$\LaTeX\,2_\varepsilon$

Eine Einführung

Heiner Lamprecht
Tübingen, Januar 2000

© Heiner Lamprecht, 1999

Kontakt:

Philosophenweg 79
72076 Tübingen
Tel: 07071 / 600 016
Fax: 07071 / 600 164
email: heiner@kijumfo.de

Dieses Buch gilt als Freie Software im Sinne der GNU General Public License (GPL). Das Buch ist als PostScript-Datei verfügbar unter:
http://agenda21.ggi.uni-tuebingen.de/heiner/latex/

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Der Einsatz von Computern in der Verarbeitung von Texten ist heutzutage zur Selbstverständlichkeit geworden. Doch obwohl es eine Vielzahl von unterschiedlichsten Textverarbeitungsprogrammen auf dem Markt gibt, hat dies die Qualität, in der die Texte ausgedruckt werden, gegenüber den Möglichkeiten einer Schreibmaschine nur unwesentlich verändert. Einer der Gründe dafür ist, daß die Programme in der Regel den Autor eines Textes nicht genügend unterstützen.

Das Setzen von Texten in einer Qualität, die ein professioneller Schriftsetzer in einem Verlag erreicht, kann man jedoch auch auf einem kleinen Computer mit einfachen Mittel erzielen. Mit Hilfe von TEX und LATEX ist es möglich, sich das Wissen und die Erfahrung aus über fünfhundert Jahren Buchdruck zunutze zu machen. Die Arbeitsweise unterscheidet sich dabei jedoch grundlegend von der anderer Systeme.

1.1 Die Entstehung von TEX

Im Februar 1977 begann Donald E. Knuth, ein Textverarbeitungssystem zu entwickeln, das heute als TEX (sprich „Tech", kann auch „TeX" geschrieben werden) bekannt geworden ist. Es dient zum Setzen und Drucken von Texten und hat mittlerweile eine sehr große Verbreitung besonders an wissenschaftlichen und technischen Instituten und auch bei Verlagen erreicht. Es ist auf allen Rechnerplattformen erhältlich, vom kleinen Home-PC über Workstations bis hin zu Großrechenanlagen wie der CRAY. Dabei ist die Bedienung von TEX überall gleich. Diese Tatsache verbunden mit dem Umstand, daß es kostenlos zur Verfügung steht, hat TEX praktisch zum Standard in vielen Bereichen der Wissenschaft werden lassen. Knuth selbst schreibt in seinem Buch „The TEXbook" (Knuth, 1986): *„TEX is an new typesetting system intended for* *the creation of beautiful books – and especially for books that contain a lot of mathematics. By preparing a manuscript in TEX format, you will be telling the computer exactly how the manuscript is to be transformed into pages whose typographic quality is comparable to that of the world's finest printers.“*

LaTeX (sprich „Lah-tech“, kann auch „LaTeX“ geschrieben werden) ist ein sogenanntes Makro-Paket von Leslie Lamport (Lamport, 1994). Es vereinfacht die Nutzung von TeX erheblich und ermöglicht es dem Autor so, sein Schriftstück in einfacher Weise in Buchdruck-Qualität zu setzen und auszudrucken.

LaTeX 2_ε (sprich „Lah-tech zwei e“) ist die aktuelle Version von LaTeX. Es wird nicht mehr von Leslie Lamport betreut, sondern von einer größeren Gruppe von Entwicklern (LaTeX3 Project Team, 1995b). Diese Version wird in dem vorliegenden Handbuch beschrieben. Sofern nicht anders angegeben, wird im Folgenden unter „LaTeX“ immer „LaTeX 2_ε“ verstanden.

1.2 Die Idee von TeX und LaTeX

1.2.1 Autor, Designer und Setzer

Für eine Publikation übergibt der Autor dem Verleger normalerweise ein maschinengeschriebenes Manuskript. Der Buch-Designer des Verlages entscheidet dann über das Layout des Schriftstücks (z. B. Länge einer Zeile, Schriftart, Abstände vor und nach Kapiteln usw.) und schreibt dem Setzer die dafür notwendigen Steuerdaten hinzu.

Im Gegensatz zu einem menschlichen Buchdesigner, der die Absicht und inhaltliche Struktur eines Textes aufgrund seines Fachwissens erkennen kann, muß LaTeX vom Autor gesagt bekommen, was zum Beispiel eine Überschrift oder eine Fußnote sein soll. Diese Informationen werden innerhalb des Textes durch Befehle an LaTeX übergeben. Anschließend wird LaTeX aufgerufen, das nun diese Befehle abarbeitet und den Text formatiert.

Damit unterscheidet sich die Arbeitsweise grundlegend von anderen Programmen wie zum Beispiel Star-Writer. Bei diesen Systemen bestimmt der Autor während der Texteingabe interaktiv das Aussehen des Textes am Monitor. Er sieht also bereits beim Schreiben, wie der Text nach dem Ausdrucken aussieht. Daher werden diese Programme auch als WYSIWYG-Software („**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et“) bezeichnet.

Bei LaTeX sieht der Autor beim Schreiben der Eingabedatei noch nicht, wie der Text nach dem Formatieren aussehen wird. Er kann aber durch Aufruf des entsprechenden Programms jederzeit das endgültige Aussehen auf dem Monitor oder einem Ausdruck anschauen und danach seine Eingabedatei entsprechend korrigieren und die Arbeit fortsetzen.

Diese Arbeitsweise von LATEX sieht auf den ersten Blick etwas kompliziert aus. LATEX ist dadurch aber in der Lage, erheblich genauer zu arbeiten. Während der Autor bei anderen Systemen bei der Positionierung des Textes auf sein eigenes Augenmaß angewiesen ist, arbeitet LATEX mit einer Genauigkeit in der Größenordnung der Wellenlänge des Lichtes!

1.2.2 Layout-Design

Typographisches Design ist eine Kunst und ein Handwerk zugleich. Ungeübte Autoren machen oft gravierende Formatierungsfehler. Fälschlicherweise glauben viele Laien, daß Buchdruck-Design vor allem eine Frage der Ästhetik ist – wenn das Schriftstück vom künstlerischen Standpunkt aus „schön“ aussieht, dann ist es schon gut „designed“. Schriftstücke werden jedoch gelesen und nicht in einem Museum aufgehängt. Die leichtere Lesbarkeit und bessere Verständlichkeit ist also wichtiger als das schöne Aussehen. Professionelle Schriftsetzer greifen daher auf eine Erfahrung von mehr als fünfhundert Jahren Buchdruck zurück.

Beispiele: Die Schriftgröße und Nummerierung von Überschriften soll so gewählt werden, daß die Struktur der Kapitel und Unterkapitel klar erkennbar ist. Die Zeilenlänge soll so gewählt werden, daß anstrengende Augenbewegungen des Lesers vermieden werden, nicht so, daß sie das Papier möglichst schön ausfüllt.

Bei LATEX ist es gelungen, diese Erfahrung auf den Computer zu übertragen. Mit WYSIWYG-Systemen erzeugen selbst geübte Autoren im allgemeinen zwar ästhetisch schöne, aber schlecht strukturierte Schriftstücke. LATEX verhindert solche Formatierungsfehler, indem es den Autor dazu zwingt, die logische Struktur des Textes anzugeben, und dann automatisch das dafür am besten geeignete Layout verwendet. Der Autor sollte – wenn möglich – immer die von LATEX vorgeschlagene Formatierung übernehmen. Es kann zwar selbstverständlich immermal wieder nötig sein, in diese Formatierung von Hand einzugreifen. Damit sollte man jedoch möglichst sparsam umgehen, um das ausgewogene Erscheinungsbild der verschiedenen Elemente eines Dokumentes nicht zu gefährden.

Kapitel 2

Arbeitsweise von LaTeX

2.1 Die Eingabedatei

Möchte man mit LaTeX einen Text setzen, so muß man diesen zuerst zusammen mit den entsprechenden Befehlen mit einem Editor erstellen. Manche Editoren, wie zum Beispiel der Emacs, kennen bestimmte Modi, die die Eingabe erleichtern. Eine sehr mächtige Unterstützung bietet das auctex-Paket, das als Zusatz für den Emacs erhältlich ist. Dieses Paket unterstützt nicht nur die vereinfachte Eingabe diverser Befehle incl. der Abfrage nach den jeweiligen Parametern, es unterstützt auch die Verwaltung von Marken zur Referenzierung innerhalb des Textes. Schließlich bietet das Paket komfortable Werkzeuge zur Aufteilung eines Dokumentes in mehrere Dateien.

Für LaTeX ist es für das Aussehen des formatierten Textes unerheblich, wie der Text im Editor aussieht. Ist der Text erstellt, so bekommt er üblicherweise die Dateiendung `.tex`. Das Formatieren wird dann mit dem Befehl `latex` gestartet. Beim Aufruf von LaTeX kann man auch direkt den Dateinamen angeben. LaTeX erzeugt eine Datei mit der Endung `.dvi`. Die Endung steht für *device independent*. Die Datei ist also von der Hardware unabhängig. Sie kann mit entsprechenden Programmen auf dem Monitor angezeigt, oder mit einem Drucker ausgedruckt werden. Dabei spielt die Auflösung des Druckers keine Rolle. Die verwendeten Schriften werden immer in der benötigten Größe benutzt oder erzeugt.

LaTeX schreibt während des Formatierens verschiedene Informationen über Überschriften, Zitate, Verweise o. ä. in Hilfsdateien. Diese Informationen werden beim nächsten Durchlauf benutzt. Daher kann es notwendig werden, vor dem Ausdrucken des Dokumentes LaTeX mehrfach hintereinander aufzurufen, um alle Referenzen aufzulösen.

Der Ablauf des Formatierens ist in Abb. 2.1 dargestellt. LaTeX 2_ε liest beim

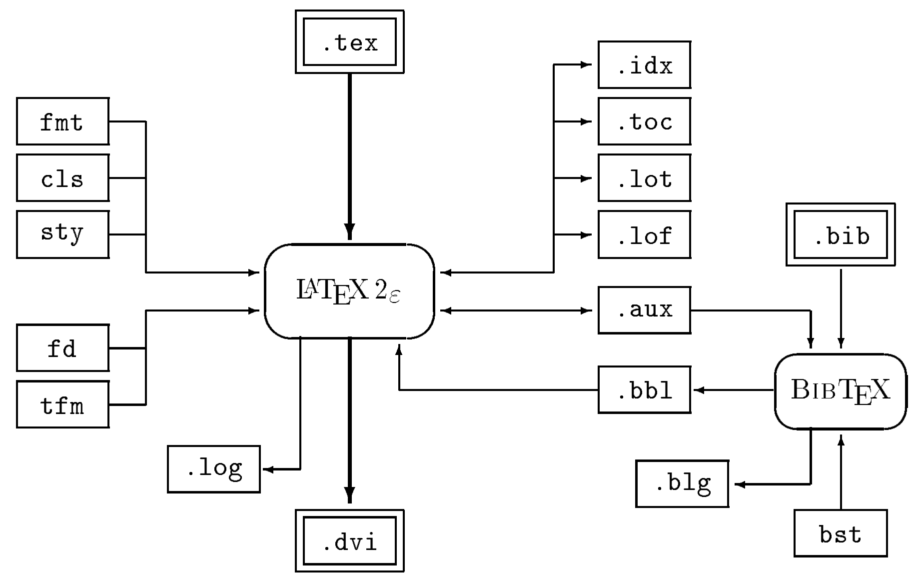


Abbildung 2.1: Schematischer Ablauf eines LATEX 2ε-Laufes

Aufruf als erstes eine Formatdatei (`fmt`). Danach folgt die erste Zeile der Eingabedatei (`.tex`), in der die verwendete Klasse (`cls`) festgelegt ist. Es folgen eventuelle Zusatzpakete (`sty`), die in der Präambel der Eingabedatei angegeben sind. Anschließend wird die Hilfsdatei (`.aux`) eingelesen, in der Informationen über Verweise, Zitate, ... stehen. Erst dann beginnt LATEX 2ε damit, den Textteil der Eingabedatei zu lesen und zu setzen.

Während des Formatierens liest und schreibt LATEX 2ε eventuelle Dateien für Inhalts- , Abbildungs- oder Tabellenverzeichnisse sowie den Index (`.toc`, `.lof`, `.lot`, `.idx`). Sämtliche Aktionen protokolliert LATEX 2ε in einer weiteren Datei (`.log`), bevor schließlich die Ausgabe erzeugt wird (`.dvi`). Gleichzeitig wird auch die Hilfsdatei (`.aux`) neu erstellt.

Die Abbildung zeigt darüber hinaus das Zusammenspiel von LATEX 2ε und BIBTEX. Dieses Programm dient zur Erstellung umfangreicher Literaturverzeichnisse. Wird nach einem LATEX 2ε-Lauf BIBTEX aufgerufen, so liest dieses Programm zunächst die Hilfsdatei ein. Anschließend wird die darin genannte Datenbank (`.bib`) und eine Stildatei (`bst`) eingelesen. Die Angaben für das Literaturverzeichnis werden dann in die Bibliographiedatei (`.bbl`) geschrieben. Bei einem erneuten Aufruf übernimmt LATEX diese Angaben und setzt das Literaturverzeichnis.

Heute ist der 9. Januar 2000. Oder:	`Heute ist der \today.`
Heute ist der 9. Januar 2000.	`Oder: Heute ist der \today .`
Falsch ist:	`Falsch ist:`
Am 9. Januar 2000regnet es.	`Am \today regnet es.`
Richtig:	`Richtig:`
Am 9. Januar 2000 scheint die Sonne. Oder:	`Am \today{} scheint die Sonne.`
Am 9. Januar 2000 schneit es.	`Oder: Am \today\ schneit es.`

Abbildung 2.2: Verwendung von Leerstellen nach Befehlen

2.1.1 LaTeX-Befehle

Befehle dienen der Steuerung der Formatierung durch LaTeX. Nahezu sämtliche Befehle werden durch einen Backslash (\) eingeleitet. Sie werden grundsätzlich entweder von einem oder mehreren Buchstaben oder einem Sonderzeichen oder einer Ziffer gefolgt. Besteht der Befehl aus einem oder mehreren Buchstaben, so wird er durch ein Sonderzeichen, eine Ziffer oder ein Leerzeichen beendet. Groß- und Kleinbuchstaben haben auch in Befehlsnamen *verschiedene* Bedeutung. Die Leerstelle nach Eingabe eines Befehl wird *nicht* gesetzt. Will man dennoch eine Leerstelle erreichen, so muß man `{}` oder `\␣` (Das ␣ steht für die Eingabe eines Leerzeichens.) eingeben.

„Unsichtbare“ Zeichen wie das Leerzeichen, Tabulatoren und das Zeilenende werden von LaTeX einheitlich als Leerzeichen behandelt. *Mehrere* Leerzeichen werden wie *ein* Leerzeichen behandelt. Eine Leerzeile zwischen Textzeilen bedeutet das Ende eines Absatzes. *Mehrere* Leerzeilen werden wie *eine* Leerzeile angesehen.

Möchte man andere Abstände zwischen Worten oder Zeilen haben, kann man das also nicht durch Eingabe von mehreren Leerzeichen oder -zeilen erreichen. Hierzu stehen verschiedene LaTeX-Befehle zur Verfügung (siehe hierzu Punkt 3.1.5 auf Seite 19).

Parameter

Vielen Befehlen muß man Parameter übergeben. Hierbei wird unterschieden zwischen den zwingend notwendigen Parametern, die in geschweiften Klammern (`{...}`) gesetzt werden, und Parametern, die weggelassen werden können (optionale Parameter). Diese werden in eckigen Klammern gesetzt (`[...]`).

Die geschwungenen Klammern können außerdem verwendet werden, Teile des Textes zu Gruppen zusammenzufassen. Die Wirkung von Befehlen, die innerhalb von Gruppen oder Umgebungen angegeben werden, endet immer mit dem Ende der Gruppe bzw. der Umgebung. In Abb. 2.2 ist `{}` eine leere Gruppe, die außer der Beendigung des Befehlsnamens `today` keine Wirkung hat.

2.1.2 Steuerzeichen

Verschiedene Zeichen haben bei LaTeX bestimmte Steuerfunktionen. Sie dürfen daher im normalen Text nicht direkt verwendet werden. Diese sind:

`$ & % # _ { }  ~  ^ "  \  | < >`

Die folgenden sieben Zeichen können jedoch durch das Voranstellen des Zeichens \ (Backslash) gedruckt werden:

$ & % # _ { }	`\$ \& \% \# \_ \{ \}`

Für die restlichen Symbole stehen gesonderte Befehle zur Verfügung. Dies gilt auch für viele andere Sonderzeichen (siehe Anhang B).

2.1.3 Kommentare

Der Text, der hinter einem Prozentzeichen (`%`) steht, wird bis zum Ende der Eingabezeile, von LaTeX ignoriert. Dies kann für Notizen des Autors oder Teile des Textes verwendet werden, die nicht oder noch nicht ausgedruckt werden sollen.

Das Auskommentieren bietet sich darüberhinaus auch zur Suche von Fehlern an, wenn diese in komplizierteren Strukturen zu finden sind.

2.1.4 Notation

In diesem Buch wird die Befehlssyntax in einer „monospaced“ Schrift in einem Rahmen angegeben. Die Parameter werden dabei zusätzlich in einer „italic“ Schriftart geschrieben.

```
\befehlsname[optionaler Parameter] {Parameter}
```

2.2 Struktur

Eine Eingabedatei besteht grundsätzlich aus zwei Teilen: der Präambel und dem Textteil. In der Präambel können Definitionen erfolgen, die für das gesamte Dokument gelten sollen. Hier können z. B. einzelne Maße oder Zähler verändert oder erzeugt sowie zusätzliche Pakete geladen werden. Fehlen diese

Angaben, so benutzt LaTeX bestimmte Grundeinstellungen, die in den verschiedenen Klassen- und Paketdateien festgelegt sind. In der Präambel darf noch kein Text gesetzt werden. Die Präambel muß mindestens aus dem Befehl

```
\documentclass
```

bestehen (siehe Abschnitt 2.3). Dies ist normalerweise die erste Zeile. Vor dieser ersten Zeile darf außer Kommentaren höchstens die Zeil

```
\NeedsTeXFormat{LaTeX2e}[Version]
```

stehen[1]. Mit dem optionalen Parameter ***Version*** kann darüberhinaus noch die Version (z.B. ***1995/12/01***) angegeben werden, ab der das Paket verwendet werden kann. Mit dem Befehl

```
\begin{document}
```

endet die Präambel und der eigentliche Textteil beginnt. Alles, was gedruckt werden soll, muß zwischen dieser Zeile und dem Befehl

```
\end{document}
```

stehen. Eventuelle Zeilen, die nach dieser noch folgen, sind zwar erlaubt, werden von LaTeX jedoch vollständig ignoriert. Abbildung 2.3 zeigt den *schematischen* Aufbau einer LaTeX 2_ε-Datei.

2.3 Dokumentenklassen

Wie in Abbildung 2.3 erkennbar, muß am Anfang der Datei das grundsätzliche Layout des Dokumentes mit

```
\documentclass[Optionen]{Klasse}
```

definiert werden. Die Klasse eines Dokumentes enthält Vereinbarungen über logische Strukturen, z. B. die Gliederungsebenen (Kapitel etc.), Listen und Umgebungen, die für alle Dokumente einer Klasse gelten.

[1] Da es beim Versenden von LaTeX-Dokumenten an andere Personen manchmal nötig ist, lokale oder weniger verbreitete Paketdateien bzw. eigene Änderungen mit der Quelldatei zusammen zu versenden, verfügt LaTeX 2_ε über die Umgebung `filecontents`. Sie hat als Argument einen Dateinamen und der Inhalt der Umgebung besteht aus dem Inhalt der Datei. Bei der Bearbeitung überprüft LaTeX 2_ε, ob die Datei existiert. Ist dies der Fall, so wird der Inhalt der Umgebung übersprungen. Fehlt die Datei jedoch, so schreibt LaTeX 2_ε den Inhalt der Umgebung in eine Datei mit dem angegebenen Namen. Beide Aktionen werden von LaTeX 2_ε gemeldet. Die Umgebung `filecontents` darf nur *vor* dem Befehl `\documentclass` verwendet werden.

```
\documentclass{article}
  Präambel
\begin{document}
  Textteil
\end{document}
```

Abbildung 2.3: Schema einer LaTeX 2_ε-Datei[2]

Zwischen den geschwungenen Klammern *muß* eine der in Tabelle 2.1 angeführten Dokumentenklassen angegeben werden. Selbstverständlich können auch eigene Klassen definiert werden.

Zwischen den eckigen Klammern *können*, durch Kommata getrennt, eine oder mehrere Optionen für Varianten der Standard-Layouts angegeben werden. Die wichtigsten Optionen sind in der Tabelle 2.2 angeführt.

Tabelle 2.1: Die wichtigsten Dokumentenklassen

`article`	für Artikel in wissenschaftlichen Zeitschriften, Vorträge, Praktikumsarbeiten, Seminararbeiten, kürzere Berichte, Anträge, Gutachten, Programmbeschreibungen, Einladungen u. v. a.
`report`	für längere Berichte, die aus mehreren Kapiteln bestehen, Diplomarbeiten, Dissertationen, Skripten u. ä.
`book`	für Bücher
`letter`	für Briefe
`slides`	für das Erstellen von Folien
`ltxdoc` `ltxguide` `ltxnews`	dient dem Erstellen der LaTeX 2_ε-Dokumentation

[2] Strenggenommen gehört der Befehl `\documentclass` bereits zur Präambel.

Tabelle 2.2: Ausgewählte Optionen für Dokumentenklassen

`10pt` für 10 Punkt hohe Schrift. Dies ist die Voreinstellung.

`11pt` für 11 Punkt hohe Schrift, wie in diesem Handbuch.

`12pt` für 12 Punkt hohe Schrift, das ist etwa die bei Schreibmaschinen übliche Größe.

Wenn keine dieser Optionen angegeben wird, verwendet LaTeX 10 Punkte hohe Schrift, was auch eine beim Buchdruck übliche Größe ist.

`a4paper` für Papier im DIN A4-Format.

`titlepage` für eine eigene Titelseite (nur bei der Dokumentenklasse `article`)

`twocolumn` für zweispaltigen Druck

`twoside` für Ausgabe mit unterschiedlichen rechten und linken Seiten, Voreinstellung bei `book`.

2.4 Zusatz-Pakete

Es gibt für LaTeX 2_ε eine fast unüberschaubare Menge an Paketen, die zusätzliche Definitionen enthalten oder spezielle Funktionen bereitstellen. Man kann außerdem recht einfach selbst Pakete schreiben, um z. B. einen eigenen Briefkopf zu erstellen (siehe Kapitel 6 auf Seite 87). Diese Pakete werden mit dem Befehl

```
\usepackage[Optionen]{Pakete}
```

geladen. Dabei können mehrere Pakete mit *einem* Befehl geladen werden. Allerdings müssen eventuelle Optionen dann auch von *allen* dieser Pakete unterstützt werden. Alternativ kann man die Optionen auch in der Zeile mit `\documentclass` übergeben. Diese Optionen werden dann nur an die Pakete weitergeleitet, die sie auch unterstützen. Eine Auswahl von Paketen findet sich in der Tabelle 2.3. Um zum Beispiel einen zweispaltigen Artikel mit eingebundenen PostScript-Bildern zu erstellen, lauten die beiden ersten Zeilen:

```
\documentclass[twocolumn]{article}
\usepackage{epsfig}
```

Möchte man zusätzlich umfangreichere Verweise verwenden, so benötigt man außerdem das Paket `varioref` (siehe Abschnitt 4.4.1 auf Seite 54):

```
\documentclass[twocolumn]{article}
\usepackage{epsfig}
\usepackage[german]{varioref}
```

Alternativ kann man auch schreiben:

```
\documentclass[twocolumn,german]{article}
\usepackage{epsfig,varioref}
```

Tabelle 2.3: Pakete (eine Auswahl)

`array` Erweiterte Versionen der Umgebungen `array`, `tabular` und `tabular*`.

`floatfig` Ermöglicht das einfache Herumfließen von Text um Bilder

`ftnright` Plaziert bei zweispaltigem Layout Fußnoten am unteren Ende der rechten Spalte.

`latexsym` Stellt eine Reihe alter LaTeX-Symbole wie $\Box$ zur Verfügung.

`longtable` Bietet Tabellen über mehrere Seiten, die automatisch umgebrochen werden.

`multicol` ermöglicht mehrspaltigen Satz mit Kolumnenausgleich.

`showkeys` Druckt die Namen aller `\label`-, `\ref`- und `\pageref`-Befehle im Text aus.

`seminar` Ein umfangreiches Paket, das für das Setzen von Overhead-Folien benutzt wird.

Kapitel 3

Setzen von Text

In diesem Kapitel sollen grundlegende Dinge des Textsatzes behandelt werden. Dabei wird es um die Gestaltung des Seiten-Layouts ebenso gehen, wie um Schriftauswahl oder das Erzeugen von speziellen Zeichen.

3.1 Seitenlayout

3.1.1 Satzspiegel

Unter dem Begriff „Satzspiegel“ versteht man die Fläche auf einer Seite, die mit Text und Abbildungen gefüllt ist. Um diese Fläche herum gruppiert sich der Freiraum, der in *Kopf*, *Bund* (Leerraum in der Mitte einer Doppelseite), *(Außen-) Seite* und *Fuß* eingeteilt wird. An den Rändern kann manchmal noch zusätzlicher Text in Form sog. *Marginalien* stehen.

Größe, Form und Anordnung der einzelnen Felder ergeben zusammen das *Seitenlayout*. Dieses kann auf mehreren Seiten durchaus unterschiedlich sein. So unterscheidet sich z. B. das Layout der ersten Seite eines Kapitel von den übrigen Textseiten. Möchte man außerdem den Text für zweiseitigen Druck formatieren, so unterscheidet LaTeX zwischen linken und rechten Seiten. Bei einseitigem Druck sind die Ränder auf allen Seiten jeweils gleich, bei zweiseitigem Druck ist dagegen darauf zu achten, daß sich der Textbereich der Seiten deckt. Hier muß also zwischen *innerem* und *äußerem* Rand unterschieden werden. Bei einseitigem Druck dagegen gilt als innerer Rand immer der linke.

Die Parameter, mit denen der Satzspiegel bei LaTeX 2_ε beschrieben wird, sind in Tabelle 3.1 beschrieben. Diese Maße werden normalerweise in den geladenen Klassendateien bzw. in entsprechenden Zusatzpaketen definiert. Man kann sie jedoch in der Präambel des Dokumentes individuell ändern (siehe Abschnitt 3.1.7). Änderungen innerhalb eines Dokumentes sind in der Regel nicht sinnvoll, da sie das Erscheinungsbild des Textes empfindlich stören können.

