

# 1 Grundlagen L<sup>A</sup>T<sub>E</sub>X

## 1.1 Einführung

Diese Übersicht soll das Schreiben nachträglich erleichtern. Für weitere Details schaue bitte in die .tex Datei hinein! L<sup>A</sup>T<sub>E</sub>X ist eine Sammlung von Makros die in TeX programmiert sind. Das zu Grunde liegende TeX-System ist eine Programmiersprache. Es gibt entsprechend für den Anwender viele Elemente einer Programmiersprache um den Textsatz zu gestalten. Mithilfe von *Variablen* können alle Format-Eigenschaften eines Dokuments programmatisch geändert werden. Hierbei gibt es verschiedene *Variablentypen*. Wichtig sind die *Längen-* und *Zählervariablen*. Diese Variablen können über L<sup>A</sup>T<sub>E</sub>X-Befehle verändert werden. Auf TeX-Ebene gibt es Kontrollstrukturen, also *if then* Anweisungen, *Schleifen* und noch vieles mehr.

## 1.2 Böse Sachen

Zuerst einmal Sachen die **NICHT** schön sind:

1. Manuelle Leerzeichen mit \
2. Andauernd \textbf{...} verwenden. Lieber Betonungen (emph...)!

## 1.3 Struktur

Der Textkörper wird mit Abschnitts-Befehlen strukturiert. Diese unterscheiden sich nach Typ des Dokumentes (z.B. **article**). Jedoch gelten diese Befehle für alle:

- \section
- \subsection
- \subsubsection

Für **book** gilt wiederum:

- \part
- \chapter
- \paragraph
- \subparagraph

## 1.4 Zeichen

Folgende Zeichen können **nicht** einfach so eingetippt werden:

- & % \$ # \_ { } ~ ^

Diese können mit der Kombination aus dem \ Befehl und dem jeweiligen Zeichen erzeugt werden (*Escape-Befehl*). Um Leerzeichen zu forcieren kann \@ benutzt werden. Ein Bindestrich kann mit -- eingefügt werden.

## 1.5 Textformatierungen

L<sup>A</sup>T<sub>E</sub>X unterstützt folgende physische Textformatierungen:

- `\textbf{bold face}` – **bold face**
- `\textit{italics}` – *italics*
- `\textsl{slanted}` – *slanted*
- `\textsc{small caps}` – SMALL CAPS

## 1.6 Schriftgrößen

Folgende lokalen Schriftgrößenänderungen sind in L<sup>A</sup>T<sub>E</sub>X möglich:

1. `\tiny`
2. `\scriptsize`
3. `\footnotesize`
4. `\small`
5. `\normalsize`
6. `\large`
7. `\Large`
8. `\LARGE`
9. `\huge`
10. `\Huge`

Um die Schriftgröße für das gesamte Dokument zu ändern gibt es Präambel-Befehle.

## 1.7 Umgebungen

Eine Umgebung schließt einen Text mit `\begin{Umgebung}` und `\end{Umgebung}` ein. Innerhalb der Umgebung gelten eigene und spezielle Regeln. Wichtige Beispiele sind:

- `enumerate` (Aufzählung)
- `itemize` (Liste)
- `equation` (mathematische Gleichung)
- `description` (Lexikon ähnliche Aufzählung)

## 1.8 Listings

Listings enthalten nicht-formatierten Text, der wichtig zur Darstellung von Quellcode jeglicher Programmiersprachen ist.

```
int main(int argc, char* argv[])
{
    printf("Hello World!");
}
```

Mit den Optionen `\lstset{language=C}`, sowie `\begin{lstlisting}[frame=single]` können die Programmiersprache festgelegt und der Texrahmen aktiviert werden.

## 1.9 Externe Dokumente

Mithilfe des `\input{file}` Befehls können weitere *.tex* Dateien in das Dokument so eingefügt werden, als würde der Inhalt der Datei im Originaldokument stehen. Dabei können die Pfadangaben *relativ* oder *absolut* sein. Den Unterdokumenten muss eine extra Zeile am Anfang eingefügt werden, die auf das Master Dokument hinweist. Alternativ kann im *TexMaker* das Master Dokument über das Menü Optionen eingestellt werden.

