

L^AT_EX – Befehle, Zähler & Längen

Tobias Heintz

Montag, 8. Mai 2006

Inhaltsverzeichnis

1	Befehle	2
1.1	Format	2
1.2	Neudefinition	2
2	Umgebungen	3
2.1	Vorhandene Umgebungen	3
2.2	Deklarationen	3
2.3	Neue Umgebungen	3
3	Zähler	4
3.1	Vorgefertigte Zähler	4
3.2	Selbst definierte Zähler	4
3.3	Zähler bearbeiten	5
3.4	Ausgabe von Zählern	5
4	Längen	5
4.1	Einheiten	5
4.1.1	Klassische Längeneinheiten	5
4.1.2	Typographische Längeneinheiten	6
4.2	Flexible Längen	6
4.3	Längenbefehle	6
4.3.1	Vorgefertigte Längenbefehle	6
4.3.2	Selbst definierte Längenbefehle	6
4.4	Längen verändern	7
4.4.1	Absolute Längen	7
4.4.2	Abhängige Längen	7
5	Quellen	7

1 Befehle

L^AT_EX kennt viele Befehle von Haus aus:

- `\textbf`, `\textit`, `\textsl`
- `\`, `\-`, `\\`
- `\gamma`, `\chi`, `\theta`
- `\verb`, `\verb*`

Manche Befehle besitzen eine Variante, die mit einem Stern am Ende geschrieben wird. Dort wird die Funktionalität des Befehls noch „verstärkt“. `\verb` ist ein Beispiel. Bei der Schreibweise mit Stern werden auch die Leerzeichen sichtbar gemacht.

1.1 Format

Alle Befehle beginnen mit einem Backslash, gefolgt von entweder genau einem Sonderzeichen oder einer beliebigen Anzahl an Buchstaben. Befehle sind case-sensitive. Manche Befehle haben Argumente. Notwendige Argumente werden in geschweiften Klammern angegeben, optionale Argumente in eckigen:

- `\date{text}`
- `\footnote[num]{text}`

Befehle konsumieren Whitespace, der direkt dahinter auftaucht. Daher kann man Befehle mit `{}` abschließen, auch und gerade, wenn sie keine Argumente erwarten.

1.2 Neudefinition

Eigene Befehle sind Makros; sprich an der Stelle im Text, an dem ein Befehl auftaucht, wird er durch die Befehlsdefinition ersetzt. Neue Befehle werden meist in der Präambel des Dokuments definiert. Auch hier ist es möglich Argumente zu definieren. Diese werden innerhalb der Definition durch `#[number]` referenziert. Es können maximal 9 Argumente übergeben werden.

```
\newcommand{name}[args][default]{definition}
```

- `{name}` – Der Name des neuen Befehls, immer mit Backslash angeben.
- `[args]` – Die Anzahl der erwarteten Argumente.
- `[default]` – Wenn dieses Argument angegeben wird, wird das erste Argument des neuen Befehls als optional deklariert. Der hier angegebene Text wird als Default verwendet.

- `{definition}` – Die eigentliche Befehlsdefinition.

Beispiel:

```
\newcommand{\mycommand}[1]{--- \textit{#1} ---}
```

`\newcommand` kann nur für noch nicht existente Befehle verwendet werden. Befehle die schon definiert wurden kann man mittels `\renewcommand` überschreiben. `\providecommand` ist eine weitere Möglichkeit und definiert einen Befehl nur dann, wenn er noch nicht existiert.

2 Umgebungen

Umgebungen werden verwendet um ganze Blöcke von Text direkt zu formatieren.

2.1 Vorhandene Umgebungen

- `flushleft`, `flushright`
- `quote`, `quotation`
- `verbatim`, `verbatim*`
- `document`

Umschließen Text mit `\begin{name}` und `\end{name}`.

2.2 Deklarationen

Ermöglichen Befehle, die den Text direkt formatieren und meistens ohne Argumente definiert sind, auf eine begrenzte Fläche anzuwenden. Dazu wird genau vor den Befehl eine öffnende geschweifte Klammer gesetzt. Der Befehl wird bis zur nächsten schließenden geschweiften Klammer angewandt.

Beispiel:

```
{\bfseries Dieser Text erscheint fett.}
```

Für alle Deklarationen wird automatisch eine gleichnamige entsprechende Umgebung definiert.

2.3 Neue Umgebungen

Man kann eigene Umgebungen definieren, die genau wie die Vorgefertigten angewandt werden können. Auch hier können Argumente angegeben werden, die Referenzierung erfolgt wie bei den Befehlen.

```
\newenvironment{name}[args][default]{pre}{post}
```

- `{name}` – Der Name der neuen Umgebung. Dieser darf aus Buchstaben und Zahlen bestehen.
- `[args]` – Die Anzahl der Argumente.
- `[default]` – Wenn dieses Argument angegeben wird, wird das erste der regulären Argumente zu einem optionalen.
- `{pre}` – Der Text, der für jedes `\begin{name}` substituiert wird.
- `{post}` – Der Text, der für jedes `\end{name}` substituiert wird.

Beispiel:

```
\newenvironment{myenv}[1]{\vspace{10pt}\noindent\textbf{#1}\dotfill}\begin{itseries}}{\end{itseries}}
```

3 Zähler

Zähler funktionieren in \LaTeX wie in jeder anderen Programmiersprache. Es sind prinzipiell Variablen, die einen Integerwert halten.