Tabelle 3.1: Parameter für den Satzspiegel

`\textheight`	Höhe des Textkörpers ohne Kopf und Fuß
`\textwidth`	Breite des Textkörpers
`\columnsep`	Spaltenabstand bei Mehrspaltensatz
`\columseprule`	Breite der Trennlinie zwischen den Spalten (Voreinstellung: 0pt)
`\columnwidth`	Breite der Spalten, wird von LaTeX automatisch berechnet
`\linewidth`	Länge der aktuellen Zeile. Entspricht normalerweise `\columnwidth`, kann sich jedoch in anderen Umgebungen ändern
`\evensidemargin`	Bei zweiseitigem Druck linker Rand auf geraden Seiten
`\oddsidemargin`	Bei zweiseitigem Druck linker Rand von ungeraden Seiten, sonst linker Rand auf allen Seiten
`\footskip`	Abstand zwischen der Grundlinie der letzten Zeile und der Grundlinie der Fußzeile
`\headheight`	Höhe der Kopfzeile
`\headsep`	Abstand zwischen Kopfzeile und Textkörper
`\topmargin`	Raum oberhalb der Kopfzeile
`\marginparpush`	Mindestabstand zweier aufeinanderfolgender Randnotizen
`\marginparsep`	horizontaler Abstand zwischen Randnotizen und Textkörper
`\marginparwidth`	Breite der Randnotizen
`\paperheight`	Papierhöhe
`\paperwidth`	Papierbreite

Mit Hilfe des Zusatzpaketes `layout` kann man sich den jeweils gültigen Satzspiegel anzeigen lassen. Das Paket erzeugt ein verkleinertes Bild der Seite, das alle wichtigen Informationen über den Satzspiegel enthält.[1] Der Satzspiegel des vorliegenden Buches ist im Anhang D auf den Seiten 252 und 253 zu sehen.

3.1.2 Seitenformat

Das genaue Aussehen von Kopf- und Fußzeile wird mit dem *Seitenformat* festgelegt. Hierfür stehen bei LaTeX mehrere verschiedene Möglichkeiten zur Verfügung, die in Tabelle 3.2 aufgezählt sind. Mit dem Befehl

$\boxed{\texttt{\textbackslash pagestyle\{}\textit{Stil}\texttt{\}}}$

[1] Es ist möglich, daß das Paket zu mehreren Fehlermeldungen führt (`Bad \line or \vector argument`). Diese Meldungen können jedoch getrost übergangen werden.

kann man das Seitenformat im Vorspann ändern. Wird der Befehl `\pagestyle` weggelassen, so wird automatisch `plain` als Seitenstil genommen, sofern nicht in geladenen Paketen etwas anderes festgelegt ist.

Möchte man das Aussehen einer einzelnen Seite verändern, so kann man das mit dem Befehl

```
\thispagestyle{Stil}
```

machen. Seine Wirkung bezieht sich nur auf die laufende Seite, die nachfolgenden bekommen wieder den vorher geltenden Seitenstil.

Ändern der Kopfzeile

Bei den beiden Stilen `headings` und `myheadings` kann man mit den Befehlen

```
\markright{rechter Kopf}
\markboth{linker Kopf}{rechter Kopf}
```

den Seitenkopf verändern. Die Wirkung entfalten die Befehle erst ab der zweiten Seite des Dokumentes. Der Befehl `\markboth` wird bei zweiseitigem Druck verwendet, um unterschiedliche Kopfzeilen zu erreichen. Bei einseitigem Druck gilt jede Seite als rechte Seite.

Tabelle 3.2: Parameter für den Seitenstil

`plain`	Der Seitenkopf ist leer, die Fußzeile besteht aus der zentrierten Seitennummer.
`empty`	Kopf- und Fußzeile sind leer. Es wird keine Seitennummer ausgegeben.
`headings`	Die Kopfzeile enthält die Seitennummer sowie Informationen über die Überschrift. Diese sind von der gewählten Dokumentenklasse abhängig. In der Regel wird die aktuelle Kapitelüberschrift ausgegeben. Bei `book` oder der Option `twoside` wird auf den linken Seiten das aktuelle Kapitel und auf den rechten die aktuelle Überschrift ausgegeben. Bei Anfangsseiten von Kapiteln bleibt der Kopf leer und die Fußzeile enthält die Seitennummer.
`myheadings`	Wie bei `headings`, allerdings wird die Kopfzeile nicht automatisch bestimmt, sondern *muß* mit den Befehlen `\markright` und `\markboth` festgelegt werden.

Stehen auf einer Seite mehrere `\section` oder `\subsection` Befehle, so werden bei linken Seiten jeweils die letzten, bei rechten jeweils die ersten Befehle der Seite für den Text der Kopfzeile benutzt.

Seitennumerierung

Die Numerierung der Seiten erfolgt automatisch, unabhängig davon, ob eine Seitenzahl ausgegeben wird oder nicht. Der Ausgabestil der Numerierung wird mit

```
\pagenumbering{Stil}
```

bestimmt. Zur Auswahl stehen fünf verschiedene Möglichkeiten: `arabic` erzeugt eine Numerierung mit fortlaufenden arabischen Ziffern, `roman` benutzt kleine römische Zahlen, `Roman` große römische Zahlen, `alph` Kleinbuchstaben (a–z) und `Alph` benutzt Großbuchstaben (A–Z). Der Standardwert ist `arabic`. Jede Änderung des Stils mit `\pagenumbering` setzt die Seitenzahl wieder auf 1 zurück. Möchte man den Zähler von Hand ändern, so dient dazu der Befehl

```
\setcounter{page}{neue Seitenzahl}
```

3.1.3 Seitenformat anpassen

Außer den oben genannten Möglichkeiten, das Aussehen der Seite zu beeinflussen, steht das Paket `fancyheadings` zur Verfügung (Oostrum, 1995). Mit Hilfe dieses Pakets kann man auf recht einfache Art und Weise Kopf- und Fußzeile definieren. Die Kopf- und Fußzeilen können dabei nicht nur länger als `\textwidth` sein, sondern auch mehrere Zeilen umfassen.

Die Definition erfolgt über insgesamt sechs Befehle, die das Aussehen jeweils der linken und rechten Seite und der Mitte der Kopf- und Fußzeilen steuern. Außerdem wird zwischen einer linken und einer rechten Textseite unterschieden.

Die Befehle zur Steuerung der Kopfzeile lauten:

```
\lhead[LK-gerade]{LK-ungerade}
\chead[MK-gerade]{MK-ungerade}
\rhead[RK-gerade]{RK-ungerade}
```

Dabei legt `\lhead` das Aussehen des linken Teiles der Kopfseite fest, *`LK-gerade`* enthält optional die Angaben für gerade Textseiten, *`LK-ungerade`* die für die ungeraden Seiten. Die Befehle für die Fußzeilen lauten entsprechend:

```
\lfoot[LF-gerade]{LF-ungerade}
\cfoot[MF-gerade]{MF-ungerade}
\rfoot[RF-gerade]{RF-ungerade}
```

Zusätzlich zu diesen Kopfdefinitionen kann das `fancyheadings`-Paket noch horizontale Linien als Begrenzung der Kopf- und Fußzeile ziehen. Die Dicke der Linien wird mit den beiden Befehlen

```
\headrulewidth{Breite}
\footrulewidth{Breite}
```

gesteuert. Die Voreinstellung für `\headrulewidth` beträgt 0.4pt, für `\footrulewidth` 0pt.

Durch eine Neudefinition der beiden Befehle können auch erheblich komplexere Seitenlayouts verwirklichen.

3.1.4 Blocksatz

Normalerweise wird Text in Büchern in Blocksatz, d. h. mit Randausgleich an beiden Seiten gesetzt. LaTeX führt den Zeilen- und Seitenumbruch dabei automatisch durch. Hierzu wird für jeden Absatz die bestmögliche Aufteilung der Wörter auf die Zeilen bestimmt, und – wenn notwendig – werden Wörter automatisch getrennt. Bei der Option `twoside` wird dieser Ausgleich sogar auf beiden Seiten gleichmäßig vorgenommen, um auf einer Doppelseite ein möglichst ausgewogenes Erscheinungsbild zu erreichen. Andere Systeme (WYSIWYG) sind nur in der Lage, die Wortabstände innerhalb einer Zeile gleichmäßig zu gestalten.

Das Ende von Wörtern und Sätzen wird durch Leerzeichen gekennzeichnet. Hierbei spielt es keine Rolle, ob man ein oder 100 Leerzeichen eingibt.
Eine oder mehrere Leerzeilen kennzeichnen das Ende von Absätzen.

```
Das Ende von Wörtern und Sätzen
wird durch Leerzeichen gekennzeichnet.
Hierbei spielt es keine Rolle, ob man
ein      oder                    100
Leerzeichen eingibt.

Eine oder mehrere Leerzeilen
kennzeichnen das Ende von
Absätzen.
```

Das genaue Aussehen der Absätze hängt von der benutzten Dokumentenklasse ab. In der Regel werden Absätze bei Artikeln, Berichten und Büchern durch das Einrücken der ersten Zeile gekennzeichnet. In Briefen dagegen werden Absätze wie in diesem Abschnitt meistens durch einen vertikalen Abstand ohne horizontalen Einzug dargestellt. Das Einrücken eines Absatzes kann mit dem Befehl

---

```
\noindent
```

unterbunden werden. Dem Befehl darf *keine* Leerzeile folgen.

Es kann jedoch auch manchmal nötig sein, den Zeilen- oder Seitenumbruch manuell zu beeinflussen. Hierzu stehen verschiedene Befehle zur Verfügung: Die Befehle `\\` und `\newline` bewirken einen Zeilenwechsel ohne neuen Absatz, der Befehl `\\*` einen Zeilenwechsel, bei dem außerdem kein Seitenwechsel erfolgen darf. Der Befehl

```
\newpage
```

erzwingt einen Seitenwechsel. Mit den vier Befehlen

```
\linebreak[n]
\nolinebreak[n]
\pagebreak[n]
\nopagebreak[n]
```

kann man festlegen, ob an einer bestimmten Stelle ein Zeilen- bzw. Seitenwechsel eher günstig oder eher ungünstig ist. Der Parameter `[n]` gibt dabei die Dringlichkeit an. Der Wert muß zwischen `0` und `4` liegen. Eine `4` kann auch weggelassen werden und ist gleichbedeutend mit „zwingend".

Darüberhinaus gibt es den Befehl

```
\enlargethispage{Länge}
```

Damit läßt sich die Texthöhe einer gegebenen Seite um ***Länge*** vergrößern oder verkleinern.[2] Auf diese Weise kann man z. B. eine zusätzliche Zeile auf einer Seite unterbringen.

LaTeX bemüht sich, den Zeilenumbruch besonders schön zu machen. Wenn es keine den strengen Regeln entsprechende Möglichkeit für einen glatten rechten Rand findet, läßt es eine Zeile zu lang und gibt eine entsprechende Warnung aus ("overfull hbox"). Dies tritt besonders dann auf, wenn LaTeX keine geeignete Stelle für die Silbentrennung findet. Innerhalb der `sloppypar`-Umgebung ist LaTeX etwas weniger streng in seinen Ansprüchen und vermeidet solche überlangen Zeilen, indem die Wortabstände stärker – notfalls auch unschön – vergrößert werden. In diesem Fall werden zwar Warnungen gemeldet ("underfull hbox"), das Ergebnis ist aber meistens brauchbar. Dieses Vorgehen bietet sich vor allem bei schmalen Textteilen an.

[2] ***Länge*** besteht aus einer Zahl und einer der in Tabelle 3.5 angegebenen Längeneinheiten.

Tabelle 3.3: Horizontale Abstände

`\,`	ein sehr kleiner Abstand („Länge eines Kommas“) wie z. B. in Abkürzungen
`\␣`	eine Leerstelle,
`~`	eine Leerstelle, an der kein Zeilenumbruch erfolgen darf
`\quad`	so breit, wie ein Buchstabe hoch ist („weißes Quadrat“)
`\qquad`	doppelt so breit wie ein `\quad`
`\hfill`	ein Abstand, der sich von 0 bis ∞ ausdehnen kann.
`\hspace{`*Länge*`}`	Abstand der Größe *Länge*.
`\hspace*{`*Länge*`}`	Abstand der Größe *Länge*, wird auch am Zeilenanfang oder -ende gesetzt.

3.1.5 Wortabstände

LaTeX muß, um einen glatten rechten Textrand zu erreichen, die Wortabstände ein wenig variieren. In manchen Fällen soll der Abstand jedoch nicht zu groß werden. Dies kann mit den Befehlen in Tabelle 3.3 erreicht werden. Exakt festgelegte Abstände werden mit den Befehlen

```
\hspace{Länge}
\hspace*{Länge}
```

erzeugt. Wenn der Abstand auch am Beginn oder am Ende einer Zeile erhalten bleiben soll, wird die zweite Version benutzt. Die Längenangabe besteht im einfachsten Fall aus einer Zahl und einer Einheit. Die wichtigsten Einheiten sind in Tabelle 3.5 auf Seite 22 angeführt.

Der Befehl `\hfill` kann dazu genutzt werden, den Rest einer Zeile auszufüllen. Man kann den Leerraum auch mit einem Strich oder mit Punkten ausfüllen. Die Befehle dazu lauten `\hrulefill` und `\dotfill`.

Hier ist 1.5 cm Abstand.	`Hier\hspace{1.5cm}ist 1.5 cm Abstand.`
links rechts	`links\hfill rechts \\`
„ ‚Parsifal‘ dauert länger als ‚Cats‘.“	`"‘\,,Parsifal‘ dauert länger als ,Cats‘."’`
Punkte	`Punkte \dotfill \\`

3.1.6 Zeilenabstand

Der Zeilenabstand ist normalerweise vorgegeben. Um in einem Schriftstück größere Zeilenabstände, als in der Dokumentenklasse vorgesehen sind, zu verwenden, gibt es in LaTeX den Befehl

```
\linespread{Faktor}
```

für „eineinhalbzeilige“ Ausgabe:

```
\linespread{1.3}
```

für „doppelzeilige“ Ausgabe:

```
\linespread{1.6}
```

Der Befehl wirkt vom Ort des Auftretens bis zum Ende des Dokumentes oder der aktuellen Umgebung.

Tabelle 3.4: Vertikale Abstände

`\smallskip`	etwa $\frac{1}{4}$ Zeile
`\medskip`	etwa $\frac{1}{2}$ Zeile
`\bigskip`	etwa 1 Zeile
`\vfill`	ein Abstand, der sich von 0 bis ∞ ausdehnen kann
`\vspace{...}`	exakte Angabe des Abstands

Spezielle vertikale Abstände

Zusätzliche Abstände zwischen zwei Zeilen *innerhalb* eines Absatzes oder einer Tabelle erreicht man mit dem Befehl

```
\\[Länge]
```

Die Abstände zwischen Absätzen, Kapiteln usw. werden von LaTeX automatisch bestimmt. In Spezialfällen kann man zusätzliche Abstände *zwischen zwei Absätzen* mit dem Befehl

```
\vspace{Länge}
\vspace*{Länge}
```

erreichen. Dieser Befehl sollte immer zwischen zwei Leerzeilen angegeben werden. Wenn der Abstand auch am Beginn oder am Ende einer Seite erhalten bleiben soll, wird analog zu `\hspace` ein `*` angehängt. Die angegebene Länge ist dabei eine *feste* Vorgabe, die von LaTeX nicht verändert werden kann. Dies gilt nicht für die Befehle `\smallskip`, `\medskip` und `\bigskip`.

3.1.7 Längenbefehle

Es wurde bereits angedeutet, daß LaTeX mit einer sehr großen Genauigkeit arbeitet. Intern kennt TeX viele verschiedene Maßangaben, die zum Teil individuell verändert werden können. Einige Längenangaben werden in der Tabelle 3.1 vorgestellt. Eine Änderung der eingestellten Maße erreicht man mit den Befehlen

```
\setlength{\Längenbefehl}{neue Länge}
\addtolength{\Längenbefehl}{Änderung}
```

Die Maßangaben bestehen grundsätzlich aus einer Dezimalzahl mit einem möglichen Vorzeichen, gefolgt von einer zwingend notwendigen Maßeinheit. Die möglichen Maßeinheiten sind in Tabelle 3.5 aufgeführt. Bei der Schreibweise der Zahlen ist sowohl ein Dezimalpunkt, als auch ein Komma erlaubt[3]. Hat eine Zahl keine Nachkommastelle, so kann der Punkt, bzw. das Komma entfallen. `16,3pt`, `19.7pt` und `15pt` sind also alles erlaubte Eingaben[4].

Grundsätzlich unterscheidet TeX bei der Verwendung zwischen *festen* und *elastischen* Maßen. Feste Maße können bei der Verarbeitung nicht verändert werden. Bei ihnen genügt die Angabe einer einzelnen Zahl mit Längeneinheit. Elastische Maße jedoch können gedehnt oder auch verkleinert werden. Zu diesem Zweck muß ihnen einen *Soll*-, ein *Dehn*- und ein *Schrumpf*-Wert übergeben werden. Die Syntax für die Zuweisung lautet also

```
\setlength{\Längenbefehl}{Soll plus Dehn minus Schrumpf}
```

Ein solches dehnbares Maß ist zum Beispiel der Abstand zwischen zwei Absätzen (`\parskip`). Gibt man als Definition zum Beispiel an:

```
\setlength{\parskip}{1.5ex plus 0.5ex minus 0.3ex}
```

[3] Dies gilt nicht für Positionsangaben in einer `picture`-Umgebung. Dort trennt ein Komma die beiden Koordinaten, während der Punkt als Dezimalpunkt dient. Aus diesem Grund sollte man sich die Schreibweise mit Dezimalpunkt angewöhnen.

[4] Sämtliche Maßangaben werden intern in `Punkte` als kleinste Einheit umgerechnet. Da LaTeX Maßangaben mit vier Nachkommastellen akzeptiert, bedeutet dies, daß LaTeX auf 0.0001pt $\approx 35 * 10^{-9}$m, also 35 Nanometer, genau rechnet!

Tabelle 3.5: Einheiten für Längenangaben

`mm`	Millimeter
`cm`	Zentimeter = 10 mm
`in`	Inch = 25,4 mm
`pt`	Point = $\frac{1}{72.27}$ in = 0,351 mm
`em`	die Breite eines `M` im aktuellen Zeichensatz
`ex`	die Höhe des Buchstabens `x` im aktuellen Zeichensatz

so hat das zur Folge, daß der Abstand zwischen Absätzen normalerweise das 1.5fache der Höhe eines x im momentanen Zeichensatz beträgt. Er kann jedoch auf maximal das Doppelte eines x gedehnt, oder auf das 1.2fache zusammengedrückt werden.

3.2 Textausrichtungen

3.2.1 Zentrierter Text

Mit der Umgebung

`\begin{center}`

Zeile 1 \\\\ *Zeile 2* \\\\ ... *Zeile n*

`\end{center}`

kann man Text horizontal zentriert setzen. Die einzelnen Zeilen werden mit \\\\ beendet (siehe Abb. 3.1). Fehlt diese Angabe oder ist eine Zeile zu lang, nimmt LaTeX den Zeilenumbruch selbst vor, allerdings ohne Silbentrennung.

Man kann darüberhinaus mit `\\[`*Länge*`]` abgeben, wie groß der Zeilenabstand sein soll. Der Parameter muß dabei eine der erlaubten Maßeinheiten besitzen.

Bei der `center`-Umgebung wird automatisch Abstand zum vorhergehenden und nachfolgenden Text gelassen. Innerhalb von anderen Umgebungen kann man Text auch mit `\centering` zentrieren. Dabei wird der gesamte nachfolgende Text bis zum Ende der Umgebung zentriert. Will man nur eine einzelne Zeile zentrieren, so dient dazu der Befehl

`\centerline{`*Zeilentext*`}`

links Backbord	`\begin{flushleft}` `  links \\` `  Backbord` `\end{flushleft}`
rechts Steuerbord	`\begin{flushright}` `  rechts  \\` `  Steuerbord` `\end{flushright}`
Die goldene Mitte	`\begin{center}` `  Die \\ goldene \\ Mitte` `\end{center}`

Abbildung 3.1: Linksbündig, rechtsbündig und zentriert

3.2.2 Links- und Rechtsbündig

Möchte man den Text links- oder rechtsbündig setzen, so kann man innerhalb von Umgebungen die beiden Befehle `\raggedright` (linksbündig) und `\raggedleft` (rechtsbündig) benutzen. Sie wirken bis zum Ende der Umgebung. Innerhalb der Umgebungen findet keine automatische Silbentrennung statt.

Darüber hinaus stehen die beiden Umgebungen `flushleft` und `flushright` zur Verfügung. Linksbündigen Text verwendet man manchmal in Briefen. Rechtsbündiger Text jedoch wird praktisch nie verwendet. Da Schrift von links nach rechts gelesen wird (mit Ausnahme arabischer Schriften, die ebenfalls mit LaTeX gesetzt werden können), wird das Lesen bei rechtsbündigem Text erheblich erschwert.

3.2.3 Beidseitige Einrückungen

Beidseitige Einrückungen des Textes dienen in der Regel dazu, Zitate, Verse, o. ä. zu setzen. Hierzu gibt es in LaTeX drei verschiedene Umgebungen.

Zitate

Die beiden Umgebungen `quote` und `quotation` rücken den Text auf beiden Seiten ein.

> Ähnlich wie bei der `center`-Umgebung wird bei diesen beiden vor und nach der Umgebung ein kleiner vertikaler Abstand freigelassen.
>
> Der Text der Umgebung darf beliebig lang sein. Befehle für Zeilen- oder Seitenumbruch können wie üblich benutzt werden.
>
> Absätze werden normal mit Leerzeilen voneinander getrennt. Aufgrund des Zwischenraumes vor und nach der Umgebung sollten Leerzeilen an Anfang oder Ende entfallen.

Der Unterschied zwischen den beiden Umgebungen ist folgender:

> Das erste Beispiel wurde mit `quote` erstellt. Dabei werden Absätze mit einem kleinen vertikalen Zwischenraum getrennt. Außerdem wird die erste Zeile nicht horizontal eingerückt.
>
> Anders bei der `quotation`-Umgebung. Hier werden Absätze ohne Abstand und dafür mit Einrückungen dargestellt. Daher sollte man diese Umgebung auch nur in Texten verwenden, in denen die Absätze auf diese Art gesetzt werden.

Verseinrückungen

Als weitere Möglichkeit der beidseitigen Einrückung von Text, steht die Umgebung `verse` zur Verfügung. Wie der Name bereits andeutet, ist sie vor allem zum Setzen von Gedichten, Versen u. ä. gedacht.

> Die erste Zeile wird nicht eingerückt.
> Innerhalb einer Strophe werden die Zeilen mit \\ umgebrochen.
> Die einzelnen Strophen werden durch Leerzeilen getrennt.
> Am Ende der Strophe steht kein \\.
>
> Ist eine Zeile zu lang für die beidseitig eingerückte Zeilenlänge, so wird sie automatisch umgebrochen. Dabei wird sie mit einer tiefer eingerückten Zeile fortgesetzt.

Die verschiedenen Umgebungen zum Einrücken von Text dürfen ineinander verschachtelt werden. Innerhalb z. B. einer `verse` Umgebung dürfen also weitere `verse`, `quote` oder `quotation`-Umgebungen auftreten. Bis zu sechs verschiedene Ebenen sind erlaubt. Die Einrücktiefe und der vertikale Abstand zur umgebenden Struktur nimmt dabei mit der Verschachtelungstiefe ab.

3.2.4 Texthervorhebungen

Oftmals möchte man in Texten einzelne Wörter hervorheben. Mit der Schreibmaschine erreicht man dies, indem das Wort unterstrichen wird. Im Buchdruck verwendet man dazu eine andere Schriftart. Entgegen der Annahme vieler Laien werden hervorgehobene Worte allerdings nicht **fett** gedruckt, sondern *kursiv*. In LaTeX dient dazu der Befehl

$\boxed{\texttt{\textbackslash emph\{...\}}}$

Er schaltet auf die „hervorstechende" („emphasize") Schriftart um. *Wird also der umgebende Text bereits kursiv gesetzt, dann schaltet der Befehl* auf die *aufrechte Schriftart* um.

Möchte man dennoch einmal Text unterstreichen, so dient dazu der Befehl

$\boxed{\texttt{\textbackslash underline\{\underline{Text}\}}}$

Da der angegebene Text intern in eine Box gesetzt wird, ist innerhalb des Textes kein Zeilenumbruch möglich! Man sollte stattdessen das `ulem`-Paket verwenden, das die Hervorhebung von Text umdefiniert. Das Paket bietet darüberhinaus auch andere Möglichkeiten, Text zu unterstreichen. Zunächst definiert das Paket den Befehl `\emph` neu, so daß der hervorgehobene Text unterstrichen wird. Da auch hier jedes Wort in eine Box gesetzt wird, ist die automatische Silbentrennung ausgeschaltet. Hier müßte man also von Hand eingreifen. Ein Seitenumbruch zwischen zwei Worten geschieht jedoch automatisch.

Einschalten kann man diese andere Form der Hervorhebung mit dem Befehl

$\boxed{\texttt{\textbackslash ULforem}}$

Sie gilt vom Auftreten des Befehls bis zum Ende der aktuellen Umgebung oder bis zum Auftreten des Befehls

$\boxed{\texttt{\textbackslash normalem}}$

mit dem wieder auf die normale Art der Hervorhebung zurückgeschaltet wird.

Nur zu unterstreichen ist etwas langweilig. Man kann auch ~~durchstreichen~~ oder ausstreichen (leserlich)	`Nur zu \uline{unterstreichen} ist` `etwas \uwave{langweilig}.` `Man kann auch \sout{durchstreichen}` `oder ausstreichen (\xout{leserlich})`

3.2.5 Silbentrennung

Automatischer Zeilenumbruch bei Blocksatz setzt eine automatische Silbentrennung voraus. Die Worttrennung bei englischen Texten gelingt TEX dabei besser als den meisten amerikanischen oder englischen Autoren!

Bei deutschen Texten gab es früher vor allem Probleme beim Trennen von Wörtern, in denen Umlaute auftreten. Dies lag daran, daß die Umlaute nicht als Buchstaben, sondern mit einem extra Befehl verarbeitet wurden. Mit der Verwendung der DC-Schriften ist dieses Problem behoben. Dennoch kann es vorkommen, daß einzelne Wörter, besonders Namen, zusammengesetzte Wörter oder Fremdworte, falsch oder ungünstig getrennt werden. Wenn die automatische Silbentrennung nicht das richtige Ergebnis liefert, kann man dies mit den folgenden Befehlen richtigstellen. Der Befehl

```
\hyphenation{Wörter}
```

bewirkt, daß die angeführten Wörter *jedesmal* nur an den mit „-“ markierten Stellen getrennt werden können. Er sollte am Beginn der Datei stehen. Dieser Befehl kann nicht für Zeichenfolgen genutzt werden, die Nummern oder Sonderzeichen enthalten.

```
\hyphenation{ Segel-yacht Segel-yachten }
```

Der Befehl `\-` innerhalb eines Wortes bewirkt, daß das Wort dieses eine Mal nur an den mit `\-` markierten Stellen getrennt werden kann. Dieser Befehl eignet sich für *alle* Zeichenfolgen, auch für solche, die Ziffern oder sonstige Sonderzeichen enthalten.

Bei dem Paket `german` steht der Befehl `"-` zur Verfügung. Er erlaubt die Trennung auch an den nicht angegebenen Stellen im Wort.

In der deutschen Sprache gibt es Wörter, die ihre Buchstabenfolge beim Trennen ändern. Um TEX diese Art der Trennung anzuzeigen, gibt es den Befehl

```
\discretionary{Vor}{Nach}{Ohne}
```

Die ersten beiden Parameter geben die Textteile für den Fall einer Silbentrennung an, der dritte den Text ohne Trennung.

Setzt man ein Wort z. B. in eine Box (`\mbox{...}`), so wird es in keinem Fall getrennt. Dies gilt auch, wenn die Box länger ist als die erlaubte Zeilenlänge. In diesem Fall ragt die Box nach rechts heraus und TEX gibt eine Warnung (“overfull hbox”) aus.