## 1.10 Längen

Mithilfe des `\setlength{variable}{länge}` Befehls können die in **TeX** auftretenden Längen verändert werden. **Alle Längen haben eine Einheit!** Wichtige Variablen sind zum Beispiel:

- `\parindent` - Definiert den Einzug eines Absatzes
- `\parskip` - Definiert den Abstand zwischen zwei Absätzen
- `\baselineskip` - Der Zeilenabstand

**Anwendungsbeispiele:**

1. `\setlength{\parindent}{10pt}` - Einzug auf 10pt setzen
2. `\setlength{\parskip}{0.5\baselineskip}` - Der Abstand zwischen zwei Absätzen wird auf die Höhe einer halben Zeile gesetzt.

*Relative Längenangaben* (siehe 2. Beispiel) beziehen sich auf einen Referenzwert. Dieser Bezug erfolgt mithilfe einer Multiplikation (in **LaTeX** wird dazu der Multiplikator einfach direkt an den Referenzwert geschrieben). Eine Addition geht nicht. Das wäre Stretching! *Absolute Längenangaben* werden mit Maßzahl und Einheit (**WICHTIG!**) direkt angegeben. Es ist sinnvoll **immer relative Längenangaben** zu nutzen, denn bei einer Änderung des Schriftgrads von 10pt auf bspw. 12pt ändert sich der Abstand zwischen den Absätzen genau richtig mit. Auch erweist sich dies bei der Breite von Bildern und bei der Spaltenweite

von Tabellen als besonders hilfreich. Mithilfe des `\the` Befehls können Längen (Wert der Variablen) angezeigt werden. Es gibt viele verschiedene Längeneinheiten:

- Punkt - **pt** - entspricht: **1pt**
- Millimeter - **mm** - entspricht: **2.84pt**
- Zentimeter - **cm** - entspricht: **28.4pt**
- Inch - **in** - entspricht: **72.27pt**

**Wichtiger Tipp:**

Die wichtigsten Längen für die Gestaltung einer Seite können über den `\layout{}` Befehl angezeigt werden. Dazu muss aber vorher das Package *layout* mit `\usepackage{}` geladen werden. Interessante weitere Längen sind:

- `\columnsep`, `\columnwidth` - Bei der mehrspaltigen Gestaltung
- `\tabcolsep` - Für Tabellen
- `\itemsep` - Für Listen

Längen lassen außerdem über diverse Befehle verändern:

- `\setlength{\länge}{wert}` - Setzt eine Länge auf einen Wert
- `\addtolength{\länge}{wert}` - Addiert eine Länge zu einem Wert
- `\settowidth{\länge}{beispieltext}` - Breite des Beispieltexts auf `\länge` setzen
- `\settoheight{\länge}{beispieltext}` - Höhe des Beispieltexts auf `\länge` setzen
- `\settodepth{\länge}{beispieltext}` - Tiefe des Beispieltexts auf `\länge` setzen

Bei letzteren Befehlen wird der Beispieltext jedoch **NICHT** angezeigt! Interessant ist, dass der Beispieltext selbst das Ergebnis eines Befehls sein kann.

Um ein schöneres Layout zu erreichen streckt L<sup>A</sup>T<sub>E</sub>X einige Größen. Hierzu zählen der Abstand zwischen Wörtern für Blocksätze und die Abstände zwischen den Absätzen um die Seite optimal auszunutzen. Wenn das nicht klappt, meckert L<sup>A</sup>T<sub>E</sub>X über eine *overfull* oder *underfull* \vbox. **Eigene Längen definieren:** Eigene Längen können mit dem Befehl `\newlength` definiert werden. Dies kann zum Beispiel bei den Spaltenbreiten von Tabellen im ganzen Dokument, oder für eine einheitliche Breite von Bildern (automatisch skaliert) nützlich sein. Die Verwendung geschieht dabei wie folgt:

```
\newlength{\imgWidth}
\setlength{\imgWidth}{0.8\textwidth}
```

Zuerst wird die Länge *imgWidth* mit `\newlength` erzeugt und dann mit den Längenänderungsbefehlen gesetzt. Alle Bilder im Text können mit dieser Variablen auf 80% der Textbreite skaliert werden. Um alle Bilder neu zu skalieren, muss nur die 0.8 geändert werden.

