichkeiten
  - sowohl Befehle als auch Umgebungen
  - Neudefinition und Umdefinition existierender Strukturen
  - mit und ohne Parameter (optionale Parameter schwierig)
- Definition neuer Strukturen
  - `\newcommand`
  - `\newenvironment`
- Umdefinition existierender Strukturen
  - `\renewcommand`
  - `\renewenvironment`

- sprechende Namen verwenden
- auf Kollision mit existierenden Befehlsnamen achten

Erstellung eigener neuer Befehle

Allgemeine Struktur

UNI FREIBURG

Listing 16: Befehl zur Erstellung eigener Befehle in LaTeX

```
\newcommand{\Befehlsname}{Definition}

\newcommand{\Befehlsname}[Argumentanzahl]{Definition}
```

- Verwendung von Argumenten
  - Zugriff auf Argumente in der Befehlsdefinition: `\#1`, `\#2`, ...
  - Zahl der Argumente immer minimal halten

Tipp

Leerzeichen nach der Ausgabe *nicht* mit im Befehl codieren

Erstellung eigener neuer Befehle

Ein konkretes Beispiel: Operatoren

UNI FREIBURG

Listing 17: Befehle für Operatoren im mathematischen Modus

```
\newcommand{\op}[1]{\hat{#1}}   % Operator (allgemein)

\newcommand{\ham}{\mathcal{H}} % Hamilton-Funktion

\newcommand{\hamop}{\op{\ham}} % Hamilton-Operator
```

- allgemeiner Befehl für Operatoren: `\op`
  - funktioniert nur im mathematischen Modus
  - erfordert zwingend ein Argument
  - setzt ein „Dach“ über sein Argument
- Befehle für Hamilton-Funktion und Hamilton-Operator
  - kalligrafisches H mit `\ham`: $\mathcal{H}$
  - Hamilton-Operator mit `\hamop`: $\hat{\mathcal{H}}$

Überschreiben existierender Befehle

Allgemeine Struktur

UNI FREIBURG

Listing 18: Befehl zum Überschreiben existierender Befehle in LaTeX

```
\renewcommand{\Befehlsname}{Definition}

\renewcommand{\Befehlsname}[Argumentanzahl]{Definition}
```

- grundsätzliche Bedeutung
  - Variante, Aussehen vorgegebener Strukturen zu verändern
  - oft innerhalb von Paketen und Klassen eingesetzt
- durchaus real auftretender Anwendungsfall
  - Bsp.: Darstellung von Vektoren im Formelsatz

☛ Versehentliche Umdefinition eines nichtexistenten Befehls führt zu einer eindeutigen LaTeX-Fehlermeldung

Überschreiben existierender Befehle

Reales Beispiel: Vektoren umdefinierten (fett und kursiv)

UNI FREIBURG

Listing 19: Umdefinieren von Vektoren: fett und kursiv statt mit Pfeil

```
\usepackage{bm}
\renewcommand{\vec}[1]{\bm{#1}}
```

- Fettdruck in mathematischen Formeln
  - Es gibt mehrere Varianten.
  - Die beste Variante hängt nicht zuletzt von der jeweils verwendeten Schrift ab.
  - in jedem Fall über zusätzliches Paket zu lösen
- Beispiel
  - Vektor in herkömmlicher Schreibweise: $\vec{a}$
  - Vektor nach Umdefinition wie oben: $\boldsymbol{a}$

Erstellung eigener Umgebungen

Allgemeine Struktur

UNI FREIBURG

Listing 20: Befehl zur Erstellung eigener Umgebungen in LaTeX

```
\newenvironment{Umgebungsname}%
  {Definition zu Beginn}{Definition am Ende}

\newenvironment{Umgebungsname}[Argumentanzahl]%
  {Definition zu Beginn}{Definition am Ende}
```

- Grundsätzliches zur Syntax
  - Definition von Beginn und Ende einer Umgebung
  - Beginn und Ende sollten zusammenpassen
  - Formatierungen nur innerhalb der Umgebung gültig
- sinnvolle Einsatzgebiete
  - semantische Textauszeichnung
  - viele der Strukturen auf diesen Folien . . .

Erstellung eigener Umgebungen

Reales Beispiel: Aufzählungsliste mit eigenen Markierungen

UNI FREIBURG

Listing 21: Definition einer Umgebung für zentrale Aspekte (vereinfacht)

```
\usepackage{fontawesome} % Sonderzeichen (font awesome)

\newenvironment{keyaspects}%
    {\begin{itemize}\renewcommand{\labelitemi}{\faKey}}%
    {\end{itemize}}
```

Listing 22: Nutzung der oben definierten Umgebung

```
\begin{keyaspects}
\item Beispiel der Umgebung \texttt{keyaspects}
\end{keyaspects}
```

- Beispiel der Umgebung `keyaspects`

Überschreiben existierender Umgebungen

Allgemeine Struktur

UNI FREIBURG

Listing 23: Befehl zum Überschreiben existierender Umgebungen in LaTeX

```
\renewenvironment{\Befehlsname}%
  {Definition zu Beginn}{Definition am Ende}

\renewenvironment{\Befehlsname}[Argumentanzahl]%
  {Definition zu Beginn}{Definition am Ende}
```

- seltener Anwendungsfall
  - setzt ggf. tiefere Kenntnis von LaTeX voraus
  - originale Umgebung sollte vorher verstanden worden sein
- kommt normalerweise in Klassen etc. vor
  - LaTeX definiert eine Reihe von Strukturen vor
  - können entsprechend angepasst werden
  - Vorteil: Nutzer kennt die Standard-Strukturen

Ort(e) für die Definition eigener Befehle

Von der Präambel zu eigenen Paketen und Klassen

UNI FREIBURG

- Präambel eines Dokuments
  - Normalfall
  - Befehle/Umgebungen im gesamten Dokument verfügbar
  - ggf. sinnvoll gruppieren und kommentieren
- ausgelagerte Dateien
  - werden mit `\input{}` geladen
  - normalerweise in der Präambel eingebunden
  - erhöhen die Übersicht
  - erleichtern die Verwendung in mehreren Dokumenten
- eigene Pakete (und Klassen)
  - für den normalen Anwender selten
  - Schreiben eigener LaTeX-Pakete erfordert eine tiefere Beschäftigung mit LaTeX

Zentrale Aspekte

- LaTeX-Dokumente sind reine Textdokumente.
  Sie gliedern sich in Präambel und Dokumentkörper.
- Absätze werden durch Leerzeilen im Dokument markiert,
  Leerraum dient der Strukturierung des Quellcodes.
- LaTeX ist eine Programmiersprache für den Textsatz.
  Es unterscheidet zwischen Befehlen und Umgebungen.
- Fehler und Warnungen von LaTeX können kryptisch sein,
  sollten aber ernst genommen werden.
- Eigene Befehle helfen bei logischer Textauszeichnung
  und erleichtern Schreiben und Lesen von Dokumenten.