Drucker oder Druk-
ker
In Afrika gibt es sehr, sehr viele große Gürteltiere.
In Afrika gibt es sehr, sehr viele große Gür-
teltiere.

```
Dru\discretionary{k-}{k}{ck}er
Drucker oder Drucker \\
In Afrika gibt es sehr, sehr viele
große \mbox{Gürteltiere}. \\
In Afrika gibt es sehr, sehr viele
große Gürteltiere.
```

3.3 Spezielle Zeichen

LaTeX bietet verschiedene Sonderzeichen, die nicht auf einer normalen Tastatur zur Verfügung stehen. Diese müssen mit entsprechenden Befehlen erzeugt werden.

3.3.1 Anführungszeichen

Für Anführungszeichen werden im Buchdruck *nicht* das auf Schreibmaschinen übliche Quotes-Zeichen (`"`) verwendet. Statt dessen wird zwischen öffnenden und schließenden Anführungszeichen unterschieden. Die englische Anführungszeichen erhält man durch Eingabe von zwei Grave-Akzenten (öffnend) bzw. zwei Apostrophen (schließend). In deutschen Texten werden andere Anführungszeichen („Gänsefüßchen“) verwendet. Diese erhält man durch Eingabe von `"‘` (Grave) für die öffnenden und `"’` (Apostroph) für die schließenden.

“No,” he said, “I don’t know!”
„Ja“, sagte er, „Ich weiß es!“

```
‘‘No,’’ he said, ‘‘I don’t know!’’ \\
"‘Ja"’, sagte er, "‘Ich weiß es!"’
```

Verschiedene Editoren unterstützen den Autor bei der Auswahl der Anführungszeichen. So erkennt der Emacs bei Verwendung des auctex-Paketes automatisch, ob öffnende oder schließende Anführungszeichen benötigt werden. Schreibt man deutsche Texte, so kann man diese in der Datein `~/.emacs` explizit definieren:

```
(add-hook 'LaTeX-mode-hook
          '(lambda ()
             (setq TeX-open-quote "\"‘")
             (setq TeX-close-quote "\"'")))
```

3.3.2 Trenn-, Binde- und Gedankenstriche

Es gibt im Buchdruck verschiedene Striche für Bindestriche, Gedankenstriche und Minus-Zeichen. Die unterschiedlich langen Striche werden in LaTeX durch Kombinationen von Minus-Zeichen erzeugt. Das Minus-Zeichen als mathematisches Zeichen ist ebenfalls länger als ein Trennstrich. Der ganz lange Gedankenstrich (—) wird im deutschen in der Regel nicht benutzt.

10–18 Uhr	`10--18 Uhr \\`
ja – oder nein?	`ja -- oder nein? \\`
yes—or no?	`yes---or no? \\`
$0,1$ und -1	`$0, 1$ und $-1$`

3.3.3 Ligaturen

Im Buchdruck werden manche Buchstabenkombinationen enger gesetzt als die jeweiligen Einzelbuchstaben. Diese Ligaturen erleichtern das Lesen eines Textes.

ﬀ ﬁ ﬂ AV Te ... statt ff fi fl AV Te ...

Ligaturen sind schriftspezifisch. Bei Änderung der Schriftart kann es also passieren, daß keine Ligaturen erkennbar sind. Sie können unterdrückt werden, indem man zwischen die Buchstaben den Befehl `\/` setzt. Um das Lesen des Textes zu erleichtern, sollte dies immer dann gemacht werden, wenn solche Buchstabenkombinationen nach Vorsilben oder bei zusammengesetzten Wörtern zwischen den Wortteilen auftreten. Mit dem Paket `german` steht zusätzlich der Befehl `"|` zur Verfügung.

Nicht Auﬂage (Au-ﬂ-age)	`Nicht Auflage (Au-fl-age) \\`
sondern Auflage (Auf-lage)	`sondern Auf\/lage (Auf-lage) \\`
oder Auflage	`oder Auf"|lage`

3.3.4 Punkte

Anders als bei der Schreibmaschine, bei der jeder Punkt und jedes Komma mit einem der Buchstabenbreite entsprechenden Abstand versehen ist, werden Punkte und Kommata im Buchdruck eng an das vorangehende Zeichen gesetzt. Für Fortsetzungspunkte gilt dies nicht. Daher gibt es dafür den Befehl `\dots`.

Nicht so ...	`Nicht so ... \\`
sondern so:	`sondern so: \\`
Achim, Florian, . . .	`Achim, Florian, \dots`

3.3.5 Akzente und spezielle Buchstaben

LaTeX ermöglicht die Verwendung von Akzenten und speziellen Buchstaben aus sehr vielen verschiedenen Sprachen. Mit den Schriften der DC-Familie können alle Schriftzeichen aus folgenden Sprachen erzeugt werden:

> Afrikaans, Albanisch, Bretonisch, Dänisch, Deutsch, Englisch, Estnisch, Färörisch, Finnisch, Friesisch, Gälisch, Indonesisch, Irisch, Isländisch, Italienisch, Katalanisch, Madagassisch, Niederländisch, Niedersorbisch, Norwegisch, Obersorbisch, Polnisch, Portugiesisch, Rätoromanisch, Rumänisch, Schwedisch, Serbokroatisch, Slowakisch, Slowenisch, Somali, Spanisch, Suaheli, Tschechisch, Türkisch und Ungarisch.

Bei diesen Sprachen können alle Sonderzeichen und diakritischen Buchstaben erzeugt und in die jeweiligen Trennmustertabellen eingebunden werden, um eine optimale Bearbeitung durch LaTeX zu erreichen.

Tabelle 3.6: Akzente und spezielle Buchstaben

Eingabe	Ausgabe	Eingabe	Ausgabe	Eingabe	Ausgabe
\‘o	ò	\’o	ó	\ˆo	ô
\˜o	õ	\=o	ō	\.o	ȯ
\u␣o	fio	\v␣o	ǒ	\H␣o	ő
\"o	ö	\c␣o	o̧	\d␣o	ọ
\b␣o	o̲	\r␣o	o̊	\t␣oo	o͡o
\oe	œ	\OE	Œ	\ae	æ
\AE	Æ	\aa	å	\AA	Å
\o	ff	\O	ffi	\l	ł
\L	Ł	\i	ı	\j	ȷ

Tabelle 3.6 zeigt eine Auswahl der Akzente und speziellen Buchstaben. Die Akzente werden darin jeweils am Beispiel des Buchstabens o gezeigt. Sie

können aber mit jedem Buchstaben kombiniert werden. Zu beachten ist hierbei die Verwendung von Leerzeichen. Möchte man einen Akzent auf ein i oder ein j setzen, so muß der i-Punkt weggelassen werden. Dies erreicht man mit den Befehlen `\i` und `\j`.

Da leider die Zeichensatzkodierung bei der Eingabe von ASCII-Texten nicht genormt ist[5], ist es nicht unbedingt möglich, z. B. Umlaute direkt einzugeben. Es stehen jedoch verschiedene Zusatzpakete zur Verfügung, die dieses Problem beheben. Für europäische Schriftzeichen dient hierzu das `isolatin1`-Paket.

3.4 Schriftart und -größe

Bei WYSIWYG-Systemen wird oftmals mit der Zahl der verschiedenen verfügbaren Schriften geworben. Das hat dazu geführt, daß viele Laien der Meinung sind, die Verwendung vieler Schriftarten erhöhe die Qualität eines Schriftstückes. Dies ist jedoch ein großer Irrtum. Viele verschiedene Schriften erschweren in der Regel die Lesbarkeit eines Textes. Dennoch kann der *sinnvolle* Einsatz unterschiedlicher Schriftarten durchaus hilfreich zu sein.

Tabelle 3.7: Schriftarten

`\textrm{...}`	`{\rmfamily ...}`	`{\rm ...}`	Serifenschrift
`\textsf{...}`	`{\sffamily ...}`	`{\sf ...}`	serifenloser Schrift
`\texttt{...}`	`{\ttfamily ...}`	`{\tt ...}`	`Typewriter-Schrift`
`\textmd{...}`	`{\mdseries ...}`		Schriftstärke medium
`\textbf{...}`	`{\bfseries ...}`	`{\bf ...}`	**fettgedr. Schrift**
`\textup{...}`	`{\upshape ...}`		senkrechter Schrift
`\textit{...}`	`{\itshape ...}`	`{\it ...}`	*kursive Schrift*
`\textsl{...}`	`{\slshape ...}`	`{\sl ...}`	*schräggest. Schrift*
`\textsc{...}`	`{\scshape ...}`	`{\sc ...}`	KAPITÄLCHEN
`\emph{...}`		`{\em ...}`	Text *hervorheben*
`\textnormal{...}`	`{\normalfont ...}`		Basisschrift

Die bei TeX zur Verfügung stehenden Schriften sind in ihrem Aussehen aufeinander abgestimmt, um ein möglichst ausgewogenes Erscheinungsbild zu

[5] Verschiedene Betriebssysteme verwenden für die Sonderzeichen andere Codes.

Tabelle 3.8: Schriftgrößen

`\tiny`	winzig kleine Schrift	`\scriptsize`	sehr kleine Schrift (wie Indizes)
`\footnotesize`	kleine Schrift (wie Fußnoten)	`\small`	kleine Schrift
`\normalsize`	normale Schrift	`\large`	große Schrift
`\Large`	größere Schrift	`\LARGE`	sehr große Schrift
`\huge`	riesig groß	`\Huge`	gigantisch

erreichen. Allerdings kann man in einem normalen Text nur relativ wenige verschiedene Schriften nutzen. Diese sind in Tabelle 3.7 aufgeführt.

Normalerweise wählt LaTeX die geeigneten Schriftarten und Schriftgrößen anhand der Befehle, die die logische Struktur des Textes beschreiben (Überschriften, Hervorhebungen usw.). In Einzelfällen kann man die Schriftart und -größe auch explizit mit den in den Tabellen 3.7 und 3.8 aufgeführten Befehlen verändern. Die kurzen Befehle verwendet man innerhalb von Gruppen und Klammern.[6] Für kurze Texte kann man die `\textXX`-Befehle nutzen, die als Argument den entsprechenden Text erhalten.

Zur Auswahl der Schriftgröße stehen die Befehle aus Tabelle 3.8 zur Verfügung. Eine Änderung der Schriftgröße ändert auch die Zeilenabstände. Wenn die Leerzeile, die den Absatz beendet, noch innerhalb der Gültigkeit des Größen-Befehls steht, so wird auch der Abstand zum folgenden Absatz verändert. Die Gruppe darf also nicht zu früh beendet werden! Anders als bei anderen Textverarbeitungsprogrammen erfolgt die Angabe der Schriftgröße nicht *absolut*, sondern *relativ* zu der in der verwendeten Klasse festgelegten Größe. Der Wert für `\normalsize` beträgt dabei entweder 10, 11 oder 12pt.

Dies sind KAPITÄLCHEN.

```
Dies sind \textsc{Kapitälchen}
```

Dies ist **klein, fett und *kursiv***.

```
Dies ist {\small \bfseries
  klein, fett und {\itshape kursiv}}.
```

Groß !!!

```
\textit{\Large Groß !!!}
```

[6] In der alten Version LaTeX2.09 wurden nur diese Befehle benutzt. Die kurzen Befehle in LaTeX 2_ε sind nur eingeführt worden, um eine bessere Übertragbarkeit alter Dokumente zu erreichen. Man sollte bei neuen Dokumenten immer die längeren Versionen verwenden.

3.4.1 Low-Level Befehle im NFSS

Die Schriftauswahl wird in LaTeX 2ε mit dem „New Font Selection Sheme“ (NFSS) gesteuert (LaTeX3 Project Team, 1995a). Gegenüber den Mechanismen in der alten Version LaTeX2.09 ist es hier möglich, Schriftbefehle zu kombinieren. Die Befehle `\bf\it` führten früher z. B. nicht zu der gewünschten kursiven, fetten Schrift. Verwendet man jetzt `\bfseries\itshape` so erhält man ***das gewünschte Ergebnis***. Es ist also möglich, einzelne Parameter der Schrift zu ändern; vorausgesetzt, es steht der entsprechende Zeichensatz zur Verfügung.

Es soll hier nur eine kurzer Überblick über die verschiedenen Low-Level Befehle gegeben werden. Die Schriften werden anhand folgender Kriterien beschrieben: Kodierung, Familie, Serie, Form und Größe. Diese werden repräsentiert durch die Parameter `\fontencoding`, `\fontfamily`, `\fontseries`, `\fontshape` und `\fontsize`.

Möchte man z. B. die Schriftgröße über die normalen Befehle hinaus ändern, so kann man die Schrift mit dem Befehl

```
\fontsize{Größe}{Durchschuß}
```

skalieren. *Größe* gibt dabei die gewünschte Größe der Schrift an. Wird die Einheit weggelassen, so wird automatisch `pt` angenommen. *Durchschuß* bezeichnet den Zeilenabstand. Ausgewählt wird die neue Schrift mit dem Befehl

```
\selectfont
```

Es kann jedoch vorkommen, daß kein externer Zeichensatz in der gewünschten Größe vorhanden ist. Die möglichen Schriftgrade sind in der Schriftdefinition (`fd`-Datei) festgelegt.

Dieser Absatz ist mit den unten aufgeführten Befehlen gesetzt. Dabei wird die Palatino Schrift verwendet. Sie könnte für das gesamte Dokument eingestellt werden, indem einfach das `palatino` Paket geladen wird.

```
\fontencoding{T1}\fontfamily{ppl}\fontseries{n}\fontshape{m}
\fontsize{11}{13}\selectfont
```

3.5 Zusätzliche Schriften

Es gibt eine große Anzahl zusätzlicher Schriftarten, die in LaTeX-Dokumenten verwendet werden können. Im Folgenden soll ein kurzer Überblick über verschiedene Zusatzpakete für Schriften gegeben werden. Die Auswahl erhebt auf keinen Fall den Anspruch auf Vollständigkeit. Sie soll Gegenteil nur einen Überblick geben, für welche Bereiche Schriften und Sonderzeichen zur Verfügung stehen. Eine große Liste von Schriften hat Marcus J. Ertl zusammengestellt.[7]

3.5.1 PostScript-Schriften

Außer den bereits vorgestellten Schriften können alle PostScript-Schriften verwendet werden. Diese bieten eine Reihe von Vorteilen. Zum einen sind PostScript-Dateien wirklich von der Hardware unabhängig. Zum anderen bieten sie eine große Auswahl verschiedener Schriftarten, die außerdem frei skalierbar sind, so daß man sie in jeder gewünschten Größe verwenden kann.

Der einfachste Weg, diese Schriften zu verwenden, ist das Laden eines entsprechenden Zusatzpaketes. Welche vorhanden sind, ist von der Installation abhängig.

3.5.2 Altdeutsche Schriften

Die häufigsten im modernen Buchdruck verwendeten Schriften werden als „Antigua-Schriften" bezeichnet. Dies mag den Laien zunächst verwundern, bedeutet der Name übersetzt doch schließlich „alte Schriften". Der Name ist aber durchaus berechtigt, denn einige der unter diesem Namen zusammengefaßten Schriften gehen bis ins Altertum zurück. Die seit dem Frühmittelalter im romanischen Sprachraum verwendeten Antigua-Schriften basieren mit ihren Kleinbuchstaben auf den sog. karolinischen Minuskeln und mit den Großbuchstaben auf den römischen Majuskeln.

Neben diesen Schriften entwickelten sich bereits recht früh die gotischen Schriften, z. B. die Fraktur-Schrift. Diese zeichnen sich im Gegensatz zu den runden, flüssigen Formen der Antigua-Schriften durch strenge, hochstrebende Zeichen aus, bei denen die runden Formen durch eckige, gebrochene Linien ersetzt sind. Im germanischen und slawischen Sprachraum beherrschten diese „gebrochenen" Schriften teilweise bis in unser Jahrhundert hinein das Druckbild. In Deutschland war bis zu einem Verbot 1941 Fraktur die vorherrschende Druckschrift.

Viele Geisteswissenschaftler, besonders Germanisten, Historiker und Theologen waren sehr daran interessiert, diese Schriften auch im computerunterstützen Buchsatz verwenden zu können. Die Möglichkeit hierzu ist Yannis Haralambous zu verdanken, der drei altdeutsche Schriften mit Hilfe von Metafont nachgebildet hat. Zu diesen drei Schriften hat er auch noch einen sehr schönen Zeichensatz mit Initialen geschaffen.

[7] `http://homepages.hrz.tu-darmstadt.de/ st002556/texfonts/liste.html`

Man erhält die Schriften mit dem `oldgerm` Paket. Das Paket stellt die Befehle

```
\gothfamily
\frakfamily
\swabfamily
```

bereit. Mit ihnen kann man Texte in gotiſcher Schrift, auch Textur genannt, Fraktur-Schrift und Schwabacher setzen. Selbstverständlich existieren auch die kurzen Schriftbefehle

```
\textgoth{Text}
\textfrak{Text}
\textswab{Text}
```

Die Schriften bestehen jeweils nur aus einer Schriftserie mit einem Schriftschnitt. Befehle wie `\itshape` sind also unwirksam. Die Größenbefehle können aber selbstverständlich verwendet werden.

Gotische Schrift

Die gotische Schrift entspricht bis auf die Großbuchstaben, die jedoch ebenfalls aus dem 15. Jahrhundert stammen, der von Johannes Gutenberg für seinen ersten Bibeldruck verwendeten Schrift. Die Schrift besteht wie alle gebrochenen Schriften aus 25 Groß- und 27 Kleinbuchstaben. Bei den Großbuchstaben wird nicht zwischen `I` und `J` unterschieden, bei den Kleinbuchstaben gibt es dagegen zwei verschiedene Formen für das `s`. Es wird zwischen dem „langen s“ ſ und dem „runden s“ s unterschieden. Das lange ſ wird im Wortinneren verwendet, das kurze s am Wortende. Außerdem ist zu beachten, daß das kurze s bei zusammengesetzten Worten ebenfalls im Wortinneren verwendet wird: Ausgang aber Eſſen. LaTeX 2_ε kann dieses leider nicht von alleine beachten, daher muß man bei der Eingabe an ein rundes s ein ':' anhängen, also `Aus:gang` aber `Essen`.

Neben den beiden s-Formen bietet die gotische Schrift eine große Zahl Ligaturen. Im einzelnen sind das:

```
ae ba be bo ch ck ct da de do ha he ho ff fi ffi ffl ij ll
oe pa pe po pp qq qz ss ssi st sz tz va ve vu
```

Diese ergeben (hier vergrößert dargestellt):

ae ba be bo ch ck ct da de do ha he ho ff fi ffi ffl ij ll
oe pa pe po pp qq qz ſſ ſſi ſt ſz tz va ve vu

Die Ligaturen können wie gewohnt mit `\/` aufgehoben werden. Da die Schrift in dem Kodierschema `U` vorliegt, ist es nicht möglich, Umlaute mit den normalen Befehlen einzugeben. Verwendet man das Paket `german`[8], so kann man mit dem Befehl `\originalTeX` die normale Behandlung der Umlaute einstellen. Nun kann man mit `"a`, `"e`, `"o`, `"s` und `"u` die Umlaute eingeben. Mit dem Befehl `\germanTeX` wird wieder die Behandlung durch das Paket `german` eingestellt. Wie alle altdeutschen Schriften unterscheidet die Gotische Schrift nicht zwischen `I` und `J`. Das Ausgabezeichen ist jeweils das gleiche: J.

Schwabacher Schrift

Am Ende des 15. Jahrhundert kam zusätzlich die Schwabacher Schrift auf. In ihr gingen die strengen Linien der Gotischen Schrift teilweise in runde Formen über. Dennoch gehört auch sie zu den gebrochenen Schriften.

Die Schwabacher Schrift bietet nur sieben Ligaturen: `tt ch ck ff sf ss st sz`. Sie ergeben tt ch ck ff ſf ſſ ſt ß. Neben den normalen Umlauten können auch die alten Formen mit einem hochgestellten kleinen e erzeugt werden:

`*a *e *o *u` aͤ eͤ oͤ uͤ

Fraktur Schrift

Die Fraktur Schrift kam am Beginn des 16. Jahrhunderts auf. Sie ist vor allem durch die sehr schmalen Kleinbuchstaben und die sog. „Elefantenrüssel“, mit denen viele Großbuchstaben geschmückt sind, gekennzeichnet. Die erste Fraktur-Schrift wurde von Johann Schönsperger 1513 in Augsburg entwickelt, aber besonders im 18. Jahrhundert gab es durch Schriftschneider wie G. I. Breitkopf und J. F. Unger eine Reihe Änderungen. Die Entwürfe von Breitkopf dienten auch Yannis Haralambous als Vorlage.

Die Fraktur-Schrift enthält alle Zeichen der Schwabacher-Schrift. Zusätzlich noch die Ligatur tz für `tz`. Darüberhinaus bietet sie eine Unterscheidung von `I` und `J`. Das Zeichen J für das `J` kann über den Befehl `\symbol{36}` erreicht werden. Es ist allerdings zu beachten, daß sich dieses Zeichen kaum durchgesetzt hat.

Initialen

In vielen alten Schriften wurden die ersten Zeichen eines Abschnittes oder Kapitels oftmals besonders ausgeschmückt. Yannis Haralambous hat für das `oldgerm` Paket einen derartigen Zeichensatz entwickelt. Abbildung 3.2 zeigt

[8] Je nach Installation wird dieses Paket oft als Voreinstellung geladen. Ein Blick in die Protokolldatei wird hierüber Aufschluß geben.

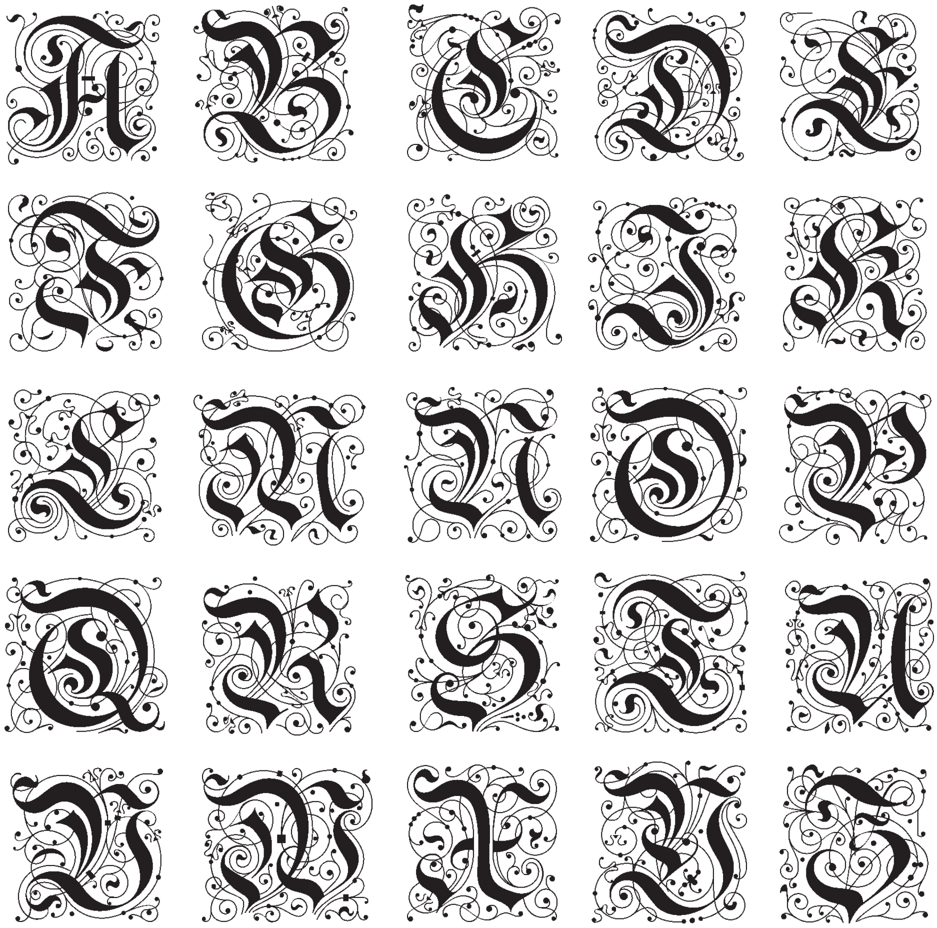


Abbildung 3.2: Initialen von Yannis Haralambous

(vergrößert) alle verfügbaren Initialen. Leider enthält das Paket keine Befehle, die die Verwendung dieser Buchstaben direkt ermöglicht. Ich habe daher mein Paket um die folgenden Zeilen erweitert:

```
\DeclareTextFontCommand{\textinit}{\initfamily}
\newcommand\initfamily{\usefont{U}{yinit}{m}{n}}
\newdimen\inittmp
\newcommand{\yinit}[1]{\setbox0=\hbox{\textinit{#1}}%
  \hangindent=\wd0\hangafter=-4 \advance\hangindent by .2em%
  \inittmp=.9\ht0 \hskip-\wd0 \hskip-.25em%
  \raisebox{-\inittmp}[0pt][0pt]{\unhbox0}\hskip.25em}
\newcommand{\yinitpar}[1]{\noindent\yinit{#1}}
```

Ich möchte an dieser Stelle nicht auf die genaue Bedeutung der einzelnen TeX-Befehle eingehen. Wer mehr über die Verwendung der TeX-internen Boxbefehle wissen möchte, sei an das Buch von Norbert Schwarz (Schwarz, 1991) verwiesen. Wenn die nötigen Font-Dateien (`yinit.tfm`, `yinit.fd` und `yinit.mf`) vorhanden sind, kann man mit den Schriftbefehlen `\textinit` und `\initfamily` die Schrift auswählen. Da man jedoch in der Regel keine längeren Texte mit Initialen schreibt, sondern nur ein Zeichen, kann man einen entsprechenden Absatz mit dem Befehl

`\yinitpar{`*Initiale*`}`

beginnen. Der Befehl setzt den entsprechenden Buchstaben in der ersten Zeile tiefergestellt. Die ersten vier Zeilen des Absatzes werden horizontal eingerückt. Da die Wirkung jedoch am Ende des Absatzes endet, muß der Absatz mindestens vier Zeilen lang sein. Andernfalls kann es sein, daß der nachfolgende Absatz in die Initiale hineinreicht. Als Beispiel für die Verwendung dient in Abbildung 3.3 ein Zitat von Jean Jacques Rousseau.

Jean Jacques Rousseau
Auszug aus dem Plan des Ewigen Friedens
des Herrn Abbe de Saint-Pierre
1756 / 1761

Wie niemals ein größerer, schönerer und nützlicherer Plan den menschlichen Geist beschäftigt hat als der eines ewigen und umfassenden Friedens unter allen Völkern Europas, so hat auch kein Autor in stärkerem Maße Beachtung der Öffentlichkeit verdient als derjenige, der Mittel und Wege zu Verwirklichung dieses Planes vorschlägt. Man kann sich schwer vorstellen, daß ein solcher Gegenstand einen empfindsamen und edlen Menschen völlig unberührt sein läßt und ich weiß nicht, ob die Phantasie eines wahrhaft menschenfreundlichen Herzens, dessen Begeisterungsfähigkeit alle Dinge möglich erscheinen läßt, nicht doch der herben, kalten Vernunft vorzuziehen ist, deren Gleichgültigkeit gegenüber dem öffentlichen Wohl das größte Hindernis ist für jede Bemühung, dieses zu fördern.