## 1.11 Befehle

### 1.11.1 Eigene Kommandos

Mit dem Befehl `\newcommand` lassen sich eigene Befehle definieren.

```
\newcommand{\name}{definition}
\newcommand{\zB}{z.\,B.\, \@}
\newcommand{\abk}{Abkuerzung}
```

In der hier gezeigten Form dienen sie als Abkürzung für häufig auftretende Befehle, die sonst umständlich einzutippen wären. Die fortgeschrittene Variante des `\newcommand` Befehls gebraucht **Argumente**. So wird der selbst-definierte Befehl zur Funktion.

```
\newcommand{\makeFett}[1]{\textbf{#1}}
```

In der Definition wird über *#1* auf das erste Argument zugegriffen, über *#2* auf das zweite Argument, etc. Dies ist für **maximal 9 Argumente** möglich! Möchte man bestehende Befehle verändern, so kann man dies mit dem `\renewcommand` Befehl tun. Die Syntax ist dabei die gleiche, wie bei `\newcommand`. Hierbei können auch **Argumente** übergeben werden!

```
\renewcommand{\name}[1]{definition}
\renewcommand{\em}{\bfseries}
```

Bei dem Beispiel mit dem `\em` geht allerdings durch die Erneuerung die Schalter-Logik verloren.

### 1.11.2 Zähler

Zähler sind wie **Längen** ein Variablen-Typ mit denen in  $\text{\LaTeX}$  programmiert werden kann. Mit Zählern regelt  $\text{\LaTeX}$  (fast) alle Verweise im Dokument. Wie bei **Längen** können eigene Zähler definiert werden. Diese können auch mit diversen Befehlen bearbeitet werden. Hierbei gibt es eine Liste vordefinierter Zähler:

- part
- chapter
- section

- subsection
- subsubsection
- paragraph
- subparagraph
- page
- equation
- figure
- table
- footnote
- mpfootnote

Für Listen gelten außerdem diese Zähler:

- enumi
- enumii
- enumiii
- enumiv

Beispiel:

```
\stepcounter{enumi}
\addtocounter{section}{zahl}
\setcounter{equation}{zahl}

\newcounter{numDoener}

\thenumDoener % Formatierter Text

\value{numDoener} % Wert unformatiert
```

Der Befehl `\the` zeigt die Zähler an (ohne `\`). Das Besondere an `\value` ist, ist dass der unformatierte Wert zum Gebrauch in Rechnungen genutzt werden kann. **Zählerdarstellung:** Mithilfe folgender Befehle kann die Darstellung von Zahlen mithilfe des `\renewcommand` Befehls verändert werden.

1. `\arabic` – 1. 2. 3. ...
2. `\roman` – i. ii. iii. ...
3. `\Roman` – I. II. III. ...

4. `\alph` – a. b. c. ...

5. `\Alph` – A. B. C. ...

Zur Änderung **aller** Aufzählungen des Dokuments wird `\theenumi` usw. in der Präambel neu definiert. Dies ist notwendig, da  $\text{\LaTeX}$  anscheinend den Befehl indirekt zur Darstellung nutzt.

```
\documentclass{article}

\renewcommand{\theenumi}{\Roman{enumi}}

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

Zur Änderung einer einzigen Aufzählung wird `\theenumi` usw. direkt in der entsprechenden Aufzählung neu definiert.

```
\documentclass{article}

\begin{document}
  \begin{enumerate}

    \renewcommand{\theenumi}{...}

    ...
  \end{enumerate}
\end{document}
```