3.1 Vorgefertigte Zähler

Einige der Zähler, die \LaTeX bereits definiert hat und automatisch inkrementiert:

- `page`
- `chapter`, `section`, `subsection`
- `equation`
- `enumi`, `enumii`, `enumiii`, `enumiv`

3.2 Selbst definierte Zähler

Neue Zähler können mittels des Befehl `\newcounter` erstellt werden. Hierbei ist es möglich einen Zähler als abhängig von einem anderen zu definieren. Das heißt, daß sobald der zugehörige Zähler verändert wird, der Neudefinierte wieder auf Null gesetzt wird. Diese Technik kommt beispielsweise bei den `section-` und `subsection-`Zählern zur Verwendung.

```
\newcounter{name}[link]
```

- `{name}` – Der Name des neuen Zählers. Dieser darf noch nicht vorhanden sein.
- `[link]` – Der Name des zu verknüpfenden Zählers. Dieser muss schon vorhanden sein.

3.3 Zähler bearbeiten

Es gibt einige Funktionen mit denen die Zählerwerte verändert werden können.

- `\addtocounter{name}{value}` – Addiert value zum angegebenen Zähler. Der Wert kann auch negativ sein.
- `\setcounter{name}{value}` – Setzt den angegebenen Zähler auf den Wert value.
- `\stepcounter{name}` – Erhöht den angegebenen Zähler um eins.

3.4 Ausgabe von Zählern

Möchte man die Darstellung der eingebauten Zähler verändern, kann man dies am einfachsten mit den `\the[countername]`-Befehlen tun. Für jeden der vorgefertigten Zähler gibt es eine solche Funktion, die man einfach mit Hilfe von `\renewcommand{\the[countername]}{[definition]}` überschreiben kann. In der Definition sollte der Zähler mittels einem der folgenden Befehle ausgegeben werden.

- `\arabic` – Arabische Ziffern (1, 2, 3...)
- `\roman`, `\Roman` – Römische Ziffern (i, ii, iii...)
- `\alph`, `\Alph` – Buchstaben (a, b, c...)
- `\fnsymbol` – Fußnotensymbole

Diese Befehle können auch verwendet werden, um neu erzeugte Zähler auszugeben.

4 Längen

Längen tauchen bei \LaTeX -Dokumenten an vielen Stellen auf. Sie können absolut oder als Längenbefehle angegeben werden.

- `\hoffset`, `\voffset`
- `\textheight`, `\textwidth`
- `\vspace`, `\hspace`

4.1 Einheiten

4.1.1 Klassische Längeneinheiten

Die klassischen Längeneinheiten, die auch aus dem Alltag bekannt sind.

- Inch (2,54 cm)
- cm
- mm

4.1.2 Typographische Längeneinheiten

- pt – 1/72 in ($\sim 0,35$ mm)
- em – ungefähr die Breite des 'M'
- ex – ungefähr die Höhe des 'x'
- pc – pica (12 pt)
- bp – big point (1/72 in)
- dd – didôt ($\sim 1,07$ pt)
- cc – cícero (12 dd)
- sp – scaled point (1/65536 pt)

4.2 Flexible Längen

Normalerweise haben Längen einen absoluten Wert, sie können aber auch flexibel definiert sein. Das heißt, sie sind so lang wie sie gebraucht werden. Das macht sich zum Beispiel `\dotfill` zugunsten, um Punkte einer beliebigen Länge auszugeben. Häufig bis zum Ende der Zeile.

4.3 Längenbefehle

Beim Angeben von Längen mit Hilfe von Längenbefehlen können diese multipliziert werden.

Beispiel:

```
\setlength{\baseparskip}{1.5\baseparskip}
```

4.3.1 Vorgefertigte Längenbefehle

Die weiter oben genannten werden von vielen weiteren ergänzt:

- `\parindent` – Legt die Weite des Einzugs fest.
- `\baselineskip` – Legt den Zwischenraum zwischen zwei Zeilen fest.
- `\parskip` – Legt den zusätzlichen Platz nach einem Absatz fest.

4.3.2 Selbst definierte Längenbefehle

Man kann auch selbst Längenbefehle definieren, das passiert mit `\newlength`:

```
\newlength{name}
```

- `{name}` – Legt den Namen des Längenbefehls fest, dieser muss neu sein.

4.4 Längen verändern

4.4.1 Absolute Längen

Längenbefehle können absolut mit den folgenden Befehlen gesetzt werden. Als Wert kann hier natürlich auch ein anderer Längenbefehl angegeben werden.

- `\setlength{name}{value}` – Setzt den angegebenen Längenbefehl auf value.
- `\addtolength{name}{value}` – Addiert value zu dem angegebenen Längenbefehl. value kann auch negativ sein.

4.4.2 Abhängige Längen

Längenbefehle können auch abhängig durch die folgenden Befehle gesetzt werden. Abhängig heißt in diesem Sinne, daß die erzeugte Länge von dem Argument abhängt.

- `\settowidth{name}{text}` – Setzt den angegebenen Längenbefehl auf die Weite des Arguments text.
- `\settoheight{name}{text}` – Setzt den angegebenen Längenbefehl auf die Höhe des Arguments text.
- `\settodepth{name}{text}` – Setzt den angegebenen Längenbefehl auf den Raum zwischen der Grundlinie und der unteren Enden der Buchstaben des Arguments text.

5 Quellen

<http://www.eng.cam.ac.uk/help/tpl/textprocessing/teTeX/latex/latex2e-html/>
<http://www.math.tu-berlin.de/Rechnerbetrieb/Forschungsbereich/LaTeX/tex/cookbook/>

<http://www.weinelt.de/latex/>

Tobias Oetiker, u. a.: The Not So Short Introduction to LATEX2e, Version 3.7