Ich zweifle nicht, daß viele Leser von vorneherein sich mit Ungläubigkeit wappnen, um der Freude zu widerstehen, überredet zu werden, und ich beklage sie, weil sie so jämmerlich ihren Eigensinn für Weisheit halten. Ich hoffe jedoch, daß wenigstens eine fühlende Seele mit mir die köstliche Erhebung teilen wird, mit welcher ich die Feder für ein der Menschheit so wichtiges Anliegen ergreife.

Abbildung 3.3: Beispiel für die Verwendung des `oldgerm` Paketes

3.5.3 Unzialschrift

Die sog. „Unzialschrift“ (lat.: *uncus* ›gekrümmt‹) ist aus den römischen Majuskeln entstanden. Sie ließ sich auf Pergament sehr gut schreiben und wurde so als Buchschrift die Hauptschrift zwischen dem 4. und 8. Jahrhundert. Ihr verwandt ist die „Halbunzialschrift“, die durch Kursiv- und Minuskelelemente geprägt ist.

Für LaTeX stehen insgesamt drei derartige Schriften zur Verfügung. Sie wurden von Peter Wilson entworfen. Das Paket `uncial` bietet eine Version der Schrift, die vor allem zwischen dem 3. und 6. Jahrhundert verwendet wurde. Die Version des `auncial`-Paketes wurde hauptsächlich zwischen dem 6. und 10. Jahrhundert benutzt. Das Paket `huncial` stellt die Halbunzialschrift zur Verfügung. Sie war zwischen dem 3. und 9. Jahrhundert im Gebrauch.

In allen drei Schriften wurden Großbuchstaben nur sehr wenig benutzt, in der Regel nur am Anfang eines Absatzes. Sie waren einfach größere Versionen der normalen Buchstaben. Außerdem stehen keine Umlaute zur Verfügung.

Ausgewählt werden die Schriften mit den Befehlen

```
\aunclfamily
\hunclfamily
\unclfamily
```

Selbstverständlich können auch die kurzen Schriftbefehle verwendet werden:

```
\textauncl{Text}
\texthuncl{Text}
\textuncl{Text}
```

`uncial`:
Verfügbare Zeichen: `abcdefghijklmnopqrstuvwxyz 1234567890 ,.-;: ‘ ’`
abcdefghijklmnopqrstuvwxyz 1234567890 ,.-;: ‘ ’

`auncial`:
Verfügbare Zeichen: `abcdefghijklmnopqrstuvwxyz 1234567890 ,.-;:& ‘ ’`
abcdefghijklmnopqrstuvwxyz 1234567890 ,.-;:& ‘ ’

`huncial`:
Verfügbare Zeichen: `abcdefghijklmnopqrstuvwxyz 1234567890 ,.-;:& ‘ ’`
abcdefghijklmnopqrstuvwxyz 1234567890 ,.-;:& ‘ ’

3.5.4 Schreibschriften

Schreibschriften per Computer zu setzen, erfordert einigen Aufwand. Das Hauptproblem dabei sind die unterschiedlichen Verbindungen zwischen den einzelnen Buchstaben. Für LaTeX 2ε gibt es dafür insgesamt drei Schriften.

Das `la`-Paket von Johannes Heuer erzeugt neue Schriftgrößenbefehle, bei denen ein `la` vor den normalen Befehl vorangestellt wird. Der folgende Absatz wurde mit dem Befehl `\lalarge` begonnen.

Das LA-Paket erzeugt Schreibschrift. Bei genauem Hinsehen wird man allerdings feststellen, daß einige Buchstaben nicht ganz korrekt aneinander gesetzt werden. Dieses Problem tritt vor allem bei Umlauten und dem ß auf.

Setzt man dem Größenbefehl `lla` statt `la` voran, so werden zusätzlich noch Linien gezogen.

Eine weitere Schreibschrift hat Thomas Weinmann mit dem `twcal14`-Paket bereitgestellt. Die Schrift existiert nur in einer Größe (14pt), außerdem sind keine Umlaute verfügbar. Ausgewählt wird die Schrift mit dem Befehl

`\twcal`

Dieser Schrift gelingt es durch die Verwendung sehr vieler Ligaturen besser, die einzelnen Buchstaben lueckenlos aneinander zu setzen.

Die dritte Möglichkeit bietet das `va`-Paket. Diese Schrift basiert auf der Vereinfachten Ausgangsschrift. Erzeugt wird die Schrift, die in den Standardgrößen vorhanden ist, mit dem Befehl

`\vafont`

Die Schrift va *basiert auf der Vereinfachten Ausgangsschrift. Selbstverständlich können Umlaute ohne Probleme verwendet werden.*

Eine kalligraphische Version der Schrift steht ebenfalls zur Verfügung. Sie kann gewählt werden mit dem Befehl

`\vacalfont`

3.5.5 Asiatische Schriften

Das Setzen nicht-europäischer Schriften erzeugt ähnliche Probleme wie Schreibschriften. Liegt liegt die Ursache hauptsächlich in einem anderen und oftmals längeren Alphabet. Daher sind viele Zeichen auch nur über Zusatzbefehle oder mit dem Befehl `\symbol` zu erhalten. Für LaTeX stehen eine ganze Reihe solcher Schriften zur Verfügung, so z. B. für Thai oder Japanisch. Da ich beide Sprachen nicht beherrsche, beschränke ich mich an dieser Stelle darauf, beispielhaft den `usl-thai` Zeichensatz zu zeigen. Dieser Zeichensatz wurde von P. Kositsurungkakul und C. Lusinsap entworfen. Er enthält insgesamt 63 Zeichen.

`usl-thai`:
Verfügbare Zeichen: `abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!+,12345789`

3.5.6 Antike Schriften

Für LaTeX sind auch zahlreiche Schriften verfügbar, die zu längst ausgestorbenen Sprachen gehören. Beispielhaft sollen hier vier verschiedene Schriften vorgestellt werden.

Arabische Schrift

Die arabische Schrift von Alan Stanier wurde im 2. Vorchristlichen Jahrhundert in vielen Sprachen in Süd-Arabien verwendet. Der Zeichensatz heißt `SouthArabian`.

`SouthArabian`:
Verfügbare Zeichen: `bdfghklmnqrstwyzDGHSTZ12345‘’`

Ugarit

Ugarit ist der Name einer antiken kanaanäischen Stadt an der Mittelmeerküste, die seit etwa 2000 v. Chr. nachweisbar ist. Bei Ausgrabungen wurden zahlreiche Tontafeln gefunden, die Schriftzeichen enthielten. Alan Stanier hat diese Zeichen als Grundlage für seinen Zeichensatz `ugarit` benutzt. Es sind insgesamt 30 Zeichen verfügbar. Der Gedankenstrich (-) kennzeichnet dabei den Wortzwischenraum.

`ugarith`:

Verfügbare Zeichen: `abdghiklmnpqrstuwyzDGHST01234-`

Römische Schrift

Die Beschriftung der Trajans-Säule in Rom stand Pate für die Zeichensätze des `trajan`-Paketes von Peter Wilson. Das Paket enthält zwei verschiedene Zeichensätze, zum einen die Normalschrift, zum anderen eine leicht geneigte Variante. Es sind jeweils nur Großbuchstaben und einige Satzzeichen verfügbar.

`trjn`

Verfügbare Zeichen: `ABCDEFGHIJKLMNOPQRSTUVWXYZ ,.-;:&‘ ’!?`

ABCDEFGHIJKLMNOPQRSTUVWXYZ ,.-;:&‘ ’!?

ABCDEFGHIJKLMNOPQRSTUVWXYZ ,.-;:&‘ ’!?

Runenzeichen

Auch Runenzeichen können mit LaTeX erzeugt werden. Dabei stehen sogar gleich mehrere Schriften zur Verfügung. Die Schrift `srune` von Jobst-Hartmut Lüddecke enthält zum einen das alte Furthac-Alphabet, benannt nach den ersten sechs Zeichen, und zum anderen zusätzliche Zeichen aus anderen Runenschriften. Insgesamt enthält der Zeichensatz 43 Zeichen, darunter 15 Ligaturen.

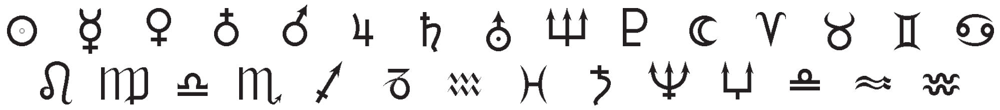
Eingabe	Rune	Eingabe	Rune	Eingabe	Rune	Eingabe	Rune
F	ᚠ	J	ᛃ	D	ᛞ	EO	Z
U	ᚢ	Y	ᚣ	O	ᛟ	AE	ᚫ
TH	ᚦ	P	ᛈ	X	ᛪ	EA	ᛠ
A	ᚨ	Z	ᛉ	K	ᛕ	AH	ᚽ
R	ᚱ	S	ᛊ	V	Λ	AA	ᚽ
C	ᚲ	T	ᛏ	EE	ᛇ	GE	ᛄ
G	ᚷ	B	ᛒ	NG	ᛜ	TS	ᛥ
W	ᚹ	E	ᛖ	CK	ᛤ	RR	ᛦ
H	ᚺ	M	ᛗ	ST	ᛞ	:	:
N	ᚾ	L	ᛚ	GG	ᚸ	.	.
I	ᛁ	Q	ᛩ	IO	ᛡ		

3.5.7 Sonderzeichensätze

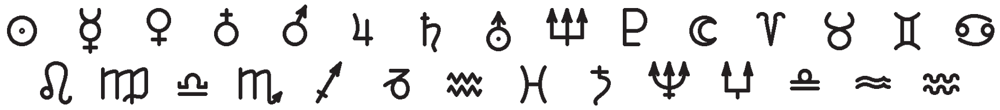
Mittlerweile existieren auch viele Zeichensätze für Spezialanwendungen. Hierzu zählen beispeispielsweise die Schachzeichen von Piet Tutelaers, mit denen sehr einfach ein Schachbrett oder auch eine Zugdokumentation dargestellt werden kann; oder auch der Barcode-Zeichensatz von Dimitri Vulis.

Ein weiters Beispiel für Sonderzeichen stellen die astronomischen Zeichen dar. Der Zeichensatz, der von Peter Schmitt entworfen wurde, enthält jeweils 29 astronomischen Planeten- und Tierkreissymbole in drei verschiedenen Varianten.

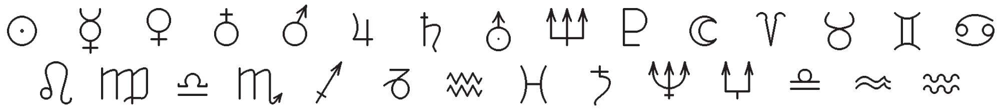
„Kalligraphische Version" (Nummer 0 – 28):



„Uniform Version (fett)" (Nummer 100 – 128):



„Uniform Version (dünn)" (Nummer 200 – 228):



Die Zeichen können mit dem Befehl `\symbol{`*Nummer*`}` ausgewählt werden. Die Nummern sind in dem obigen Beispiel angegeben.

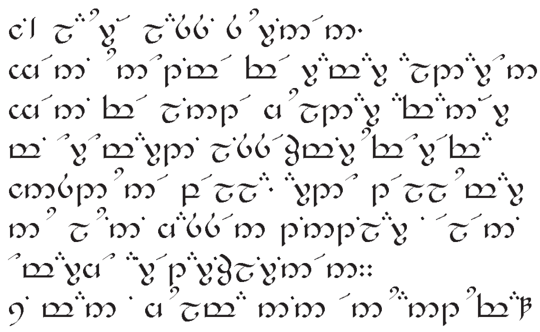
3.5.8 Phantasie-Schriften

Klingonische Schrift

Es gibt für $\LaTeX\,2_\varepsilon$ eine Reihe Schriften, die eher in den Bereich Spielerei gehören. Mit diesen ist es zum Beispiel möglich, Texte in klingonischer Schrift zu setzen. Obwohl diese Schrift wohl vor allem etwas für eingefleischte StarTrek-Fans ist, möchte ich hier dennoch eine kleine Kostprobe geben. Interessierte Leser können versuchen, den Text zu entschlüsseln …

Elbische Schriften

Auch die von John Ronald Reuel Tolkien für seine Romantrilogie „Der Herr der Ringe“ entworfenen Schriften Tengwar und Cirth sind für LaTeX verfügbar. Da diese Schriften zu den ebenfalls von Tolkien geschaffenen Sprachen gehören, ist es im Prinzip nicht möglich, normalen deutschen Text mit ihnen zu setzen. Als Beispiel für Tengwar soll hier ein Auszug aus dem Abschiedslied Galadriels dienen:



Kapitel 4

Erstellen von Dokumenten

Jedes längere Dokument ist in der Regel in verschiedene Kapitel, Abschnitte, usw. gegliedert. Es hat eventuell eine Zusammenfassung („abstract"), ein Inhaltsverzeichnis, eine Literaturliste oder eine Titelseite. LaTeX stellt dem Autor eine Reihe von Befehlen zur Verfügung, um die verschiedenen Aufgaben schnell und einfach zu lösen. Um diese Befehle richtig anwenden zu können, muß sich der Autor jedoch *vorher* über die logische Struktur seines Dokumentes klar werden. Dies entspricht vollkommen der Philosophie von TeX und LaTeX, da die Überlegung der Gliederung eine *inhaltliche* Arbeit ist. Die Aufgabe von LaTeX ist es, dem Autor möglichst viele Fragen zum Layout abzunehmen und ihm zu helfen, sich auf den Inhalt zu konzentrieren.

4.1 Kapitel und Überschriften

Damit LaTeX einen Text setzen kann, muß man unter anderem die Gliederung beschreiben. Dies geschieht durch Befehle wie z. B. `\section{...}`. Die Befehle stehen in einer bestimmten Hierarchie, die beachtet werden muß. Diese Ordnung ist in Abb. 4.1 dargestellt. Wie man sieht, können Artikel recht einfach als Kapitel in ein Buch eingebaut werden. Die Abstände zwischen den Kapiteln, die Numerierung und die Schriftgröße der Überschrift werden von LaTeX automatisch bestimmt. Sämtlichen Gliederungsbefehlen kann man als optionales Argument eine Kurzform der jeweiligen Überschrift angeben. Diese wird dann für das Inhaltverzeichnis und den Seitenkopf verwendet[1].

Die allgemeine Syntax der Gliederungsbefehle lautet also:

[1] Möchte man in einer Überschrift eine Fußnote verwenden, so sollte man das optionale Argument ebenfalls benutzen und die Fußnote nur im zweiten Ausdruck verwenden. Andernfalls würde die Fußnotenmarkierung auch ins Inhaltsverzeichnis übernommen werden, wo sie nur stört. Beispiel: `\section[`*Überschrift*`]{`*Überschrift*`\footnote{`*Fußnotentext*`}}`

```
\Gliederungsbefehl[Kurzform]{Überschrift}
\Gliederungsbefehl*{Überschrift}
```

Möchte man die Überschriften nicht numerieren, so benutzt man dazu die Befehle der Form `\Gliederungsbefehl{...}`. Hierbei wird auch der interne Zähler für die Gliederungsebenen nicht verändert, was eventuell zu Problemen beim Mischen von `*`-Formen mit den Standardformen führen kann. Bei den Befehlen mit einem `*` erfolgt auch kein Eintrag ins Inhaltsverzeichnis oder er wird auch nicht für den Seitenkopf verwendet.

Bei `book` und `report`:

`\chapter` → `\section` → `\subsection`

Bei `article`:

`\section` → `\subsection` → `\subsubsection`

Abbildung 4.1: Gliederungsebenen der Dokumentenklassen

Normal werden nur drei Ebenen durchnumeriert, also bis `\subsection` bei `book` und `report` und `\subsubsection` bei `article`. Die Numerierungstiefe kann jedoch mit dem Befehl

```
\setcounter{secnumdepth}{Tiefe}
```

geändert werden. Dabei kann *Tiefe* bei `article` einen der Werte 0,1,..., 4 annehmen, bei `report` und `book` -1, 0, 1,..., 4. Die Zuordnung der Zahlen ist in Tabelle 4.1 zu erkennen.

-1	`chapter`	2	`subsubsection`
0	`section`	3	`paragraph`
1	`subsection`	4	`subparagraph`

Tabelle 4.1: Liste der Gliederungsbefehle

Zusätzlich zu den oben genannten Gliederungsbefehlen gibt es in allen drei Dokumentenklassen den Befehl

```
\part{...}
```

Er stellt die oberste Gliederungsebene dar. Der Befehl hat jedoch keinen Einfluß auf die nachfolgende Gliederung oder Numerierung.

4.2 Titelseite

Eine Titelseite für das Dokument erzeugt man mit der Umgebung

```
\begin{titlepage}
        Text der Titelseite
\end{titlepage}
```

Bei der Gestaltung der Titelseite ist man völlig frei. Man kann sich jedoch auch an ein Standardformat von LaTeX halten. Hierzu dienen – ohne die Umgebung `titlepage` – die Befehle

```
\title{Titel}
\author{Autorenname}
\date{Datum}
```

Die beiden ersten Befehle sind zwingend notwendig; fehlt der letzte, so wird automatisch das aktuelle Datum eingesetzt. Die einzelnen Angaben werden horizontal zentriert übereinander gesetzt (vgl. Abbildung 4.2 auf der nächsten Seite). Zu lange Zeilen werden automatisch umgebrochen, man kann jedoch auch mit `\\` einen Zeilenumbruch erzwingen. Hat ein Dokument mehrere Autoren, so kann man die Einträge mit dem Befehl `\and` trennen:

```
\author{Autor 1 \\ Adresse 1 \and Autor 2 \\ Adresse 2}
```

Die einzelnen Autorenangaben werden jeweils für sich horizontal zentriert untereinander gesetzt und die beiden Blöcke nebeneinander ausgegeben. Mit dem Befehl

```
\thanks{...}
```

kann man auf der Titelseite Fußnoten erreichen. Im Gegensatz zu normalen Fußnoten wird auf der Titelseite jedoch keine horizontale Linie gezogen.

Möchte man einzelne Angaben nicht ausgeben, so kann man ganz einfach eine Leerform (z. B. `\author{}`) verwenden. Die Ausgabe des Titels erfolgt mit dem Befehl

```
\maketitle
```

Bei den Dokumentenklassen `report` und `book` wird eine eigene Titelseite erzeugt. Diese erhält keine Seitennummer und die nachfolgende Seite bekommt die Nummer 1. Bei `article` wird der Text des Titels oben auf der ersten Seite ausgegeben. Mit der Option `titlepage` kann man jedoch auch bei dieser Klasse einen eigene Titelseite erreichen.

Wie schreibe ich ein Buch?

P. J. Schreiberling

9. Januar 2000

```
\title{
  Wie schreibe ich ein Buch?
  }
\author{
  P.\,J.\,Schreiberling
  }
\maketitle
```

Abbildung 4.2: Erstellen eines Titelblattes

4.3 Verzeichnisse

4.3.1 Inhaltsverzeichnis

LaTeX kann automatisch für ein ganzes Dokument verschiedene Verzeichnisse anlegen. So kann man zum Beispiel mit dem Befehl

```
\tableofcontents
```

die Ausgabe eines Inhaltsverzeichnisses erreichen. In dieses Verzeichnis werden die jeweiligen Überschriften bzw. ihre Kurzform (siehe 4.1) mit den zugehörigen Seitennummern aufgenommen. Die Einträge im Inhaltverzeichnis werden so gesetzt, daß die logische Struktur des Dokumentes leicht zu erkennen ist. Die Gliederungsebene, bis zu der die Überschriften ins Inhaltverzeichnis übernommen werden, entspricht standardmäßig der Tiefe, bis zu der die Überschriften durchnumeriert werden, also `\subsection` für `book` und `report` und `\subsubsection` für `article`. Will man die Tiefe des Eintrags ändern, ohne die Numerierungstiefe zu ändern, so muß man den Befehl

```
\setcounter{tocdepth}{Tiefe}
```

verwenden. Der Befehl wird analog zur Festlegung der Numerierungstiefe der Gliederungen benutzt.

Gliederungsbefehle in der `*`-Form werden nicht automatisch ins Inhaltverzeichnis aufgenommen. Möchte man diese dennoch eintragen, so erreicht man dies mit dem Befehl

```
\addcontentsline{toc}{Format}{Überschrift}
```

Dabei bezeichnet ***Format*** die Formatierung des Eintrags. Gibt man hierfür z. B. `chapter` an, so wird der Eintrag so formatiert wie eine normale Kapitelüberschrift. Der Befehl sollte immer unmittelbar *nach* dem Gliederungsbefehl erscheinen, um sicherzustellen, daß die richtige Seitenzahl ins Inhaltsverzeichnis eingetragen wird.

Das Inhaltsverzeichnis erscheint im Dokument an der Stelle, an der der Befehl `\tableofcontents` steht. Dies ist meistens am Anfang des Dokumentes. Zu dem Zeitpunkt, an dem das Inhaltsverzeichnis gesetzt wird, ist jedoch noch nicht bekannt, auf welcher Seite welche Überschrift steht. Diese Information besteht erst, wenn das Dokument vollständig formatiert worden ist. Um diesen Widerspruch zu lösen, schreibt LaTeX während des Formatierens die Informationen über die Gliederung in eine Datei[2] mit der Endung `.toc` (siehe Abb. 2.1). Beim nächsten Durchlauf wird nun die erzeugte Datei mit dem Namen ***Dokumentenname***`.toc` abgearbeitet und das Inhaltsverzeichnis gesetzt. Das hat zur Folge, daß das ausgegebene Inhaltsverzeichnis sich immer auf die jeweils vorherige Version bezieht. Daher ist es angebracht, die endgültige Version zweimal formatieren zu lassen. LaTeX läßt beim Setzen des Dokumentes Platz für das Inhaltsverzeichnis. Ist dieses jedoch länger als dieser Freiraum, so verschieben sich beim zweiten Lauf die Seitennummern erneut und ein drittes Formatieren wird nötig. Vor dem letzten Ausdruck sollte man daher die Verzeichnisse kontrollieren.

4.3.2 Abbildungs- und Tabellenverzeichnis

Mit LaTeX kann man außerdem auch automatisch ein Tabellen- und ein Abbildungsverzeichnis erstellen. Die Handhabung ist analog zum Inhaltsverzeichnis. Die Ausgabe erreicht man mit den Befehlen

```
\listoffigures
\listoftables
```

[2] Die entsprechenden Dateien werden nur erzeugt, wenn das Inhaltsverzeichnis ausgegeben werden soll. Läßt man die Überschriften nur durchnumerieren, ohne ein Verzeichnis anzulegen, so wird die Datei auch nicht angelegt.

Ein Eintrag in die entsprechenden Verzeichnisse erfolgt mit dem Befehl

```
\caption[Kurzform]{Titel}
```

in den Umgebungen `figure` und `table` (siehe 5.1 und 5.2). Genauso wie beim Inhaltsverzeichnis werden auch hier die Informationen während des Setzens in zusätzlichen Dateien gespeichert. Sie bekommen die Endung `.lof` für das Abbildungsverzeichnis und `.lot` für das Tabellenverzeichnis. Eine zusätzliche Eintragung ist ebenso möglich:

```
\addcontentsline{lof}{figure}{Eintrag}
\addcontentsline{lot}{table}{Eintrag}
```

4.3.3 Literaturverzeichnis

Normalerweise enthält eine wissenschaftliche Arbeit ein Literaturverzeichnis. In diesem steht die verwendete Literatur, auf die im eigentlichen Text verwiesen wird. Häufig steht dabei bei Beginn der Arbeit an dem Dokument noch nicht endgültig fest, welche Einträge in dem Literaturverzeichnis gemacht werden müssen. Andererseits ist es sehr mühsam, wenn bei jeder Änderung des Literaturverzeichnisses das gesamte Dokument nach entsprechenden Verweisen durchsucht werden müßte. LaTeX bietet hierzu dem Nutzer verschiedene Instrumente. Die Umgebung

```
\begin{thebibliography}{Mustermarke}
    Bibliographie
\end{thebibliography}
```

erzeugt ein Literaturverzeichnis. Mit *Mustermarke* wird festgelegt, wieviel Platz für die Markierungen gelassen werden soll. Gibt man für diesen Parameter z. B. `{99}` ein, so läßt LaTeX einen zweistelligen Platz für die Markierungen frei. Längere Markierungen ragen nach links über den Rand hinaus. Die einzelnen Eintragungen für das Verzeichnis beginnen jeweils mit dem Befehl

```
\bibitem[Marke]{Schlüssel} Eintragstext
```

Wird der optionale Parameter *Marke* weggelassen, so werden die einzelnen Einträge fortlaufend durchnumeriert. Für *Marke* ist jeder beliebige Eintrag erlaubt. Der Parameter *Schlüssel* ist zwingend notwendig. Er legt ein Schlüsselwort fest, mit dem im Text auf den Eintrag verwiesen wird. Dieser Schlüssel erscheint weder im Text noch im Literaturverzeichnis. Er muß wie alle Schlüssel eindeutig sein.

Der dem Befehl `\bibitem` folgende Text enthält den eigentlichen Eintrag ins Literaturverzeichnis. Der Text erscheint dabei, ähnlich wie bei der `description`-Umgebung horizontal eingerückt. Die einzelnen `\bibitem`-Befehle werden in der Reihenfolge ihres Auftretens gesetzt.

Im Haupttext wird auf die Eintragungen mit dem Befehl

`\cite[`*`Zusatz`*`]{`*`Schlüssel`*`}`

verwiesen. ***Schlüssel*** ist dabei das im Literaturverzeichnis festgelegte Schlüsselwort des Eintrags. ***Zusatz*** ist ein optionaler Parameter, der eventuellen Text *nach* dem Verweis ausgibt, so zum Beispiel die Seite eines Zitats. An der Stelle des `\cite`-Befehls erscheint die im Literaturverzeichnis festgelegte Marke bzw. die fortlaufende Nummer des Eintrags, eingeschlossen in eckigen Klammern. Mit Angabe des Parameters ***Marke*** wird also auch der im Text erscheinende Verweis festgelegt.

Die `thebibliography`-Umgebung wird in aller Regel am Ende des Dokumentes ausgegeben. Die Überschrift über das Literaturverzeichnis erscheint immer in Form der obersten Gliederungsebene, also `chapter` bei `book` und `report`, bzw. `section` bei `article`. Der Text der Überschrift lautet in den ersten beiden Fällen „Literaturverzeichnis" und im letzten Fall „Literatur".

Verwendung von BibTeX

Steht das Programmpaket BibTeX zur Verfügung, so kann man sich das Erstellen eines Literaturverzeichnisses erheblich vereinfachen. In diesem Fall werden die Einträge in eine externen Datei geschrieben, die selbstverständlich von beliebig vielen Dokumenten benutzt werden kann. Im Hauptdokument wird nur noch der Befehl `\cite` verwendet und die externe Datei mit dem Befehl

`\bibliography{`*`Dateiname`*`}`

angegeben. Das Literaturverzeichnis wird dann automatisch erstellt (siehe auch Abb. 2.1 auf Seite 6). Die Einträge werden automatisch sortiert. Der angegebene Dateiname erhält *keine* Endung.

In vielen Fällen ist die genaue Art der Formatierung des Literaturverzeichnisses vorgeschrieben. Erstellt man sein Verzeichnis mit der `thebibliography`-Umgebung, so muß man diese Formatierungen von Hand durchführen. Bei der Verwendung von BibTeX übernimmt LaTeX dieses. Hierzu muß jedoch der Stil der Formatierung mit dem Befehl

`\bibliographystyle{`*`Stildatei`*`}`

festgelegt werden. ***Stildatei*** gibt dabei eine `bst`-Datei an, in der die Formatierung festgelegt ist. Bei einer Standartinstallation sind bereits zahlreiche dieser Stildateien vorhanden. Bei Bedarf können diese den eigenen Wünschen angepaßt werden.

BibTeX unterscheidet zwischen verschiedenen Arten von Einträgen (z. B. einem Buch, einem Artikel in einer Zeitschrift, ...) im Literaturverzeichnis. Diese Information wird beim Setzen des Literaturverzeichnisses berücksichtigt. An dieser Stelle soll nicht weiter auf die genaue Verwendung von BibTeX eingegangen werden. Ausführlichere Beschreibungen finden sich z. B. in dem Buch „Der LaTeX Begleiter" (Goossens et al., 1994).

Bei der Verwendung von BibTeX werden die Einträge, auf die innerhalb des Textes verwiesen wird, automatisch in das Literaturverzeichnis aufgenommen. Sollen noch weitere Einträge aufgenommen werden, so kann dies mit dem Befehl

```
\nocite{Schlüssel}
```

erreicht werden. In diesem Fall wird ein Eintrag in das Literaturverzeichnis aufgenommen, ohne daß im Text darauf verwiesen wird. Der Befehl darf an beliebiger Stelle im Textteil verwendet werden.

4.4 Verweise

Besonders in längeren Texten wird häufig auf verschiedene Teile innerhalb des Textes Bezug genommen. Wollte man diese Verweise jeweils von Hand setzen, so wäre dies äußerst mühsam. Bei bereits geschriebenem Text ist es sicher noch machbar, möchte man jedoch auf Textteile verweisen, die erst später geschrieben werden, so sind die Seitenzahlen und die Überschriften noch nicht bekannt. Auch kann es durchaus vorkommen, daß während der oft umfangreichen Arbeiten an größeren Dokumenten die Gliederung verändert, neue Teile eingebaut oder mache Passagen gestrichen werden. Von Hand können solche Verweise also erst ganz am Schluß der Arbeit gesetzt werden. Bei LaTeX kann man sich die Arbeit dagegen vollkommen abnehmen lassen.

Möchte man auf einen bestimmten Teil des Textes verweisen, so muß man zunächst mit dem Befehl

```
\label{Marke}
```

eine *unsichtbare* Markierung an der entsprechenden Stelle anbringen. Die Markierung kann eine beliebige Zeichenfolge[3] sein, Groß- und Kleinschreibung muß

[3] Es dürfen natürlich *keine* anderen Befehle oder Sonderzeichen verwendet werden.

beachtet werden. Die Marken müssen selbstverständlich eindeutig sein. Im Text wird nun mit den Befehlen

```
\ref{Marke}
\pageref{Marke}
```

auf die markierte Stelle verwiesen. Der untere Befehl gibt nur die Seite, auf der die Marke steht, aus. Der obere Befehl gibt die zu der Marke gehörende Gliederungsnummer aus, sofern diese existiert. Bei Überschriften ist diese Nummer immer die unterste Gliederungsebene, in der der Befehl `\label` steht. Um hierbei Fehler zu vermeiden, ist es sinnvoll, die Markierungen immer direkt nach dem Gliederungsbefehl anzubringen. Der Befehl darf jedoch nicht innerhalb des Gliederungsbefehles stehen. Abb. 4.3 zeigt die richtige und falsche Verwendung des Befehls `\label`.

4 Analyseverfahren *Text ... Text ... Text* **7 Resumee** Wie unter Punkt 4 bereits gesehen, ...	`\section{Analyseverfahren}` `\label{analyse}` `\emph{Text \dots Text \dots Text}` `\section{Resumee}` `Wie unter Punkt \ref{analyse}` `bereits gesehen, ...`
Richtig: `\section{Fahrtechnik}\label{fahr}` `\begin{table}[h]` `  \begin{center}` *Inhalt der Tabelle* `    \caption{...} \label{...}` `  \end{center}` `\end{table}`	Falsch: `\section{Fahrtechnik\label{fahr}}` `\begin{table}[h]` `  \begin{center}` *Inhalt der Tabelle* `    \caption{...}` `  \end{center}` `\end{table} \label{...}`

Abbildung 4.3: Richtige und falsche Verwendung des Befehls `\label`

Verweise sind bei allen Einheiten, die von LaTeX gezählt werden, möglich. Man kann also auch auf Abbildungen, Tabellen oder Einträgen in Listen (Aufzählungen) verweisen. Soll z. B. auf eine Abbildung verwiesen werden, so muß der Befehl `\label` *innerhalb* der entsprechenden Umgebung auftreten. Andernfalls wird auf die gerade aktuelle Gliederungsebene des Textes verwiesen. Bei Gleitobjekten (vgl. Abschnitt 5) muß der Befehl nach dem Befehl `\caption` stehen, da erst mit diesem Befehl der Zähler erhöht wird.

4.4.1 Flexible Verweise mit `varioref`

Besonders bei umfangreichen Texten mit vielen Verweisen möchte man oft eine Kombination von `\ref` und `\pageref` verwenden. Man könnte sich nun einen neuen Befehl definieren wie z. B.

```
\newcommand\fref[1]{\ref{#1} auf Seite~\pageref{#1}}
```

Es kann jedoch auch vorkommen, daß das Objekt, auf das verwiesen wird, auf der gleichen Seite steht. In diesem Fall wäre ein Verweis wie oben eher störend. Hier hilft das Paket `varioref`, von Frank Mittelbach (Mittelbach, 1998c), das den Befehl

`\vref{`*Marke*`}`

bereitstellt. Der Befehl schaut zunächst, ob die aufgerufene Marke auf der gleichen Seite liegt. Ist dies der Fall, so erzeugt `\vref` nur einen `\ref`-Befehl. Wenn die Marke auf der folgenden oder vorherigen Seite liegt, so erzeugt `\vref` zusätzlich den Text[4] „auf der vorherigen Seite“ oder „auf der nächsten Seite“, bei doppelseitigem Ausdruck stattdessen „auf der gegenüberliegenden Seite“. Ist der Abstand zwischen Marke und Verweis größer als eine Seite, so erscheint der Ausdruck „auf Seite“. Die hier angegeben Variationen funktionieren jedoch nur, wenn die Seitenzählung mit arabischen Ziffern erfolgt. Andernfalls kann der Abstand zwischen den Seiten nicht bestimmt werden.

Außer dem Befehl `\vref` bietet das Paket noch den Befehl

`\vpageref[`*Diese Seite*`][`*Andere Seite*`]{`*Marke*`}`

Liegt der Verweis auf der gleichen Seite wie die Marke, so wird der Text, der unter `\reftextcurrent` definiert ist[5] ausgegeben. Gibt man die beiden optionalen Parameter an, so kann man die Textausgabe noch genauer steuern. Mit ***Diese Seite*** wird der Text festgelegt, der erscheinen soll, wenn Marke und Bezug auf der gleichen Seite liegt, ***Andere Seite*** wird verwendet, wenn beide auf verschiedenen Seiten liegen.

[4] Standardmäßig erzeugt `varioref` englischen Text. Mit der Option `german` kann man jedoch auch deutschen Text einstellen. Alternativ kann man in der Paketdatei in der Zeile `\ExecuteOptions{german}` eintragen.

[5] Wird standardmäßig von `varioref` auf „auf dieser Seite“ gesetzt, kann aber bei Bedarf neu definiert werden.

4.4.2 Marken anzeigen mit `showlabels`

Bearbeitet man größere Dokumente, die eventuell auch noch auf verschiedene Dateien verteilt sind, ist es manchmal schwer, den Überblick über die gesetzten Marken zu behalten. Hier hilft das `showlabels` Paket. Wird es in der Präambel des Dokumentes durch die Anweisung `\usepackage{showlabels}` geladen, so erscheinen automatisch neben den Stellen, an denen mit dem `\label` Befehl Marken erzeugt werden, der Text der Marke. Man kann nun recht einfach feststellen, wo welche Marke gesetzt wurde.

Um die Anzeige auszuschalten, muß man einfach das Laden des Paketes aus der Präambel wieder streichen oder auskommentieren.

4.5 Fußnoten und Randnotizen

4.5.1 Fußnoten ...

Fußnoten[6] werden automatisch kapitelweise numeriert und am unteren Ende der Seite ausgedruckt. Oberhalb der Fußnoten erscheint eine dünne horizontale Linie. Die Fußnoten selbst erscheinen ein wenig eingerückt in einer kleineren Schrift (`footnotesize`). Es ist auch möglich, auf Fußnoten zu verweisen. Will man diese beim Verweisen gesondert hervorheben, so kann man sich leicht einen eigenen Befehl definieren.

Hier steht normaler Text[a].
... die Fußnote (a) wird hier referenziert.

[a] Fußnote mit Marke

```
\newcommand{\ftnref}[1]{~(\ref{#1})}
Hier steht normaler Text\footnote{
Fußnote mit Marke \label{ftnote}}.
... die Fußnote\ftnref{ftnote}
wird hier referenziert.
```

4.5.2 ... in Boxen

In bestimmten Modi sind Fußnoten nicht erlaubt, so z. B. innerhalb von LR-Boxen (siehe 4.7.1 auf Seite 65), Tabellen und mathematischen Formeln. Möchte man in diesen Bereichen trotzdem eine Fußnote haben, so muß man zunächst mit dem Befehl

`\footnotemark[`*Nummer*`]`

[6] Das ist eine Fußnote

die Markierung der Fußnote setzen. Der optionale Parameter *Nummer* legt die Nummer der Fußnote fest, wird er weggelassen, so wird der augenblickliche Zähler verwendet und um 1 erhöht. Außerhalb der verbotenen Modi wird nun mit

```
\footnotetext[Nummer]{Fußnotentext}
```

der eigentliche Text der Fußnote festgelegt. Hat man bei dem ersten Befehl den Parameter *Nummer* weggelassen, so muß dies auch beim zweiten Befehl passieren. Wenn man jedoch mehrere `\footnotemark` Befehle hintereinander verwendet hat, so ist der interne Zähler zu weit gestellt. In diesem Fall muß zunächst der Zähler wieder auf den Stand des ersten `\footnotemark` Befehls zurückgesetzt werden. Dies geschieht mit

```
\addtocounter{footnote}{Differenz}
```

Differenz ist dabei eine negative Zahl, die angibt, um wieviele Stellen der Zähler zurückgesetzt werden soll. Vor jedem weiteren `\footnotetext` Befehl muß der Zähler dann wieder um 1 erhöht werden. Dies kann entweder mit `\addtocounter` geschehen, oder mit dem Befehl

```
\stepcounter{footnote}
```

Hier ist ein $\boxed{\text{Text[7] mit mehreren[8]}}$ Fußnoten in einer Rahmenbox. Der Text der Fußnoten muß außerhalb der `fbox` angegeben werden. Bei mehreren Fußnoten muß dazu der Zähler zunächst wieder entsprechend zurückgesetzt werden.

[7] Erste Fußnote
[8] Zweite Fußnote

```
Hier ist ein \fbox{
Text\footnotemark\ mit
mehreren\footnotemark} Fußnoten ...
... zurückgesetzt werden.
\addtocounter{footnote}{-1}
\footnotetext{Erste Fußnote}
\stepcounter{footnote}
\footnotetext{Zweite Fußnote}
```

4.5.3 ... in Miniseiten

Im Gegensatz zu Boxen, dürfen innerhalb einer `minipage`-Umgebung (siehe Abschnitt 4.7.2 auf Seite 67) Fußnoten wie gewohnt verwendet werden. Die Fußnoten werden dabei innerhalb der Minipage und nicht am unteren Ende der eigentlichen Seite ausgegeben. Dabei werden die Fußnoten unmittelbar mit dem *nächsten* Befehl `\end{minipage}` gesetzt. Dies geschieht unabhängig von einer eventuellen Verschachtelung von `minipage`-Umgebungen.

Mit Hilfe einer Minipage kann man auch in einer Tabelle Fußnoten einsetzen, ohne die Befehle `\footnotemark` und `\footnotetext` verwenden zu müssen. Die Fußnoten werden dabei unmittelbar unter der Tabelle ausgegeben, wenn diese gleichzeitig mit der Minipage endet. Bei diesem Vorgehen ist außerdem gewährleistet, daß die Fußnoten in der Tabelle automatisch unabhängig von den Fußnoten im übrigen Text numeriert werden.

Die Kennzeichnung von Fußnoten erfolgt innerhalb einer Minipage mit hochgestellten Kleinbuchstaben[a]. Dabei wird mit jeder `minipage` ein neuer Zähler erzeugt. Er hat den Namen `mpfootnote`.

[a] Mehr als 26 Fußnoten wird man wohl selten in einer Minipage haben wollen ...

```
\begin{minipage}{70mm}
Die Kennzeichnung von Fußnoten erfolgt
innerhalb einer Minipage mit
hochgestellten
Kleinbuchstaben\footnote{Mehr als 26
Fußnoten wird man ...}
... den Namen \texttt{mpfootnote}.
\end{minipage}
```

4.5.4 `ftnright` und `footnpag` Paket

Setzt man Text in mehreren Spalten, so ist es oft wünschenswert, daß die Fußnoten am unteren Rand der rechten Spalten und nicht am Fuß jeder Spalte erscheinen. Dies ermöglicht das Paket `ftnright`. Das Aussehen kann man in Abbildung 4.4 auf der nächsten Seite erkennen. Es zeigt eine Seite der Paketokumentation von Frank Mittelbach (MITTELBACH, 1998B).

Das Paket `footnpag` schließlich kann Fußnoten ähnlich einem Inhaltsverzeichnis auf einer extra Seite ausgeben.

4.5.5 Randnotizen

In manchen Texten ist es nötig, Notizen am Seitenrand zu setzen. Dies kann normaler Text oder auch Striche oder kleinere Graphiken sein. Diese sog. Marginalien erreicht man mit dem Befehl

`\marginpar{`***Randnotiz***`}`

Dabei erscheint *Randnotiz* am rechten Rand. Der Text beginnt in Höhe der Zeile, in der der Befehl auftritt. Das nebenstehende Beispiel wurde also erzeugt mit

Hier ist eine Randnotiz

2.1 Description of the new layout

The result of this effort is presented in this paper and the reader can judge for himself whether it was successful or not.[2] The main idea for this layout is to assemble the footnotes of all columns on a page and place them all together at the bottom of the right column. Allowing for enough space between footnotes and text, and in addition, setting the footnotes in smaller type[3] I decided that one could omit the footnote separator rule which is used in most publications prepared with TEX.[4] Furthermore, I decided to place the footnote markers[5] at the baseline instead of raising them as superscripts.[6]

All in all, I think this generates a neat layout, and surprisingly enough, the necessary changes to the LATEX output routine are nevertheless astonishingly simple.

2.2 The use of the package

This package might be used together with any other package for LATEX which does not change the three internals changed by **ftnright.sty**.[7] In most cases, it is best to use this package as the very last package in the preamble to make sure that its settings are not overwritten by other packages.

It is unfortunate that the current LATEX has nearly no provisions to make such changes without overwriting the internal routines. In the LATEX3 implementation, we will certainly add some hooks that will make such changes more easy.

3 The documentation driver

The first bit of code contains the documentation driver file for TEX, i.e., the file that will produce the documentation you are currently reading. It will be extracted from this file by the **docstrip** program. If you don't want to make any changes to the presentation you can alternatively process the **.dtx** file directly with LATEX 2_ε to obtain the documentation.

```
⟨*driver⟩
\documentclass[twocolumn]{ltxdoc}


\newcommand{\TUB}{{\sl TUGboat\/}}
\renewcommand\DescribeMacro[1]{\fbox
          {\PrintDescribeMacro{#1}}}
\renewcommand\DescribeEnv[1]{\fbox
          {\PrintDescribeEnv{#1}}}
\renewcommand\PrintMacroName[1]{}

\setlength{\parindent}{1em}
\setlength{\parskip}
          {2pt plus1pt minus1pt}
\setlength{\headsep}{20pt}
\setlength{\columnsep}{1.5pc}
\renewcommand{\bottomfraction}{.4}


\setlength{\textwidth}{\paperwidth}
\addtolength{\textwidth}{-2in}
\setlength{\oddsidemargin}{0pt}
\setlength{\evensidemargin}{0pt}

\usepackage{ftnright}


\flushbottom
\CodelineIndex
\RecordChanges      % produce history
\EnableCrossrefs

\setcounter{IndexColumns}{2}
\IndexPrologue{\section{Index}
 All numbers denote code lines where
```

2. Please note, that this option only changed the placement of footnotes. Since this article also makes use of the `doc` package [1], that assigns tiny numbers to code lines sprincled throughout the text, the resulting design is not perfect. This package is now a standard part of LATEX 2_ε.

3. The standard layout in *TUGboat* uses the same size for footnotes and text, giving the footnotes, in my opinion, much too much prominence.

4. People who prefer the rule can add it by redefining the command `\footnoterule` [3, p. 156]. Please, note, that this command should occupy no space, so that a negative space should be used to compensate for the width of the rule used.

5. The tiny numbers or symbols, e.g., the '5' in front of this footnote.

6. Of course, this is only done for the mark preceeding the footnote text and not the one used within the main text where a raised number or symbol set in smaller type will help to keep the flow of thoughts, uninterrupted.

7. These are the macros `\@startcolumn`, `\@makecol` and `\@outputdblcol` as we will see below. Of course, the package will take only effect with a document class using a twocolumn layout (like `ltugboat`) or when the user additionally specifies `twocolumn` as a document class option in the `\documentclass` command.

2

Abbildung 4.4: Beispiel für das ftnright Paket

```
Der Text \marginpar{\small Hier ist eine \\ Randnotiz}
beginnt in Höhe der Zeile, in der der Befehl auftritt.
```

Der Befehl `\marginpar` erzeugt eine Absatzbox (siehe Abschnitt 4.7.2 auf Seite 67) von knapp 2 cm Breite. Da es schwierig ist, einen so schmalen Text richtig umzubrechen, ist es sinnvoll, den Zeilenumbruch wie in dem Beispiel von Hand vorzunehmen. Für einzelne Zeichen kann man den Befehl aber problemlos benutzen. Auch Striche werden am Seitenrand oftmals verwendet, um bestimmte Teile des Textes hervorzuheben.

Randnotizen erscheinen normalerweise am rechten Rand des Textes. Bei der Option `twoside` erscheinen sie immer an der Außenseite. Setzt man den Text zweispaltig (Option `twocolumn`), so erscheinen die Markierungen bei der linken Spalte am linken Rand und bei der rechten Spalte am rechten Rand. Dies kann zu Problemen führen, da beim Schreiben des Textes ja nicht bekannt ist, ob die Randnotiz links oder rechts erscheint. Will man als Notiz z. B. einen Pfeil erzeugen, der auf den Text zeigt, so muß das Zeichen ja von der Seite abhängen. Zu diesem Zweck kann man dem Befehl ein optionales Argument übergeben: $\Longleftarrow$

```
\marginpar[Linke-Notiz]{Rechte-Notiz}
```

Dabei gibt ***Linke-Notiz*** den Text an, der verwendet wird, wenn die Randnotiz links vom Text erscheint. Der Pfeil wurde zum Beispiel erzeugt mit dem Befehl

```
\marginpar[\hfill$\Longrightarrow$]{$\Longleftarrow$}
```

Der Befehl `\hfill` war nötig, da der Text innerhalb der Randnotiz linksbündig geschrieben wird, der Pfeil jedoch direkt am Textrand erscheinen sollte.

Innerhalb einer Randnotiz kann kein Seitenumbruch stattfinden. Erscheint eine Randnotiz in der untersten Zeile, so ragt sie eventuell noch weiter nach unten. Sie müßte also in zwei `\marginpar`-Befehle aufgeteilt werden. Dies sollte man jedoch erst vornehmen, wenn das Dokument ganz fertig ist. Jede Änderung am Dokument kann diese Positionierung wieder kaputtmachen.

4.6 Umgebungen

Bei LaTeX stehen dem Anwender eine Reihe verschiedener *Umgebungen* zur Verfügung. Innerhalb dieser Umgebungen wird der Text entsprechend den Umgebungsparametern gesetzt. Unter Punkt 3.2 auf Seite 22 wurden bereits einige dieser Umgebungen vorgestellt. Die allgemeine Form einer Umgebung lautet:

```
\begin{Name}
      Text
\end{Name}
```

Wie bereits gezeigt, können Umgebungen ineinander geschachtelt werden. Dabei ist jedoch die Reihenfolge der Befehle zu beachten:

```
\begin{äußere Umgebung}
     \begin{innere Umgebung}
     \end{innere Umgebung}
\end{äußere Umgebung}
```

Neben diesen Umgebungen gibt es auch noch die Möglichkeit, durch das Setzen von geschweiften Klammern kurze, „namenlose“ Umgebungen oder Gruppen zu schaffen. Dies dient z. B. zum kurzfristigen Umschalten der Schriftart. Alle Befehle, die nach einer öffnenden Klammer stehen, verlieren ihre Wirkung, wenn die Klammer geschlossen wird.

Außer den bereits vorgestellten Umgebungen zur Textausrichtung gibt es eine Reihe anderer, um z. B. Listen, Tabellen oder Abbildungen zu erzeugen.

4.6.1 Listen (`itemize`, `enumerate`, `description`)

Es gibt bei LaTeX 2_ε grundsätzlich drei verschiedene Möglichkeiten, eine Aufzählung zu erstellen. Die Umgebung `itemize` erstellt eine Liste (Abb. 4.6), in der die einzelnen Glieder alle mit einem gleichen Zeichen gekennzeichnet sind. Die Umgebung `enumerate` erzeugt eine numerierte Aufzählung (Abb. 4.7). Schließlich kann man die Umgebung `description` wählen. Mit ihr lassen sich Beschreibungen erstellen (Abb. 4.5).

Auto: Vierrädriges Fahrzeug, das in der Regel der Steigerung der männlichen Potenz dient.

Fahrrad: Zweirädriges Fahrzeug, das eine höhere Intelligenz des Nutzers beweist.

```
\begin{description}
\item[Auto:] Vierrädriges Fahrzeug,
  das in der Regel der Steigerung
  der männlichen Potenz dient.
\item[Fahrrad:] Zweirädriges Fahrzeug,
  das eine höhere Intelligenz ...
\end{description}
```

Abbildung 4.5: Beispiel für `description`

- Bei `itemize` werden die einzelnen Listenelemente durch Punkte und andere Symbole gekennzeichnet.
- Wie alle Umgebungen, können Listen auch verschachtelt werden:
  - Die maximale Tiefe der Schachtelung ist 4.
  - Marke Einrücktiefe und Kennzeichnung der Elemente werden automatisch angepaßt.

```
\begin{itemize}
\item Bei \texttt{itemize} werden die
  einzelnen Listenelemente ...
\item Wie alle Umgebungen, können
  Listen auch verschachtelt werden:
  \begin{itemize}
    \item Die maximale Tiefe der
      Schachtelung ist 4.
    \item[Marke] Einrücktiefe
      und Kennzeichnung der  ...
    \end{itemize}
\end{itemize}
```

Abbildung 4.6: Beispiel für `itemize`

Abb. 4.7 zeigt, daß man Aufzählungen, Beschreibungen und Listen auch beliebig mischen kann. Selbstverständlich kann man die Form der Markierungen verändern. Zunächst kann man dem Befehl `\item` einen optionalen Parameter übergeben, der als Markierung ausgegeben wird. `\item[*]` führt zum Beispiel

1. Bei `enumerate` werden die verschiedenen Elemente mit
   (a) Ziffern oder
   (b) Buchstaben numeriert.
2. Die Numerierung erfolgt automatisch.
3. Listen können auch mit Aufzählungen verknüpft werden.
   - Die maximale Tiefe der Schachtelung ist 4.

```
\begin{enumerate}
\item Bei \texttt{enumerate} ...
  \begin{enumerate}
    \item Ziffern oder
    \item Buchstaben numeriert.
  \end{enumerate}
\item Die Numerierung ...
\item Listen können auch ...
  \begin{itemize}
    \item Die maximale ...
  \end{itemize}
\end{enumerate}
```

Abbildung 4.7: Beispiel für `enumerate`

dazu, daß alle Eintragungen mit einem Stern gekennzeichnet werden. Auf diese Weise kann man natürlich jeden Eintrag mit einer anderen Marke versehen.

Dadurch ähnelt die `itemize`-Umgebung sehr der `description`-Umgebung. Der Unterschied zwischen beiden liegt darin, daß bei `itemize` eine vorher definierte Breite für die Markierung freigehalten wird. Ist die angegebene Marke zu breit, so ragt sie nach *links* über den Rand hinaus (siehe Abb. 4.6). Bei `description` dagegen reicht die Marke so weit nach rechts wie nötig. Die Einrücktiefe der nachfolgenden Zeilen bleibt davon unbeeinflußt.

Man kann die Änderung natürlich auch allgemeiner darstellen. LaTeX ruft intern die acht Befehle

```
\labelitemi, \labelitemii, \labelitemiii, \labelitemiv
\labelenumi, \labelenumii, \labelenumiii, \labelenumiv
```

auf. Die oberen Befehle beziehen sich auf eine `itemize`-Umgebung, die unteren auf eine `enumerate`-Umgebung. Die Endungen `i`, `ii`, `iii`, `iv` geben die Schachtelungstiefe an. Die Befehle können nun mit `\renewcommand` verändert werden. Möchte man z. B. die zweite Stufe einer `enumerate`-Umgebung mit römischen Ziffern und einem Punkt numerieren, so erreicht man dies durch

```
\renwecommand{\labelenumii}{\Roman{enumii}.}
```

Je nach Kontext kann es erwünscht sein, das Aussehen der Listenzähler zu verändern. Am einfachsten geschieht dies mit dem Zusatzpaket `enumerate`. Dieses Paket definiert die Umgebung `enumerate` um, so daß sie ein optionales Argument erhält, mit dem das Aussehen des Zählers verändert werden kann. Hierzu dienen die Zeichen `A`, `a`, `I`, `i` und `1`, die den Zählstilen `\Alpha`, `\alpha`, `\Roman`, `\roman` und `\arabic` entsprechen. Außer diesen Zeichen kann man praktisch jeden beliebigen Text angeben. Werden jedoch die oben aufgeführten Zeichen verwendet, ohne daß sie in der angegebenen Weise interpretiert werden sollen, so müssen sie in `{}` eingeschlossen werden.

B1 Listentext Listentext

B2 Listentext Listentext

 (2-a) weiterer Text

 (2-b) (siehe 1)

```
\begin{enumerate}[B1]
\item Listentext Listentext \label{a}
\item Listentext Listentext
  \begin{enumerate}[(2-a)]
  \item weiterer Text
  \item (siehe \ref{a})
  \end{enumerate}
\end{enumerate}
```

Verweise sind selbstverständlich auch möglich. Allerdings wird dabei, wie das Beispiel zeigt, nicht die zusätzliche Formatierung der Zähler beachtet.

4.6.2 Regelsätze

Besonders in wissenschaftlichen Arbeiten werden häufig Regelsätze oder Thesen verwendet. Diese sollen in einem Dokument oft durchnumeriert werden. Der folgende Text wird dann in der Regel kursiv gesetzt. Dies alles kann man natürlich auch durch eine direkte Angabe des Schriftstiles und der Satznummer erreichen. Komfortabler geht es jedoch mit der `theorem`-Umgebung von LaTeX.

Zunächst wird in der Präambel mit dem Befehl

```
\newtheorem{Name}{Begriff}[Zusatz-Zähler]
```

eine neue Struktur definiert. ***Name*** ist dabei der Name, unter dem die Struktur später aufgerufen wird, ***Begriff*** legt den Text vor der Nummer fest. Mit ***Zusatz-Zähler*** kann eine Gliederungsebene angegeben werden, innerhalb der die Numerierung erfolgt.

Aufgerufen und fortlaufend numeriert wird die Struktur mit

```
\begin{Name}[Zusatz]
```

Zusatz gibt dabei einen Text an, der in Klammern hinter der Nummer des Satzes in Fettdruck ausgegeben wird. Beendet wird die Umgebung wie gewohnt mit

```
\end{Name}
```

Axiom 1 *Ein Körper verharrt im kräftefreien Raum im Zustand der Ruhe oder gradlinigen, gleichförmigen Bewegung.*

Formel 4.1 (Coriolisbeschleunigung)
$a_c = 2 * \omega_E * \sin(\phi) * v$

```
\newtheorem{formel}{Formel}[chapter]
\newtheorem{axiom}{Axiom}
\begin{axiom}
  Ein Körper verharrt ...
\end{axiom}
\begin{formel}[Coriolisbeschleunigung]
  $a_c = 2 * \omega_E * \sin(\phi) * v$
\end{formel}
```

Soll eine Struktur keinen eigenen Zähler erhalten, sondern mit einer anderen Struktur einen gemeinsamen Zähler verwenden, so kann dies durch den Befehl

```
\newtheorem{Name}[Nummer von]{Begriff}
```

erfolgen. *Nummer von* gibt dabei die Nummer der vorher bereits definierten Struktur an.

4.6.3 Tabulatoren (`tabbing`)

Mit einer Schreibmaschine kann man recht einfach Text mit Hilfe der Tabulatoren einrücken. Selbst einfache Tabellen kann man auf diese Weise erstellen. Bei LaTeX gibt es ebenfalls eine Tabulator-Funktion: die `tabbing`-Umgebung. Mit dem Befehl `\=` setzt man eine Tabulatorposition, `\>` springt zur nächsten Tabulatorposition, und `\\` trennt die Zeilen. Die Tabulatorposition wird dabei in der ersten Zeile gesetzt. Da jedoch in dieser Zeile nicht immer die längsten Spalteneinträge stehen, kann man mit dem `\kill` Befehl eine Musterzeile eingeben. Diese erscheint nicht im Text, LaTeX übernimmt aber die Spaltenbreiten aus dieser Zeile.

links	Mittelteil	rechts
Es		
war einmal	und ist	nicht mehr
ein		ausgestopfter
		Teddybär

```
\begin{tabbing}
war einmal\quad\=Mittelteil\quad\=\kill
links \> Mittelteil \> rechts\\
Es \\
war einmal \> und ist \> nicht mehr\\
ein  \>  \> ausgestopfter\\
  \>  \> Teddybär
\end{tabbing}
```

4.6.4 Direkte Ausgabe (`verbatim`, `verb`)

Der Text in der Umgebung `verbatim` wird direkt ausgegeben. Das bedeutet, er wird in seiner Erscheinung genauso gesetzt, wie dies in der Eingabedatei erfolgt ist. Alle Zeilenumbrüche, Leerzeichen und Spezialzeichen werden ohne Interpretation ausgegeben. Auf diese Weise wurden auch alle Textbeispiele in dieser Anleitung erstellt. Ist eine Zeile länger als die erlaubte Textlänge, so ragt die Zeile über den rechten Rand hinaus.

Soll nur ein kurzer Textteil auf diese Weise gesetzt werden, so kann man dies mit einem der Befehle

```
\verb|Text|
\verb*|Text|
```

erreichen. Statt des |-Strichs können auch Punkte oder Doppelpunkte benutzt werden. Allerdings müssen Anfang und Ende immer durch das gleiche Zeichen gekennzeichnet sein. Die untere Version gibt außerdem jedes Leerzeichen durch je ein ␣ aus.

Der `\dots`-Befehl …	`Der \verb|\dots|-Befehl \dots \\`
`\begin{document}`	`\verb:\begin{document}: \\`
zwei Leerstellen:␣␣	`zwei Leerstellen:\verb*|  |`

Weder die `verbatim`-Umgebung, noch der `\verb` Befehl dürfen innerhalb eines Parameters eines anderen Befehls auftreten. Außerdem dürfen sie nicht in einer `tabular`-Umgebung (siehe Abschnitt 7.1 auf Seite 99) stehen.

4.7 Boxen

Eine Box ist ein Textteil, den TeX als eine Einheit ansieht. Diese Box kann beliebig auf einer Seite verschoben werden. Da sie als eine Einheit angesehen wird, kann sie von TeX nicht umgebrochen werden. Dies gilt auch dann, wenn eine Box aus mehreren kleineren Boxen aufgebaut ist. LaTeX kennt drei verschiedene grundlegende Arten von Boxen: LR-Boxen, Absatzboxen und Rule-Boxen. Eine LR-Box ist eine Box, in der der Text von links nach rechts angeordnet wird. Daher kann eine LR-Box nur eine Zeile (oder eine oder mehrere weitere Boxen) enthalten. Eine Absatzbox dagegen kann Text in mehreren Zeilen enthalten. Eine Rulebox schließlich ist ein mit Farbe gefülltes Rechteck.

4.7.1 LR-Boxen

Für die Erzeugung von LR-Boxen stehen zwei verschiedene Befehlspaare zur Verfügung:

`\mbox{`*Text*`}`	`\makebox[`*Breite*`][`*Pos*`]{`*Text*`}`
`\fbox{`*Text*`}`	`\framebox[`*Breite*`][`*Pos*`]{`*Text*`}`

Die beiden linken Befehle erzeugen jeweils eine Box, deren Breite von dem in ihr enthaltenen Text bestimmt wird. Bei den beiden rechten wird die Breite als Argument übergeben. Der zweite Parameter (`[`*Pos*`]`) gibt die Positionierung des Textes innerhalb der Box an. Er darf die Werte `l` für linksbündigen Text und `r` für rechtsbündigen Text annehmen. Schließlich kann man noch `s` angeben. Dabei wird der Text soweit gedehnt, daß er den gesamten Raum der Box ausfüllt. Wird das Argument weggelassen oder `c` angegeben, so wird der

Text innerhalb der Box zentriert. Bei den beiden unteren Boxbefehlen (`\fbox` und `\framebox`) wird der Text mit einem Rahmen umgeben[9]. Einen engen Rahmen um den Text erhält man mit dem Befehl

`\frame{`*Text*`}`

Das oberste Beispiel in Abb. 4.8 zeigt, daß sich LaTeX nicht darum kümmert, ob der in einer `framebox` oder `makebox` enthaltene Text breiter als die Box ist. Dies macht bei einer Rahmenbox wenig Sinn, kann jedoch bei einer `makebox` eine große Hilfe sein.

rechtbündig	`\framebox[3mm][r]{rechtbündig} \\`
zentriert	`\framebox[3mm]{zentriert} \\`
linksbündig	`\framebox[3mm][l]{linksbündig} \\`
hier eine breite Box	`hier eine \makebox[20mm]{breite} Box \\`
und eine gedehnte Box	`\framebox[55mm][s] {und eine gedehnte Box} \\`
hier eine Rahmen- Box	`hier eine \framebox[20mm][l]{Rahmen-} Box`

Abbildung 4.8: LR-Boxen

Steht am Anfang einer Zeile z. B. der Befehl `\makebox[0mm][r]{Steht}`, so kann man den Inhalt über den Rand der Seite hinaus ragen lassen. Derartige Boxbefehle kann man auch sehr gut innerhalb der `picture`-Umgebung zur Positionierung von Texten verwenden.

Da innerhalb einer LR-Box kein Zeilenumbruch erfolgen kann, sind auch die Befehle `\newpage` und `\\` wirkungslos. Möchte man innerhalb einer solchen Box jedoch einen mehrzeiligen Text setzen, so kann man einfach eine `minipage`-Umgebung oder eine `parbox` verwenden.

Definieren von Boxen

Mit den Boxbefehlen ist es möglich, gleichartigen Text, der an verschiedenen Stellen im Dokument auftritt, in Boxen abzulegen und später aufzurufen.

[9] TeX bestimmt Breite und Höhe des Rahmens einer `fbox` nach deren Inhalt. Man kann damit auch größere Textteile z. B. eine Tabelle mit einem Rahmen umgeben. Will man jedoch einzelne Buchstaben einrahmen, so kann die Höhe unterschiedlich sein: $\boxed{\text{x}}$ $\boxed{\text{X}}$

```
\newsavebox{\BoxName}
```

richtet einen Speicher mit dem Namen *BoxName* ein, in dem dann LR-Boxen mit den Befehlen

```
\savebox{\BoxName}[Breite][Pos]{Text}
\sbox{\BoxName}{Text}
```

abgespeichert werden können. Der Zugriff auf die dort abgespeicherten Boxen erfolgt mit dem Befehl

```
\usebox{\BoxName}
```

4.7.2 Absatzboxen und „Miniseiten“

Möchte man mehrzeiligen Text in eine Box setzen, so dient hierzu entweder der Befehl

```
\parbox[Pos]{Breite}{Text}
```

oder die Umgebung

```
\begin{minipage}[Pos]{Breite}
\end{minipage}
```

Beide Möglichkeiten erzeugen eine *vertikale* Box der angegebenen Breite, innerhalb der beliebiger Text in Zeilen untereinander angeordnet wird. Zeilenumbrüche erfolgen automatisch oder mit den gewohnten Befehlen. Der optionale Parameter `[Pos]` bestimmt die Ausrichtung der Box zur laufenden Zeile. Mögliche Werte sind `t` und `b`. Mit `t` wird die oberste Zeile, mit `b` die unterste der Box zur laufenden Zeile ausgerichtet. Wird er weggelassen, so wird die Box vertikal zur Zeile zentriert.

Die Argumente entfalten ihre Wirkung nur innerhalb eines Absatzes oder einer Tabelle, da nur hier von einer laufenden Zeile gesprochen werden kann. Steht die Box zwischen zwei Absätzen, so hat der Parameter keinerlei Wirkung. Wie bei allen Boxen kann innerhalb einer Absatzbox oder `minipage`-Umgebung kein Seitenumbruch erfolgen.

4.7.3 `fancybox` Paket

Das `fancybox` Paket von Timothy van Zandt bietet eine Reihe weiterer Befehle für gerahmte Boxen und sogar die Möglichkeit einer Box-Umgebung.

Dies ist eine 50mm breite `parbox`. Sie ist vertikal zentriert zu der Umgebung.	UMGEBUNG	Diese Box wird mit ihrer obersten Zeile an der Umgebung ausgerichtet.

Das Beispiel wurde erzeugt mit den Befehlen:

```
\parbox{50mm}{Dies ist eine 50mm breite \texttt{parbox}. Sie ist
      vertikal zentriert zu der Umgebung.}
\hfill UMGEBUNG \hfill
\parbox[t]{50mm}{Diese Box wird mit ihrer obersten
      Zeile an der Umgebung ausgerichtet.}
```

Abbildung 4.9: Verwendung des `parbox`-Befehls

Die Boxbefehle

Insgesamt stehen vier verschiedene Boxarten zur Verfügung. Der Abstand zwischen Boxinhalt und Rahmen wird wie bei einer normalen `\fbox` durch den Wert `\fboxsep` festgelegt.

`\doublebox` Die Stärke der inneren Linie beträgt `0.75\fboxrule`, die der äußeren `1.5\fboxrule`. Der Abstand beider Linien beträgt `1.5\fboxrule plus 5pt`.

`\shadowbox` Die Dicke des Rahmens wird durch `\fboxrule` definiert. Die Breite des Schattens kann man mit `\shadowsize` ändern. Standardmäßig beträgt er 4pt.

`\ovalbox` Die Breite der Linie wird über `\thinlines` bestimmt. Den Radius der Eckbögen kann man mit dem Befehl

`\cornersize{`*`Faktor`*`}`

beeinflussen. Der Befehl multipliziert ***`Faktor`*** mit dem Minimum der Breite oder Höhe der Box. Die Voreinstellung ist `\cornersize{0.5}`.

`\Ovalbox` Diese Box ist ähnlich der `\ovalbox`. Allerdings wird die Stärke der Linie mit `\thicklines` bestimmt.

Box-Umgebungen

Das `fancybox` Paket definiert aber nicht nur Boxbefehle, man kann mit ihm auch Box-Umgebungen erzeugen. Die `Sbox` Umgebung funktioniert ähnlich wie der PlainTEX-Befehl `\sbox`. Der Inhalt der Umgebung wird dabei in einem Boxregister gespeichert, auf das mit dem Befehl `\TheSbox` zugegriffen werden kann. Auf diese Weise kann man zum Beispiel recht einfach Miniseiten rahmen.

Dies ist ein Beispiel für eine gerahmte Miniseite. Sie wurde mit dem `\Sbox`-Befehl erzeugt. Die Breite der Box ist dabei nicht in der Definition festgelegt. Ebenso kann der neuen Umgebung auch die Position als Parameter übergeben werden (siehe Abschnitt 4.7.2)

```
\newenvironment{Rahmenbox}
  {\begin{Sbox}\begin{minipage}}
  {\end{minipage}\end{Sbox}
    \fbox{\TheSbox}}
\begin{Rahmenbox}{60mm}
  Dies ist ein Beispiel für eine ...
\end{Rahmenbox}
```

Man könnte einen Rahmen um eine Miniseite auch einfach erzeugen, indem man die Miniseite in eine andere Box setzt. Es kann jedoch Situationen geben, in denen der Inhalt des Rahmens nicht einfach als Parameter einem Boxbefehl übergeben werden kann. In diesen Fällen ist eine Trennung der Anfangs- und Endanweisungen der Boxbefehle nötig.

Im `fancybox` Paket sind bereits einige neue Umgebungen vordefiniert:

- `Bcenter`, `Bflushleft`, `Bflushright` erzeugen eine `center`, `flushleft` oder `flushright` Umgebung in einer Box.
- `Bitemize`, `Benumerate`, `Bdescription` erzeugen Listen in einer Box.
- `Beqnarray` und `Beqnarray*` ähneln der `eqnarray` bzw `eqnarray*` Umgebung. Allerdings stehen die Formelnummer hier rechts.

Die Boxen, die mit diesen Umgebungen erzeugt werden, sind immer genau so breit wie die breiteste Zeile in ihnen. Ein automatischer Zeilenumbruch findet nicht statt. Sie sind vor allem dazu gedacht, innerhalb der oben vorgestellten Zierrahmen verwendet zu werden. Da die Rahmenboxen einfache LR-Boxen sind, kann man in ihnen nicht direkt mehrzeiligen Text eingeben. Dazu müßte zunächst eine `parbox` erzeugt werden. Diese jedoch kann ihre Breite nicht automatisch bestimmen.

Außer bei der Neudefinition von Umgebungen kann man den `\Sbox`-Befehl auch bei Tabellen einsetzen, in denen man den Inhalt einzelner Zellen einer Spalte in Boxen setzen möchte. Es ist sicherlich sehr aufwendig, in jeder Zeile den Text in eine `\framebox` zu setzen. Allerdings kann man mit den normalen Spaltendefinitionen keine Boxen übergeben. Mit Hilfe des `fancybox` Paketes ist dieses jedoch möglich (siehe auch Abschnitt 5.2 auf Seite 78).

1	Pflaumen
2	Birnen

```
\newenvironment{F}{\begin{Sbox}}
{\end{Sbox}\framebox[25mm]{\TheSbox}}
\begin{tabular}
    {l>{\begin{F}}c<{\end{F}}}
  ...
\end{tabular}
```

4.7.4 Rule-Boxen

$\boxed{\texttt{\textbackslash rule[}\textit{Lift}\texttt{]\{}\textit{Breite}\texttt{\}\{}\textit{Höhe}\texttt{\}}}$

erzeugt ein mit Farbe gefülltes Rechteck mit den angegebene Maßen. ***Lift*** gibt dabei wie bei `raisebox` die Verschiebung gegenüber der Grundlinie an. So erhält man mit dem Befehl `\rule[2pt]{5mm}{1.5pt}` z. B. ▬.

Mit einer Rule-Box lassen sich aber nicht nur vertikale oder horizontale Striche erzeugen, sondern auch unsichtbare Platzhalter oder Stützen. Die Werte für ***Lift***, ***Breite*** und ***Höhe*** dürfen auch Null betragen. Auf diese Weise kann man z.Ḃ vertikale Abstände zwischen Zeilen erzeugen.

4.7.5 Vertikale Verschiebung von Boxen

Man kann innerhalb eines Textes einzelne Teile relativ einfach vertikal verschieben. Hierzu dient der Befehl

Text der oberen Zeile
Text $^{\text{oben}}$ $_{\text{unten}}$ normal
Text der mittleren Zeile

Text $^{\text{oben}}$ $_{\text{unten}}$ normal

Text der unteren Zeile

```
Text der oberen Zeile
Text \raisebox{1ex}{oben}
  \raisebox{-1ex}{unten} normal \\
Text der mittleren Zeile \\
Text \raisebox{1ex}[5ex][3ex]{oben}
  \raisebox{-1ex}{unten} normal \\
Text der unteren Zeile
```

Abbildung 4.10: Verschiebung von Boxen

`\raisebox{`*Lift*`}[`*ober*`][`*unter*`]{`*Text*`}`

Der Befehl erzeugt eine rahmenlose LR-Box, die um *Lift* gegenüber der Grundlinie verschoben ist. Positive Werte verschieben den Text nach oben, negative nach unten. Die beiden optionalen Argumente geben an, daß der Text ohne Verschiebung um *ober* über und um *unter* unter die Grundlinie reicht. Werden die Angaben nicht gemacht, so geht LaTeX davon aus, daß der Text weder nach unten noch nach oben übersteht. Die Werte für *ober* und *unter* können völlig frei gewählt werden. LaTeX berücksichtigt sie bei der Bestimmung des Abstands zur vorhergehenden und nachfolgenden Zeile.

4.7.6 Verschachtelung von Boxen

Die verschiedenen Box-Befehle können beliebig verschachtelt werden. Hierzu sollte man sich vor Augen halten, daß LaTeX eine Box als ein Zeichen ansieht. Daher ist es ohne weiteres möglich, eine `parbox` in eine `fbox` zu setzen. Auf diese Weise erhält man einen Rahmen um die gesamte Box.

Dies hier ist eine 60mm breite Absatzbox, die in zwei Rahmenboxen (`\fbox`) gesetzt wurde. Die Box enthält außerdem eine `\makebox`, die 30mm breit ist.

```
\fbox{\fbox{\parbox{60mm}{
 Dies hier ist eine 60mm breite
 Absatzbox, die in zwei Rahmenboxen
 (\texttt{\fbox}) gesetzt wurde.
 ... }}}
```

Abbildung 4.11: Verschachtelung von Boxen

Als Text in einer Box können auch beliebige andere Befehle oder Umgebungen stehen. Man kann damit z. B. Bilder oder Tabellen rahmen und genau positionieren.

4.8 Mehrspaltiger Druck

4.8.1 `twocolumn` Option

LaTeX bietet die Möglichkeit, Text ein- oder zweispaltig zu setzen. Dies geschieht entweder mit der Option `twocolumn`, die für das gesamte Dokument zweispaltigen Text erzeugt, oder innerhalb eines Dokumentes mit dem Befehl

`\twocolumn[`*`Vortext`*`]`

Dieser Befehl beendet die laufende Seite und beginnt eine neue zweispaltige Seite. ***`Vortext`*** gibt den einen Text an, der am Anfang der Seite über beide Spalten gesetzt wird. Beenden kann man den zweispaltigen Druck mit dem Befehl

`\onecolumn`

Auch dieser Befehl beendet die laufende Seite und beginnt eine neue. Man kann also auf einer Seite nicht zweispaltigen und einspaltigen Text mischen. Außerdem ist es nicht möglich, mehr als zwei Spalten zu setzen.

4.8.2 `multicol` Paket

Möchte man mehr Spalten haben oder auf einer Seite die Spaltenanzahl wechseln können, so kann man dazu das `multicol` Paket von Frank Mittelbach verwenden (MITTELBACH, 1998A). Das Paket erzeugt die Umgebung **`multicols`**, die bis zu zehn Spalten auf einer Seite ermöglicht. Die Spalten werden dabei am Ende mit Ausgleich gesetzt, so daß alle Spalten in der Höhe angeglichen werden. Zwischen den Spalten kann eine vertikale Linie gezogen werden. Die Umgebung kann außerdem auch innerhalb einer anderen Umgebung verwendet werden.

Wurde das Paket in der Präambel geladen, so wird die Umgebung aufgerufen mit

`\begin{multicols}{`*`Spalten`*`}[`*`Vortext`*`][`*`Abstand`*`]`

Als zwingendes Argument erwartet die Umgebung die Anzahl der zu setzenden Spalten. Optional kann man noch einen Text angeben, der oberhalb der Spalten über die gesamte Seitenbreite gesetzt wird. Mit ***`Abstand`*** kann der Zwischenraum zur vorherigen Umgebung beeinflußt werden. Besonders wenn der ***`Vortext`*** zu lang ist, kann es sein, daß der global eingestellte Zwischenraum nicht ausreicht, oder ein Seitenumbruch an der falschen Stelle erfolgt.

4.8.3 Anpassen der `multicols` Umgebung

Das Aussehen von mehrspaltigen Umgebungen kann man mit einer Reihe verschiedener Parameter ändern. Eine Liste der Parameter mit ihren Voreingestellungen kann man in Tabelle 4.2 erkennen. Eingestellt werden die Werte grundsätzlich vor dem Aufruf der Umgebung.

Beim Aufruf der Umgebung wird zunächst geprüft, ob noch genügend Freiraum auf der Seite vorhanden ist. Hierzu dient der Wert `\premulticols` oder der optionale Parameter ***`Abstand`*** der `\egin`-Anweisung. Ist nicht genügend Freiraum vorhanden, wird ein neue Seite begonnen. Nach Ende der Umgebung wird der zusätzliche Leerraum `\multicolsep` eingesetzt.

Die Breite der Spalten wird aus den jeweils gültigen Werten von `\linewidth` und `\columnsep` und der Anzahl der Spalten berechnet. `\columnsep` gibt dabei den Raum zwischen zwei Spalten an. Die Stärke der vertikalen Linie zwischen den Spalten wird durch `\columnseprule` festgelegt. Beträgt dieser Wert 0pt, so wird die Linie unterdrückt. Ist die Linie dagegen breiter als der Zwischenraum, so überdruckt sie den Text der Spalten.

Tabelle 4.2: Parameter zur Anpassung der `multicols` Umgebung

`\premulticols`	50.0pt
`\multicolsep`	12.0pt plus 4.0pt minus 3.0pt
`\columnsep`	10.0pt
`\columnseprule`	0.0pt

4.8.4 Komplexer mehrspaltiger Druck

Möchte man ein komplexeres mehrspaltiges Layout, ähnlich dem von Zeitungen, realisieren, so stößt man bei alleiniger Verwendung des `multicol`-Paketes an Grenzen. Man kann zwar auf einer Seite mehrere `multicols` Umgebungen verwenden und diese sogar ineinander verschachteln. Diese Umgebungen können aber nicht um andere herumfließen.

Mit Hilfe der `picture`-Umgebung kann aber auch ein derartiges Layout realisiert werden. Ein Beispiel hierfür ist in Abbildung 4.12 auf Seite 75 zu sehen.

Für diese Seite wurde zunächst ein Bild erzeugt, das die gesamte Seite einnimmt. Innerhalb dieses Bildes wurden die einzelnen Artikel jeweils mit einem `\put`-Befehl an die entsprechende Stelle gesetzt. Um Fehler zu vermeiden und sich gleichzeitig die Arbeit zu erleichtern, wurden für die fünf Spalten jeweils ein Befehl definiert, der den Artikel direkt in die richtige Spalte[10] setzt. Nun

[10] Genaugenommen ist hier der Begriff „Spalte“ irreführend, da es sich nur um Bild handelt.

muß für jeden Artikel nur noch die Höhe, in der er beginnen soll, angegeben werden.

Um zum Beispiel einen Artikel in der zweiten Spalte beginnen zu lassen, wurde der folgende Befehl definiert:

```
\newcommand{\SpZwei}[3]{\put(38.5,#1){\parbox[t]{#2}{#3}}}
```

Der Befehl erhält drei Parameter. Mit dem ersten wird die Höhe der Grundlinie der ersten Zeile des Artikels festgelegt, der zweite Parameter gibt die Breite der Absatzbox an und der dritte enthält den eigentlichen Artikel.

Im Falle des Leitartikels in Abbildung 4.12 besteht dieser Inhalt zunächst aus den Titelzeilen. Ihnen folgt eine zweispaltige `multicols` Umgebung, an die sich wiederum eine dreispaltige anschließt.

Um tatsächlich den Eindruck entstehen zu lassen, als stünden die Artikel in zusammenhängenden Spalten, mußten verschiedene Abstände und Längen genau passend festgelegt werden.

```
\setlength{\textwidth}{190mm}
\setlength{\columnsep}{2.5mm}
\setlength{\parskip}{2pt plus 3pt minus 1pt}
\setlength{\parindent}{1.5em}
\newlength{\spalte}
\setlength{\spalte}{36mm}
\setlength{\multicolsep}{-1pt plus 1pt minus 1pt}
```

Schließlich wurden noch fünf Längenbefehle definiert, die die unterschiedliche Breite von Artikeln angeben. Im Falle des Leitartikels wurde also dem Befehl `\SpZwei` als zweites Argument die Länge `\drei` übergeben.

```
\newlength{\eins}
\newlength{\zwei}
\newlength{\drei}
\newlength{\vier}
\newlength{\fuenf}
\setlength{\eins}{36mm}
\setlength{\zwei}{74.5mm}
\setlength{\drei}{113mm}
\setlength{\vier}{151.5mm}
\setlength{\fuenf}{190mm}
```

november
1998 nr. 0

tübinger lichtblick

Neu in Tübingen

KIJUMFO — JUGENDZEITUNG FÜR TÜBINGEN — 0 DM – KOST NIX

Jugendgemeinderat in Tübingen

(hl) In Tübingen soll ein Jugendgemeinderat gegründet werden! Die genaue Form steht noch nicht fest.

Im nächsten Sommer wird es soweit sein: Die Tübinger Jugendlichen bekommen eine eigene politische Vertretung! Hierzu hat der zuständige Sozialausschuß am 24. September einen Grundsatzbeschluß gefaßt.

Wie er genau aussehen wird, steht noch nicht fest. Zur Zeit bestehen zwei Vorschläge, einer von der Verwaltung und einer, den das Kinder- und Jugend-Umweltforum (KIJUMFO) zusammen mit einer Gruppe von rund zwanzig Jugendlichen in den letzten Monaten erarbeitet hat.

In einer im Sommer durchgeführten Umfrage unter rund 1.800 Schülern sprach sich eine überwältigende Mehrheit für einen solchen Jugendgemeinderat aus. Näheres über den Stand der Vorbereitungen und über die Umfrage lest Ihr auf Seite 6.

Aktuell

Klage abgewiesen

Die Klage des 17jährigen Berliner Schülers Robert Rostoski, der an der Bundestagswahl teilnehmen wollte, wurde vom Berliner Verwaltungsgericht als nicht zulässig abgewiesen.

Kinderarbeit in Spanien

Spanische Gewerkschaften haben herausgefunden, daß im Südwesten Spaniens über 200 Kinder bei der Tomatenernte arbeiten. Die sieben- bis zehnjährigen hatten für einen sehr geringen Lohn 13 Stunden am Tag Kisten geschleppt.

Sex ab 13

In Österreich tritt das neue Sexualstrafrecht in Kraft. Mädchen und Jungen dürfen dann einvernehmlichen Geschlechtsverkehr haben, wenn Sie mindestens 13 (statt bisher 14) Jahre ist und Er nicht mehr als drei (bisher zwei) Jahre älter ist.

Sogenannte „beischlafähnliche Handlungen“ wie z.B. Oral- oder Analverkehr, die bisher ab 12 Jahren erlaubt waren, werden nun dem Geschlechtsverkehr gleichgestellt.

Weniger intensive Intimkontakte sind auch künftig ab 12 Jahren erlaubt, sofern der Partner nicht mehr als vier Jahre älter ist.

Schnipselweise Tübingen — Photo/Montage: hl, fe

<u>Riskier doch mal ’nen Blick!</u>

Endlich! Der Lichtblick in Tübingen

Eine Zeitung für unsere Zukunft!

(tb) Hier ist sie! Die erste Ausgabe des Tübinger lichtblicks. Die Jugendzeitung aus Tübingen für Tübingen. Eine Zeitung, wie sie hier im Lande noch nie gesehen wurde. Eine Zeitung von und für Kinder und Jugendliche. Alles, was Du wissen willst, erfährst Du ab sofort, alle zwei Monate, aus dem lichtblick. Hier bekommst Du den vollen Durchblick. Politik, Unterhaltung, der Schulblick, Tübinger Infos und vieles mehr sind Inhalte dieser Zeitung. Der lichtblick will informieren und unterhalten. Und das Gute daran: Dich kostet dies keinen Euro; den lichtblick gibt es kostenlos, direkt zu Dir in die Schule.

Nach langer Vorbereitungszeit ist es nun endlich soweit. Das Kinder- und Jugend-Umweltforum (KIJUMFO) gibt die erste Ausgabe der Jugendzeitung **lichtblick** heraus. Bisher nur eine sogenannte „Nullnummer“, die in einer kleinen Auflage an Sponsoren und Interessierte geschickt wird. Mit dieser Ausgabe sollen außerdem viele Leute gesucht werden, die mitmachen wollen. Später soll der Lichtblick dann an alle interessierten Schüler verteilt werden.

Wenn alles gut geht, gibt es also den **lichtblick** bald jeden zweiten Monat zu Dir in die Schule oder gegen Bezahlung des Portos direkt zu Dir nach Hause. Doch den **lichblick** kannst Du nicht nur lesen. Jeder, der Lust hat, kann selbst etwas schreiben, zeichnen, malen oder fotographieren. Dies erscheint dann in einer der nächsten Ausgaben. Versucht´s doch mal! Falls Du regelmäßig schreiben willst, komm doch einfach bei einer unserer Redaktionssiztungen vorbei. Die Termine erfährst du auf unserer Homepage oder per Telefon! Doch nun erstmal viel Spaß beim Lesen, Rätseln, Lachen. Denn nur gut informierte Menschen können ihre Zukunft selbst und wirklich frei gestalten.

<u>Wahlrecht:</u>

Ein Fünftel durfte nicht wählen!

Bundestagswahl ist gelaufen

Tübingen, Bonn (hl): Es ist entschieden: Der Bundestag wurde neu gewählt, die neue Bundesregierung ist vereidigt. Auch Tübingen ist in dem neuen Kabinett vertreten. Aber: Jeder fünfte Bundesbürger durfte nicht wählen!

Eine Ära ist zuende gegangen! Länger als jeder andere Kanzler zuvor hat Helmut Kohl dieses Land regiert. Seine Verdienste um die deutsche und europäische Einigung sind unbestritten.

Aber das Volk hat ihn in der letzten Wahl abgewählt. Zum ersten Mal in der Geschichte der Bundesrepublik haben die Wähler einem amtierenden Kanzler die rote Karte gezeigt und ihn vom Platz geschickt.

Doch nicht das ganze Volk war daran beteiligt. Rund ein Fünftel der Bundesbürger hat kein Wahlrecht: die Kinder und Jugendlichen. Obwohl es bereits mehrere Klagen bis hinauf zum Bundesverfassungsbericht gegeben hat (siehe unseren Artikel „Klage für Kinderwahlrecht erneut aus formalen Gründen abgewiesen“ auf Seite 2) und obwohl immer mehr prominente Politiker die Forderung nach einer Senkung oder gar Aufhebung der Altersgrenze beim Wahlrecht fordern, wird dieses Recht immer noch einem Teil des Volkes vorenthalten.

Mach’ mit beim lichtblick!

Du hast Lust, eine Zeitung zu machen? Du willst gerne Artikel schreiben, fotografieren, recherchieren? Du willst gerne zeichnen, Anzeigen sammeln, . . . ? Und das nicht nur auf Papier, sondern auch im Internet!

Hier kannst Du es!!!

Melde Dich einfach bei:

Redaktion lichtblick
c/o Heiner Lamprecht
Philosophenweg 79
72076 Tübingen
Tel.: 07071/600 162

lichtblick im Internet

Der Tübinger Lichblick ist auch im Internet zu finden! Als Beitrag der Herausgeber zur Lokalen Agenda 21 in Tübingen kann man die Artikel des lichblicks abrufen unter:

http://agenda21.ggi.uni-tuebingen.de/kijumfo

Was steht diesmal drin?

Abbildung 4.12: Beispielseite einer Zeitung

Kapitel 5

Gleitobjekte

Die bislang beschriebenen Umgebungen zum Setzen von Tabellen u. ä. entfalten ihre Wirkung genau an der Stelle, an der sie im Text auftreten. Ist auf einer Seite nicht mehr genügend Platz, so wandert die Umgebung komplett auf die nächste Seite. Manchmal erhält man dadurch jedoch nicht optimal gesetzte Texte. Außerdem ist es nicht immer nötig, daß eine Tabelle oder Abbildung *genau* an dieser Stelle erscheint. Sie könnte also auch ein wenig nach vorne oder nach hinten wandern. Dies kann man bei LaTeX mit sog. Gleitobjekten erreichen. Dabei wird versucht, die Abbildung oder Tabelle *in der Nähe* ihres Auftretens zu setzen. In der Regel landet sie auf der Seite, auf der sie im Text erscheint. Ist dies nicht möglich, so wandert sie auf die nächste Seite, wobei der Rest mit nachfolgendem Text aufgefüllt wird.

Es gibt zwei verschiedene Arten von Gleitobjekten: Abbildungen (`figure`) und Tabellen bzw. Tafeln (`table`).

5.1 Abbildungen (`figure`)

Die Umgebung

```
\begin{figure}[htbp]
\end{figure}
```

setzt den Inhalt an eine Stelle, an der er ohne Seitenumbruch vollständig hinpaßt. Die Lage kann man mit den Optionen `h`, `t`, `b`, `p` und `H` beeinflussen. Sie geben die Lage auf der jeweiligen Seite an. `t` steht dabei für „oberer Rand“, `b` für „unterer Rand“, `p` setzt die Abbildung auf eine extra Seite, `h` zeigt LaTeX an, daß die Abbildung „wenn möglich hier“ gesetzt werden soll. In LaTeX 2_ε gibt es einen weiteren Plazierungsparameter, `!` ‘bang’, der vorübergehend alle Plazierungsbeschränkungen aufhebt. Bang muß immer zusammen mit mindestens einem der vier anderen Parameter angegeben werden. Wird die Plazierung nicht näher angegeben, so wird die Voreinstellung (`[tbp]`) verwendet.

Das Paket `float` bietet darüberhinaus die Option `H`, mit der LaTeX gezwungen wird, die Abbildung „genau hier“ zu setzen (Lingnau, 1995).

Gelingt es LaTeX nicht, eine Abbildung zu plazieren, so wandert sie ans Ende des jeweiligen Kapitels oder Dokumentes und schiebt dabei sämtliche andere Gleitobjekte vor sich her! Da dies unter Umständen große Probleme bereiten kann, kann man mit Hilfe der beiden Befehle

```
\clearpage
\cleardoublepage
```

die Ausgabe aller bis dahin aufgeschobener Abbildungen erzwingen. Die laufende Seite wird dabei beendet. Bei `\cleardoublepage` beginnt der nachfolgende Text darüberhinaus auf einer *rechten* (ungeraden) Seite.

5.2 Tafeln (`table`)

Die `table`-Umgebung wird genauso verwendet wie die `figure`-Umgebung. Der Unterschied zwischen beiden besteht darin, daß sie bei der Überschrift das Wort „Tabelle“ einsetzt. Außerdem wird die Überschrift ins Tabellenverzeichnis übernommen. Bei Tabellen wird die Bezeichnung entweder *über* oder *unter* die Tabelle gesetzt. Man sollte dies jedoch innerhalb eines Dokumentes einheitlich machen.

```
\begin{table}[htbp]
  \caption{...} \label{...}
  \begin{center}
    \begin{tabular}{...}
    ...
    \end{tabular}
  \end{center}
\end{table}
```

Abbildung 5.1: Aufbau einer `table`-Umgebung

Möchte man eine Bezeichnung setzen, so dient dazu innerhalb der `figure`- oder `table`- Umgebung der Befehl

```
\caption{Titel}
```

Dabei ist das Wort „Tabelle“ (bzw. „Abbildung“) wegzulassen. Dies wird von LaTeX automatisch eingesetzt. Bei Abbildungen wird die Bezeichnung im Allgemeinen *unter* die Abbildung gesetzt.

Die Umgebungen `figure` und `table` gibt es noch in einer modifizierten Form, bei der ein `*` an den Namen gehängt wird (`figure*`, `table*`). Dies bedeutet, daß das Objekt bei mehrspaltigem Druck über alle Textspalten reicht.

5.3 Plazierung steuern

Die Plazierung von Gleitobjekten unterliegt verschiedenen Einschränkungen, die jedoch zum Teil beeinflußt werden können. So darf ein Gleitobjekt, das am unteren Ende einer Seite plaziert werden soll nicht höher als 30% der aktuellen Seitenhöhe sein. Möchte man also ein großes Gleitobjekt am unteren Ende plazieren, so wird LaTeX dies in der Regel auf eine eigenen Seite setzen. Nachfolgend werden die einzelnen Parameter aufgelistet und erläutert. Dabei bezeichnet der Begriff „Gleitobjekt“ eine `figure`- oder `table`-Umgebung. Eine „Gleitobjektseite“ ist demnach eine Seite, die ausschließlich Gleitobjekte enthält. Fußnoten und Marginalien, die strenggenommen auch Gleitobjekte sind, unterliegen nicht dem bei den übrigen Gleitobjekten verwendeten Plazierungsalgorithmus. Ihre Plazierung wird von den hier aufgeführten Parametern nicht beeinlußt.

Parameter der Plazierung von Gleitobjekten

`topnumber` (Voreinstellung: 2) maximale Anzahl von Gleitobjekten, die am Kopf einer Seite gesetzt werden dürfen. Wird mit dem Befehl `\setcounter` verändert.

`dbltopnumber` (Voreinstellung: 2) Gleiche Bedeutung wie `topnumber`, gilt für zweispaltige Gleitobjekte auf zweispaltigen Seiten.

`bottomnumber` (Voreinstellung: 1) maximale Anzahl von Gleitobjekten, die am Fuß einer Seite gesetzt werden dürfen. Wird mit dem Befehl `\setcounter` verändert.

`totalnumber` (Voreinstellung: 3) maximale Anzahl von Gleitobjekten, die insgesamt auf einer Seite plaziert werden dürfen. Wird mit dem Befehl `\setcounter` verändert.

`\topfraction` (Voreinstellung: 0.7) Anteil einer Seite, die von Gleitobjekten am Kopf der Seite maximal eingenommen werden darf. Wird mit dem Befehl `\renewcommand` verändert.

`\dbltopfraktion` (Voreinstellung: 0.7) Analog zu `\topfraction`, gilt für zweispaltige Gleitobjekte auf zweispaltigen Seiten.

`\bottomfraction` (Voreinstellung: 0.3) Anteil einer Seite, die von Gleitobjekten am Fuß der Seite maximal eingenommen werden darf. Wird mit dem Befehl `\renewcommand` verändert.

`\textfraction` (Voreinstellung: 0.2) Mindestanteil, der von Text eingenommen werden muß. Wird mit dem Befehl `\renewcommand` verändert. Der Wert gilt nicht für Seiten, die nur Gleitobjekte enthalten.

`\floatpagefraction` (Voreinstellung: 0.5) Mindestanteil einer Gleitobjektseite, der mindestens von Gleitobjekten eingenommen werden muß. Dadurch wird der Leerraum auf einer Gleitobjektseite begrenzt. Kann der Wert nicht erreicht werden, z. B. weil nicht genug Objekte zur Verfügung stehen, gibt LaTeX eine Warnung aus. Wird mit dem Befehl `\renewcommand` verändert.

`\dlbfloatpagefraction` (Voreinstellung: 0.5) Analog zu `\floatpagefraction`, gilt für zweispaltige Gleitobjekte auf zweispaltigen Seiten.

`\floatsep` (Voreinstellung: `12pt plus 2pt minus 2pt` für Dokumente mit Schriftgröße 10pt und 11pt; `14pt plus 2pt minus 2pt` für Dokumente mit Schriftgröße 12pt) Gummilänge, die den zusätzlichen vertikalen Abstand zwischen Gleitobjekten angibt. Wird mit dem Befehl `\setlength` verändert.

`\dblfloatsep` (Voreinstellung: `\floatsep`) Analog zu `\floatsep`), gilt für zweispaltige Gleitobjekte auf zweispaltigen Seiten.

`\textfloatsep` (Voreinstellung: `20pt plus 2pt minus 4pt`) Gummilänge, die den zusätzlichen vertikalen Abstand zwischen Gleitobjekten am Kopf oder am Fuß einer Seite und dem Text angibt. Wird mit dem Befehl `\setlength` verändert.

`\dbltextfloatsep` (Voreinstellung: `\textfloatsep` auf einer Textseite, `8pt plus 2fil` auf einer Gleitobjektseite) Analog zu `\textfloatsep`), gilt für zweispaltige Gleitobjekte auf zweispaltigen Seiten.

`\intextsep` (Voreinstellung: `\floatsep`) Gummilänge, die den zusätzlichen vertikalen Abstand zwischen Gleitobjekten innerhalb einer Seite und dem umgebenden Text angibt. Wird mit dem Befehl `\setlength` verändert.

`\topfigrule` Befehl zum erzeugen einer Trennlinie zwischen Gleitobjekten am Kopf einer Seite und dem nachfolgenden Text. Die Linie wird direkt vor dem Abstand `\textfloatsep` eingefügt.

`\botfigrule` Gilt analog zu `\topfigrule`. Allerdings wird die Linie nach dem Abstand `\textfloatsep` eingefügt.

`\dblfigrule` Analog zu `\topfigrule`, gilt für zweispaltige Gleitobjekte auf zweispaltigen Seiten.

5.3.1 Feinabstimmung der Plazierung

Möchte man in die Plazierung von Gleitobjekten von Hand eingreifen, so sollte man sich zunächst die Abhängigkeiten der oben aufgeführten Parameter vor Augen führen.

Verwendet man in einem Dokument beispielsweise sehr viele Gleitobjekte, so landen viele davon in der Regel auf einzelnen Seiten, die jeweils nur zu rund der Hälfte gefüllt sind. Ursache hierfür ist, daß LaTeX eine eigene Gleitobjektseite anlegt, wenn die noch nicht plazierten Objekte zusammen mindestens die Hälfte einer Seite einnehmen. Nun könnte man den Wert von `\floatpagefraction` ein wenig vergrößern. Die Folge wäre jedoch, daß die Gleitobjekte unter Umständen sehr viel weiter im Text wandern, da LaTeX mehr Gleitobjekte „sammeln“ muß, bevor eine eigene Seite für die Objekte erzeugt werden kann.

Ein weiter Grund für viele Gleitobjektseiten ist die Verwendung großer Gleitobjekte, insbesondere, wenn diese mit dem Parameter `[b]` am unteren Ende einer Seite plaziert werden sollen. Hier dürfen standartmäßig nur Gleitobjekte plaziert werden, die maximal 30% der Seitenhöhe einnehmen. Ist ein Gleitobjekt größer, so kann es an der Stelle nicht erzeugt werden. Dadurch können aber auch alle nachfolgenden Gleitobjekte, die ebenfalls mit `[b]` plaziert werden, nicht gesetzt werden. Gleiches gilt analog auch für die anderen Parameter (`[h] und [t]`). Landen in einem Dokument viele Gleitobjekte auf einer Gleitobjektseite oder sogar am Ende des Kapitels, so sollte man zunächst die Plazierungsparameter überprüfen.

5.3.2 `afterpage`

Besonders wenn sehr viele sehr große Gleitobjekte verwendet werden, kann LaTeX mit der Positionierung überfordert werden. Mit den Befehlen `\clearpage` und `\cleardoublepage` kann man zwar die Ausgabe aller bis dahin aufgelaufenen Gleitobjekte ausgeben, dies hat jedoch den Nachteil, daß die aktuelle Seite abgebrochen wird. Oft wäre es aber sinnvoll, die Ausgabe erst auf der nächsten Seite zu erzwingen. Hierfür führt das `afterpage` Paket von David Carlisle (Carlisle, 1995) den Befehl

```
\afterpage{Befehle}
```

ein. Er bewirkt, daß die Befehle, die ihm als Argument übergeben werden, erst nach der aktuellen Seite ausgeführt werden. Mit `\afterpage{\clearpage}` kann man also die Ausgabe der Gleitobjekte erzwingen, wobei die aktuelle Seite normal beendet wird. Zu beachten ist jedoch, daß der Befehl `\afterpage` nicht mit der Option `twocolumn` funktioniert.

Der Befehl kann auch in Zusammenhang mit mehrseitigen Tabellen verwendet werden. Die `longtable` Umgebung (siehe Abschnitt 7.4.2 auf Seite 123) hat manchmal Probleme, den Text vor und nach der Tabelle korrekt zu setzen. Der Befehl

```
\afterpage{\clearpage\input{tabelle.tex}}
```

kann hier helfen, die Tabelle, die in der Datei `tabelle.tex` definiert ist, korrekt zu setzen.

5.3.3 flafter

Verwendet man häufig Gleitobjekte, so wird man feststellen, daß LaTeX sie mit Vorliebe an den Anfang einer Seite setzt. Auf diese Weise kann es aber passieren, daß ein Gleitobjekt vor der Stelle auftritt, an der das Objekt in der Eingabedatei steht. Dies ist nicht immer akzeptabel. Eine Lösung des Problems ist jedoch nicht so einfach, da hierzu der Plazierungsalgorithmus grundlegend geändert werden muß. Das `flafter` Paket erreicht genau dieses. Bei Verwendung des Paketes ist sichergestellt, daß alle Gleitobjekte nie vor der Stelle auf einer Seite erscheinen, an der sie in der Eingabedatei stehen.

Der Befehl `\flafter` kann jedoch in bestimmten Fällen ein wenig zu drastisch sein. Steht beispielsweise in der Mitte einer Seite ein neuer `\section`-Befehl, so möchte man Gleitobjekte, die zu diesem Abschnitt gehören, nicht am Kopf der Seite ausgeben, da diese dann optisch noch zum vorherigen Abschnitt gehören. Das bedeutet aber weder, daß auf dieser Seite überhauptkeine Gleitobjekte am Kopf der Seite stehen dürfen, noch, daß die Gleitobjekte nicht vor ihrem Verweis erscheinen dürfen. Gelöst werden kann das Problem mit dem Befehl

```
\suppressfloats[Position]
```

Der optionale Parameter *`Position`* darf entweder `t` oder `b` sein. Der Befehl legt fest, daß auf der Seite, auf der er auftritt, keine weiteren Gleitobjekte am Kopf (`t`) oder am Fuß (`b`) der Seite auftreten dürfen. Wird ***`Position`*** weggelassen, so dürfen überhauptkeine weiteren Gleitobjekte auf der Seite ausgegeben werden. Gleitobjekte, die vor dem Auftreten des Befehls in der Eingabedatei stehen, können jedoch noch gesetzt werden.

5.4 Gleitobjekte umfließen lassen

Oft kommt es vor, daß die verwendeten Gleitobjekte nicht über die gesamte Textbreite reichen. Sind sie recht schmal, so möchte man eventuell den Text drumherum fließen lassen. Text neben die Abbildungen zu setzten, klappt aber normalerweise nur mit einer `minipage` Umgebung. Dann jedoch muß man von Hand festlegen, welcher Text neben dem Bild erscheint. Normaler Fließtext ist nicht möglich. Um dieses Problem zu beheben, stehen dem Anwender drei Pakete zur Verfügung.

Allgemein sollte man bei der Verwendung schmaler Abbildungen ein paar Dinge beachten. Da der Platz neben den Abbildungen recht schmal ist, sollte man hier nur normalen Text verwenden. Überschriften sollten auf jeden Fall vermieden werden. Formeln sind zu akzeptieren, solange sie passen.

5.4.1 `floatfig`

Das Paket `floatfig` (von Thomas J. Reid) definiert eine neue Umgebungen: `floatingfigure`. Die Syntax lautet:

```
\begin{floatingfigure}{Breite}
```

Die Umgebung ist völlig kompatibel mit der normalen `figure` Umgebung von LaTeX 2_{ε} (Kneser, 1990). Sie erzeugt eine schmale Abbildung mit der angegebene ***`Breite`***, um die der Text herumfließt. Dazu muß jedoch vor dem ersten Auftreten der Umgebung der Befehl

```
\initfloatingfigs
```

stehen. Am besten setzt man den Befehl hinter den Befehl `\begin{document}`. Die Umgebung kann nur zwischen zwei Absätzen verwendet werden. LaTeX versucht daraufhin, die Abbildung so bald wie möglich zu setzen. Die Abbildung erscheint jedoch immer nur am Anfang eines Absatzes. Reicht sie über

Dies ist ein Bild

Abb. 5.2: Beispiel einer `floatingfigure` Umgebung mehrere Absätze hinaus, so werden diese natürlich ebenfalls entsprechend eingerückt. `floatingfigure` Umgebungen können mit `figure` Umgebungen beliebig gemischt werden. Sie können ebenfalls mit Bildunterschriften versehen werden, die auch im Abbildungsverzeichnis erscheinen. Die Bilder werden so gesetzt, daß sie auf ungeraden Seiten immer rechts und auf geraden Seiten immer links erscheinen. Bei zweiseitigem Druck sitzen die Abbildungen also immer auf der Außenseite.

Die `floatingfigure` Umgebung kann nicht mit der Option `twocolumn` zusammen verwendet werden. Außerdem erscheint die Abbildung nie in einem Absatz, der am Anfang der Seite beginnt. Auch kann die Abbildung erst gesetzt werden, wenn alle vorherigen Abbildungen untergebracht werden konnten. Auf diese Weise bleibt aber auch die Nummerierung der Abbildungen erhalten.

Das Beispiel in Abbildung 5.2 wurde mit folgenden Befehlen erzeugt:

```
\begin{floatingfigure}{65mm}
  \begin{center}
    \framebox[35mm]{Dies ist ein Bild}
    \caption{Beispiel ...}\label{fig:floatfig}
  \end{center}
\end{floatingfigure}
```

5.4.2 `wrapfig`

Das Paket `wrapfig` von Donald Arseneau bietet ebenfalls die Möglichkeit, Text um eine schmale Abbildung fließen zu lassen. Es definiert die Umgebung `wrapfigure`. Die Syntax erfordert drei Argumente:

`\begin{wrapfigure}[`*Zeilen*`]{`*Position*`}{`*Breite*`}`

Zeilen gibt als optionales Argument an, wieviel schmale Zeilen gesetzt werden. Formeln gelten dabei als drei Zeilen. Fehlt das Argument, so wird die Anzahl der schmalen Zeilen automatisch bestimmt.

Position legt fest, auf welcher Seite des Absatzes die Abbildung erscheinen soll. Erlaubte Werte sind `l` und `r`.

Breite gibt die Breite des Bildes an.

Die `wrapfigure` Umgebung erzeugt auf der linken oder rechten Seite des Absatzes eine Abbildung der angegebenen Breite. Zwischen Text und Abbildung fügt LaTeX dabei einen kleinen Zwischenraum der Breite `\columnsep` ein. Die Anzahl der verkürzten Zeilen wird von der Höhe der Abbildung zuzüglich des

Wertes von `\intextsep` bestimmt, kann aber durch das optionale Argument geändert werden.

Eine `wrapfigure` Umgebung sollte nicht innerhalb von anderen Umgebungen erscheinen. Anders als `floatingfigure` kann sie aber auch bei mehrspaltigem Text verwendet werden.

Ähnlich wie bei der `floatingfigure`-Umgebung, kann innerhalb des Bildes in einer `wrapfigure`-Umgebung *kein* Seitenumbruch erfolgen. Ist ein Seitenumbruch nötig, so reicht das Bild über den unteren Rand der Seite hinaus. Die Zeilen des umfließenden Textes werden korrekt umgebrochen, jedoch werden die ersten Zeilen auf der neuen Seite immer noch eingerückt.

Eine `wrapfigure` Umgebung

Abbildung 5.3: Beispiel für eine `wrapfigure` Umgebung

Im Unterschied zu `floatingfigure` kann diese Umgebung nicht gleiten. Das bedeutet, sie tritt immer an der Stelle auf, an der sie im Text steht. Dies kann zur Folge haben, daß die Reihenfolge der Abbildungen nicht mehr stimmt.[1] Auch bestimmt LaTeX 2_ε die Seite nicht selbst. Dies ist dem Anwender überlassen. Daher sollten diese Abbildungen erst kurz vor dem endgültigen Ausdrucken gesetzt werden. Selbst kleinere Änderungen am Text können die sorgfältige Positionierung schnell wieder zunichte machen.

Das Beispiel in Abbildung 5.3 wurde mit folgenden Befehlen erzeugt.

```
\begin{wrapfigure}{l}{65mm}
  \begin{center}
    \framebox[50mm]{Eine \texttt{wrapfigure} Umgebung}
    \caption{Beispiel für eine \texttt{wrapfigure} Umgebung}
    \label{fig:wrapfig}
  \end{center}
\end{wrapfigure}
\noindent
Im Unterschied zu \texttt{floatingfigure} kann diese Umgebung ...
```

Die `wrapfigure` Umgebung sollte man möglichst nur zwischen zwei Absätzen verwenden. Soll ein Bild innerhalb eines Absatzes auftreten, so muß man die Umgebung genau an einem Zeilenumbruch einfügen.

[1] Die Nummer der Abbildung wird in dem Moment bestimmt, in dem die Abbildung erzeugt wird, also im Text auftritt, nicht an der Stelle, an der sie schließlich gesetzt wird.

5.4.3 `picinpar`

Bilder innerhalb eines Absatzes erzeugt man besser mit dem `picinpar` Paket von Friedhelm Sowa und Alan Hoenig. Das Paket bietet die `window` Umgebung, die es auch in zwei Varianten gibt: `figwindow` und `tabwindow`. Die beiden letztgenannten ermöglichen Unterschriften und sorgen für einen Eintrag in die entsprechenden Verzeichnisse. Da diese Umgebungen nicht gleiten können, muß man bei der Mischung mit normalen Gleitobjekten auf die Nummerierung achten. Aufgerufen wird die Umgebung mit dem Befehl

```
\begin{window}[Zeilen,Position,Fensterinhalt,Erklärung]
  Absatztext
\end{window}
```

Zeilen	gibt die Anzahl der Zeilen *vor* dem Bild an
Position	legt die Position des Fensters fest. Erlaubte Werte sind `r`, `l` und `c`.
Fensterinhalt	ist der Inhalt des Fensters.
Erklärung	beinhaltet eine eventuelle Unterschrift.

Einst lebte Sauron der Maia, den die Sindar in Beleriand Gorthaur nannten. Zu Anbeginn Ardas lockte ihn Melkor in seinen Bund, und er wurde  der größte und vertrauteste Diener des Feindes, und der gefährlichste, denn in hundert Gestalten erschien er, und lange vermochte er sich ein so edles und schönes Ansehen zu geben, wenn er wollte, daß er alle bis auf die Bedachtsamsten täuschte.

```
\begin{window}[3,c,%
 \epsfig{file=buch.eps,width=21mm},{}]
Einst lebte Sauron der Maia, den die
Sindar in Beleriand Gorthaur nannten.
Zu Anbeginn Ardas lockte ihn Melkor
in seinen Bund, und er wurde der
größte und vertrauteste Diener des
Feindes, und der gefährlichste,
...
\end{window}
```

Abbildung 5.4: Beispiel für eine Abbildung in einem Absatz

Kapitel 6

Briefe

LaTeX 2ε bietet eine Dokumentenklasse, die sich in vielerlei Dingen von den „normalen" Klassen zur Erstellung von Dokumenten unterscheidet. Sie dient dazu, Briefe zu setzen. Diese `letter`-Klasse ist jedoch sehr stark auf das amerikanische Brief- und Seitenformat ausgerichtet, so daß man, um auch in deutschsprachigen Raum Briefe mit LaTeX erstellen zu können, eine ganze Reihe von Anpassungen vornehmen muß.

Im folgenden soll zunächst der originale Brief-Stil von LaTeX vorgestellt werden. Im zweiten Teil dieses Kapitels werden dann Werkzeuge behandelt, mit denen man sich recht einfach einen eigenen Briefstil erstellen kann.

6.1 Das Original

Die Dokumentenklasse `letter` stellt verschiedene Befehle und Funktionen bereit, mit deren Hilfe man Briefe schreiben kann. Dabei ist es möglich, mit einer Eingabedatei gleich mehrere Briefe zu schreiben. Innerhalb eines Briefes können fast alle Befehle von LaTeX 2ε wie gewohnt verwendet werden. Es gibt jedoch auch Befehle, wie die Gliederungsbefehle, die in dieser Klasse nicht benutzt werden können. Aber es macht schließlich auch wenig Sinn, einen Brief in verschiedene Kapitel zu unterteilen und ein Literaturverzeichnis zu erstellen.

Geladen wird die Klasse wie üblich mit

```
\documentclass[Optionen]{letter}
```

Als Optionen können alle der Klasse `article` verwendet werden mit Ausnahme von `twoside` und `titlepage`.

In der Regel wird man im Vorspann der Datei den Namen und die Adresse des Absenders angeben. Die geschieht mit dem Befehl

```
\address{Absender-Anschrift}
```

Dieser Eintrag besteht normalerweise aus mehreren Zeilen, die durch \\ getrennt werden. Den Namen des Absenders legt man mit

```
\signature{Absender}
\name{Absender}
```

fest. Diese beiden Befehle sind im Prinzip identisch. Der mit diesen Befehlen festgelegte Absendername wird am Ende des Briefes als Unterschrift gesetzt.

Wenn die vorgenannten Befehle im Vorspann der Datei auftauchen, so gelten sie für das gesamte Dokument. Will man also mehrere Briefe schreiben, so genügt es, diese Angaben nur einmal zu machen. Die Befehle können jedoch auch in jeder einzelnen `letter`-Umgebung angegeben werden. Sie wirken dann nur für diesen einen Brief. Die Befehle müssen *vor* dem `opening`-Befehl stehen.

Der Textteil beginnt wie gewöhnlich mit `\begin{document}`. Die einzelnen Briefe werden mit der Umgebung

```
\begin{letter}{Empfänger}
        Brieftext
\end{letter}
```

erzeugt. Hierbei wird mit *Empfänger* Name und Adresse des Empfängers angegeben. Auch hier werden in der Regel mehrzeilige Einträge verwendet. Im Brief kann man eine Anrede und einen Gruß schreiben, die beide abgesetzt vom eigentlichen Text auftreten. Die Befehle dazu lauten

```
\opening{Anrede}
\closing{Gruß}
```

Nach dem `closing`-Befehl dürfen nur noch zwei weitere Befehle auftreten. Mit

```
\cc{Name 1 \\ Name 2 \\ ...}
```

kann eine Verteilerliste erzeugt werden. Hierbei erscheint unter dem Gruß linksbündig ein „cc:“, bzw. bei der Option `german` ein „Verteiler:“. Der Befehl

```
\encl{Anlage 1 \\ Anlage 2 \\ ...}
```

erzeugt eine Anlagenliste. Analog zur Verteilerliste erscheint ein „encl:“, bzw. ein „Anlagen:“.

Soll im Anschluß an den `closing`-Befehl dennoch weiterer Text folgen, so muß zunächst der Befehl

```
\ps
```

auftreten. Dieser Befehl erzeugt selbst *keine* Ausgabe. Er ermöglicht nur, nach dem `closing`-Befehl weiteren Text zu schreiben.

LaTeX beginnt beim Setzen des Briefes mit dem Absender. Dieser erscheint oben rechts. Kurz unterhalb der Adresse folgt ebenfalls rechts das aktuelle Datum. Als nächstes werden die Angaben über den Empfänger auf der linken Seite gesetzt. Schließlich beginnt der eigentliche Brieftext mit der Anrede. Die Grußformel erscheint ein Stück nach rechts eingerückt. Mit einigem Abstand folgt der Absendername unterhalb der Grußformel. Ein Beispiel für einen Standart-Brief von LaTeX ist in Abbildung 6.1 dargestellt. Erzeugt wurde das Beispiel mit folgenden Zeilen:

```
\documentclass{letter}
\address{Vingilot-Adventure-Traveling Corp.\\
         Abteilung Auslandsreisen \\
         Hugo P. Hansen \\
         Nan-Pelagir 123 \\
         98765 Alqualonde}
\signature{Hugo P. Hansen}
\begin{document}
\begin{letter}{Versuchsanstalt \\
         für \LaTeXe-Briefe \\
         12345 \TeX-Hausen}
\opening{Liebe Frau Scheppendorff}
Für Ihre Anfrage nach der Gestaltung eines neuen
Briefkopfes ...

\closing{Mit freundlichen Grüßen\\Ihr}
\end{letter}
\end{document}
```

In einer Datei dürfen beliebig viele `letter`-Umgebungen auftreten. Es ist jedoch zu beachten, daß die Umgebungen selbstverständlich nicht in einander verschachtelt werden können.

Abschließend kann man von LaTeX auch automatisch „Aufkleber“ mit den Empfängeradressen erstellen lassen. Hierzu muß in der Präambel der Datei der Befehl

```
\makelabels
```

auftreten. Dieser erzeugt eine Box, die 4.25 Zoll breit und 2 Zoll hoch ist. Falls mehrere Briefe mit einer Datei erzeugt werden, so werden die Adressaufkleber in zwei Spalten nebeneinander ausgegeben.

Vingilot-Adventure-Traveling Corp.
Abteilung Auslandsreisen
Hugo P. Hansen
Nan-Pelagir 123
98765 Alqualonde

30. Dezember 1999

Versuchsanstalt
für LaTeX 2_ε-Briefe
12345 TeX-Hausen

Liebe Frau Scheppendorff

Für Ihre Anfrage nach der Gestaltung eines neuen Briefkopfes für unsere Firma möchte ich mich herzlich bedanken. Ebenso für die zahlreichen Anregungen und Tips.

Die Arbeiten sind mittlerweile abgeschlossen. Wie Sie sehen können, durchaus erfolgreich. Der Vorstand hat unseren Vorschlägen gestern zugestimmt.

Mit freundlichen Grüßen
Ihr

Hugo P. Hansen

Abbildung 6.1: Beispiel eines Standartbrief

6.2 Erstellen eines eigenen `letter`-Stiles

Das Beispiel in Abbildung 6.1 auf der vorherigen Seite zeigt die Grenzen, die einem mit den Grundfuktionen der `letter`-Klasse gesetzt sind. Vor allem bei Geschäftsbriefen sind weitere Möglichkeiten der Gestaltung wünschenswert. Auch wenn diese Dinge nicht direkt bereitgestellt werden, ist es doch recht einfach, sich einen eigenen Brief-Stil zu erstellen.

In diesem Kapitel werden verschiedene Werkzeuge, mit denen zusätzliche Befehle definiert und z. B. Seitenstile geändert werden, verwendet, ohne diese näher zu erläutern.

Oftmals werden in Firmen vorgedruckte Briefbögen verwenden, bei denen genau festgelegt ist, an welcher Stelle bestimmte Eintragungen (Datum, „Ihr Zeichen:", „Ihre Nachricht vom:", . . .) erfolgen sollen. Schreibt man den Brief mit der Schreibmaschine, so kann man diese Daten recht leicht an die richtige Stelle setzen. Will man das jedoch mit dem Computer erreichen, so wird die Aufgabe äußerst kompliziert, vor allem bei WYSIWYG-Software. Mit Hilfe von LaTeX ist es möglich, völlig auf vorgedrucktes Briefpapier zu verzichten. Dank den umfangreichen Graphik-Funktionen können auch größere Logos und Abbildungen mit eingebunden werden.

Als Beispiel soll ein Briefkopf für eine (fiktive) Firma mit mehreren Abteilungen erstellt werden. Der Briefkopf soll den Namen, die Abteilung und die Telefonnummer des Autors enthalten. Gleichzeitig soll die Adresse des Empfängers so erscheinen, daß der Brief problemlos in Fensterumschlägen verschickt werden kann. Ist der Brief länger als eine Seite, so soll auf den folgenden Seiten zunächst der Absender und darunter der Empfängername, die Seitenzahl und das Datum erscheinen. Der Fuß der ersten Seite enthält die allgemeine Adresse der Firma sowie die Bankverbindung. Alle weiteren Seiten bekommen einen leeren Fuß.

Der Briefkopf

Im ersten Schritt wird das Seitenformat festgelegt. Briefe haben oftmals eine größere Zeilenlänge als umfangreichere Dokumente. Außerdem wird das Format der Absätze festgelegt.[1]

```
\setlength{\parindent}{0mm}
\setlength{\parskip}{7pt plus1.5pt minus 3pt}
\textwidth165mm \oddsidemargin-4.9mm
```

[1] Wie das Beispiel zeigt, können in Stildateien Maßangaben auch ohne `\setlength` festgelegt werden. In diesem Fall reicht es, den neuen Wert direkt hinter den Längenbefehl zu setzen. Diese Möglichkeit besteht jedoch *nur* in Stil- und Klassendateien.

```
\marginparsep4mm
\ifcase \@ptsize\relax
    \textheight 53\baselineskip \or
    \textheight 50\baselineskip \or
    \textheight 46\baselineskip \fi
```

Der oben verwendete Befehl `\ifcase` dient dazu, die Anzahl der Zeilen auf einer Seite abhängig von der verwendeten Schriftgröße festzulegen. In diesem Fall ist eine Seite bei einer Grundschrift von 10pt 53 Zeilen, bei 11pt 50 Zeilen und bei 12pt 46 Zeilen lang. Der Befehl `\fi` dient dazu, den `\ifcase`-Befehl zu beenden.

Anschließend werden mehrere Boxen definiert, die später wieder aufgerufen werden. Zur Erzeugung dieser Boxen werden mehrere Befehle benötigt, weshalb eventuell der Abschnitt 4.7 auf Seite 65 zu Rate gezogen werden sollte.

```
\newsavebox{\head}        \newsavebox{\heada}
\newsavebox{\headb}       ...
```

Aus Platzgründen werden hier nicht alle Boxen aufgeführt. Nachdem die Boxen angelegt wurden, können sie nun näher definiert werden. Da die verwendeten Schriftgrößen von der für das Dokument gewählten Größe unabhängig sein müssen, ist es nötig hier absolute Größenangaben zu verwenden. Diese sind je nach Installation verfügbar. Falls nicht, sollte man sie sich neu definieren.

```
\sbox{\heada}{\xxvpt \bfseries
     Vingilot\,-Adventure\,-Traveling Corp.}
\sbox{\headb}{\bfseries \slshape
    "`Best Travels arround Ea"'}
\sbox{\head}{\parbox{\textwidth}{\usebox{\heada} \par
    \raisebox{-1mm}{\usebox{\headb}}\rule{108mm}{0.8pt}}}
```

Für den Kopf der restlichen Seiten werden eigene Boxen verwendet.

```
\savebox{\page}[\textwidth][c]
    {\xiipt \bfseries \hspace*{\fill}
        Vingilot\,-Adventure\,-Traveling Corp. \hfill
        Nan-Pelagir 123 \hfill 98765 Alqualonde \hfill}
\savebox{\pagebox}[\textwidth][c]
    {\parbox{\textwidth}{\usebox{\page} \\
        \rule[1mm]{\textwidth}{0.8pt} \vspace*{1pt}}}
```

Seitenfuß

Der Fuß der ersten Seite unterscheidet sich ebenfalls von dem der restlichen Seiten. Er besteht aus drei Boxen, die wiederum zu einer vierten zusammengefaßt werden.

```
\sbox{\foota}{\parbox{30mm}
    {\ixpt Nan-Pelagir \\ 987654 Alqualonde}}
\sbox{\footb}{\parbox{35mm}
    {\ixpt Tel.:\hfill+99-986-4384-0 \\
        Fax:\hfill+99-986-4385-0}}
\sbox{footc}{\parbox{28mm}
    {\ixpt Konto: 5544668 \\ BLZ: 990 875 321}}
\savebox{\foot}[\textwidth][c]
    {\parbox{\textwidth}{\rule{\textwidth}{0.8pt} \\[4pt]
        \hspace*{\fill} \usebox{\foota}
        \hspace{\fill} \usebox{\footb}
        \hspace{\fill} \usebox{\footc}
        \hspace*{\fill}}}
```

Adressen

Zum Schluß folgt die Definition der Absenderadresse, die im Sichtfenster des Umschlags erscheint.

```
\sbox{\return}{\parbox{80mm}{\underline
    {\viipt V\,-A\,-T Corp.
        \hspace{\fill} Nan-Pelagir 123
        \hspace{\fill} Alqualonde}
        \rule[-6mm]{0mm}{2mm}}}
```

Damit sind alle Boxen, die für den Briefbogen verwendet werden, vollständig definiert. Als nächstes werden verschiedene Befehle erzeugt, mit denen später die Abteilung und die Telefonnummer des Absenders festgelegt werden. Außerdem werden die Texte für die Anlagen und die Verteilerliste abgeändert, sowie Befehle für „Betr." und „Bezug" und die Anrede erzeugt.

```
\newcommand\dir[1]{\def \@dir {Direktion \\ #1}}
    \def \@dir {}
...
\newcommand\tel[1]{\def \@tel {Tel.: #1}}
    \def \@tel {}
\renewcommand\enclname{\underline{\scshape Anlage(n)}}
```

```
\renewcommand\ccname{\underline{\scshape Verteiler:}}
\newcommand\subject[1]{\def \@subject
    {\underline{\scshape Betr.:} #1}}
    \def \@subject {}
\newcommand\subref[1]{\def \@subref
    {\underline{\scshape Bezug:} #1}}
    \def \@subref {}
\renewcommand\opening[1]{\thispagestyle{firstpage}
    \rule{0mm}{55mm} \\
    \ifx \@empty \@subject \else \@subject \par
    \@subref \par \vspace {6mm} \fi
    #1 \par}
```

Bevor mit der Beschreibung des Seitenstils begonnen wird, soll das Erscheinungsbild der Empfängeradresse verändert werden. Ziel ist es, zwischen dem Namen und der Anschrift einen kleinen Zwischenraum zu erzeugen. Die verwendeten Befehle sollen an dieser Stelle nicht eingehender beschrieben werden. Interessierte Leser mögen statt dessen zum Beispiel das Buch „Einführung in TEX" von Norbert Schwarz zu Rate ziehen, das die Verwendung interner Boxbefehle von TEX ausführlich behandelt.

```
\def\to@label#1#2 {\setbox0\vbox
    {\parbox{80mm}{#1} \\ {#2}}
        \vbox to 0.5mm {\vss \box0 \vss}}
```

Seitenstile

Nun können die Seitenstile für die erste und die restlichen Seiten definiert werden. Diese werden mit den beiden internen Befehlen `\ps@firstpage` und `\ps@headings` aufgerufen.

```
\def\ps@firstpage {\headheight 5mm
    \def \@oddhead
      {\parbox{\textwidth}{\usebox{\head} \par
          \vspace{18mm}
          \parbox[t]{80mm}
          {\usebox{\return} \par
              \to@label {\toname} {\toaddress}} \hfill
              \parbox[t]{52mm}
          {\ifx \@empty \@dir \else \@dir \par \fi
              \ifx \@empty \@fin \else \@fin \par \fi
              \ifx \@empty \@aus \else \@aus \par \fi
```

```
                 \ifx \@empty \@tel \else \@tel \par \fi} \par
                 \vspace{10mm}
                 \parbox{\textwidth}
                 {\hspace*{\fill} Alqualonde, den \@date}}
    \def \@oddfoot {\usebox{\foot}}
    \def \@evenhead {} \def \@evenfoot {}}

\def\ps@headings {\headheight 5mm
    \def \@oddhead {\parbox{\textwidth}
        {\parbox{\textwidth}{\usebox{\pagebox} \\
            \headtoname \hspace*{0.2em} \toname
            \hfill \@date \hfill
            \pagename \hspace*{1pt} \thepage}}}
    \def \@oddfoot {\rule{\textwidth}{0.8pt}}
    \def \@evenhead {}
    \def \@evenfoot {}}
```

Die Definition eines eigenen Briefstils ist nun fertig. Abbildung 6.2 auf Seite 97 zeigt die erste Seite eines solchen Briefes. Die Eingabedatei für dieses Beispiel sieht folgendermaßen aus:

```
\documentclass{letter}
\usepackage{firmenbrief}
\aus{Hugo P. Hansen}
\tel{+99-869-4384-15}
\begin{document}
\begin{letter}{Versuchsanstalt \\
        für \LaTeXe-Briefe \\
        12345 \TeX-Hausen}
\subject{Vorschlag für einen Briefkopf}
\subref{Ihre Anfrage}
\opening{Liebe Frau Scheppendorff,}
für Ihre Anfrage nach der Gestaltung eines neuen Briefkopfes
für unsere Firma möchte ich mich herzlich bedanken. Ebenso
für die zahlreichen Anregungen und Tips.

...

\closing{Mit freundlichen Grüßen\\Ihr}
\end{letter}
\end{document}
```

Sicherlich ist nicht zu erwarten, daß jemand, der sich erst kurze Zeit mit LaTeX 2_ε beschäftigt, einen solchen Briefkopf selbständig entwerfen kann. Der hier gezeigt Weg kann aber dafür sehr gut als Beispiel und Ausgangpunkt für eigene Entwürfe dienen.

Vingilot-Adventure-Traveling Corp.

„Best Travels arround Ea"

V-A-T Corp. Nan-Pelagir 123 Alqualonde

Versuchsanstalt
für LaTeX 2_ε-Briefe
12345 TeX-Hausen

Abteilung
Auslandsreisen
Hugo P. Hansen
Tel.: +99-869-4384-15

Alqualonde, den 13. Mai 1997

Betr.: Vorschlag für einen Briefkopf

Bezug: Ihre Anfrage

Liebe Frau Scheppendorff,

für Ihre Anfrage nach der Gestaltung eines neuen Briefkopfes für unsere Firma möchte ich mich herzlich bedanken. Ebenso für die zahlreichen Anregungen und Tips.

Die Arbeiten sind mittlerweile abgeschlossen. Wie Sie sehen können, durchaus erfolgreich. Der Vorstand hat unseren Vorschlägen gestern zugestimmt.

Mit freundlichen Grüßen
Ihr

Hugo P. Hansen

Nan-Pelagir 123	Tel.:	+99-869-4384-0	Konto: 5544668
98765 Alqualonde	Fax:	+99-869-4385-0	BLZ: 990 875 321

Abbildung 6.2: Beispielbrief

Kapitel 7

Tabellen

Wissenschaftliche Daten kann man oftmals sehr gut in Tabellenform darstellen. LaTeX 2ε bietet hier eine Reihe zum Teil sehr ausgereifter und leistungsfähiger Werkzeuge. Dabei ist allerdings zu beachten, daß LaTeX *keine* Tabellenkalkulation ist! Es ist also nicht möglich, Berechnungen innerhalb des Dokumentes durchzuführen.

Bereits mit Hilfe der in Abschnitt 4.6.3 auf Seite 64 beschriebenen `tabbing` Umgebung lassen sich umfangreiche Tabellen erstellen. Zusammen mit den in Abschnitt 4.7.1 vorgestellten Boxbefehlen könnte man sogar gerahmte Tabellenstrukturen erzeugen. Bei komplexeren Problemen wird eine solche Vorgehensweise allerdings sehr aufwendig. Im folgenden soll zunächst die normale `tabular` bzw. `array` Umgebung von LaTeX 2ε vorgestellt werden. In einem zweiten Teil des Kapitels wird dann auf die verschiedenen Makropakete eingegangen.

7.1 Eigentliche Tabellen (`tabular`)

Die `tabular`-Umgebung ermöglicht es, Daten in Tabellen zu setzen. Anders als bei der `tabbing` Umgebung, wird hier jedoch die benötigte Breite der einzelnen Spalten automatisch bestimmt. Es lassen sich außerdem leicht vertikale oder horizontale Linien einbinden. Darüberhinaus kann man das Aussehen der Spalten über die Optionen verändern. Aufgerufen wird sie mit einem der Befehle

```
\begin{array}[Position]{Deklaration}
\begin{tabular}[Position]{Deklaration}
\begin{tabular*}{Breite}[Position]{Deklaration}
```

Die `array` Umgebung kann nur im mathematischen Modus verwendet werden (siehe Abschnitt 8.3 auf Seite 137). Der optionale Parameter ***Position*** gibt

Tabelle		
Butter	5kg	12,- DM
Milch	3l	7,80 DM
Äpfel	4kg	32,- DM
	8kg	40,- DM
Gesamt:		91,80 DM

```
\fbox{\begin{tabular}{|c|c||c|}
  \hline
  \multicolumn{3}{|c|}{Tabelle} \\
  \hline \hline
  Butter & 5kg & 12,- DM \\
  ...
  \cline{2-3}
  \raisebox{1.5ex}[-1.5ex]{Äpfel}
         & 8kg & 40,- DM \\
  \hline \hline
  \multicolumn{2}{|l||}{Gesamt:}
         & 91,80 DM \\
  \hline
  \end{tabular}}
```

Abbildung 7.1: Kleine Tabelle in einem Rahmen

wie bei Miniseiten oder Absatzboxen die Ausrichtung der Tabelle zur umgebenen Textzeile an. Mögliche Werte sind: `t` für die Ausrichtung der obersten Tabellenzeile und `b` für die Ausrichtung der untersten Zeile auf die laufende Umgebung. Fehlt der Parameter, so wird die Tabelle vertikal zentriert. Der Parameter hat keine Wirkung, wenn die Tabelle zwischen zwei Absätzen steht.

Die Angaben für ***Deklaration*** bestimmen das Aussehen der einzelnen Spalten. Zur Verfügung stehen dabei `l`, `r`, `c` und `p{`*Breite*`}`. Dabei bedeutet `l` eine Spalte mit linksbündigem, `r` mit rechtbündigem und `c` mit zentriertem Text. Die letzte Möglichkeit `p{`*Breite*`}` definiert eine Spalte in der angegebenen Breite, die mehrzeiligen Text enthalten kann. Diese Deklaration erzeugt intern eine Absatzbox, die mit dem oberen Rand an der aktuellen Zeile ausgerichtet ist (`\parbox[t]{`*Breite*`}`). `|` legt einen senkrechten Strich als Spaltentrennung fest.

Selbstverständlich kann man die Spaltendeklarationen auch stärker verändern. Mit der Option `@{`*Text*`}` wird *Text* in jeder Zeile als Spaltenzwischenraum verwendet. Möchte man viele Spalten mit gleicher Deklaration verwenden, so kann man die Eingabe verkürzen, indem man die Anzahl der gleichen Spalten eingibt. Dazu schreibt man einfach `*{`*Anzahl*`}{`*Deklaration*`}`. Die angegebene ***Deklaration*** wird dann entsprechend der ***Anzahl*** wiederholt. Die Angabe `|l|r|l|r|l|r|l|r|l|r|` ist also gleichbedeutend mit `*{5}{|l|r}|`.

Verwendet man statt `tabular` die `tabular*` Umgebung, so kann man die ***Breite*** der gesamten Tabelle festlegen. Damit alle Spalten gleichmäßig verbreitert werden, sollte man `@{\extracolsep\fill}` in die Deklaration aufnehmen.

Innerhalb der `tabular`-Umgebung werden die einzelnen Spalten in der Eingabe durch ein `&` voneinander getrennt, `\\` bedeutet einen Zeilenumbruch[1]. Die einzelnen Spalteneinträge werden von LaTeX so behandelt, als ob sie in geschwungenen Klammern stünden. Dies hat zur Folge, daß z. B. Schriftbefehle ohne Klammern verwendet werden können. Mit Hilfe der in Kapitel 7.2 dargestellten Funktionen können die Schriftbefehle sogar in der Spaltendeklaration angegeben werden, sofern sie für eine gesamte Spalte gelten sollen.

Horizontale Linien in einer Tabelle erreicht man mit den beiden Befehlen

```
\hline
\cline{a-b}
```

Der obere Befehl erzeugt dabei eine Linie über die gesamte Tabellenbreite. Beim unteren Befehl reicht die Linie von Spalte ***a*** bis Spalte ***b*** (beides einschließlich). Beide Befehle dürfen nur entweder am Anfang der Tabelle oder direkt nach einem Zeilenumbruch (`\\`) auftreten, wobei beliebig viele der Befehle hintereinander folgen dürfen. Möchte man innerhalb einer Spalte eine vertikale Linie erzeugen, so geschieht dies mit dem Befehl

```
\vline
```

Einträge, die über mehrere Spalten reichen sollen, erhält man mit dem Befehl

```
\multicolumn{Anzahl}{Ausrichtung}{Inhalt}
```

Anzahl gibt die Zahl der Spalten an, die zusammengefaßt werden sollen, ***Ausrichtung*** legt das Format des Textes innerhalb der Spalte (rechts- oder linksbündig, zentriert; vertikale Linien) fest und ***Inhalt*** ist der Inhalt dieser Zelle.

7.1.1 Anpassen des Tabellenstils

Das Aussehen einer Tabelle wird von einer Reihe Parameter bestimmt, die der Anwender selbstverständlich ändern kann. Die Änderung sollte aber immer außerhalb der Tabellenumgebung erfolgen.

[1] Innerhalb einer Spalte kann man auf diese Weise also keinen Zeilenumbruch erzeugen. Mehrzeilige Einträge sind ohnehin nur bei der Option `p` erlaubt. Dort erreicht man einen Zeilenumbruch mit dem Befehl `\newline`.

Tabelle 7.1: Wichtige Maße zum Anpassen von Tabellen

`\tabcolsep`	bestimmt die halbe Breite zwischen zwei Spalten innerhalb einer `tabular` oder `tabular*` Umgebung.
`\arraycolsep`	bestimmt die halbe Breite zwischen zwei Spalten innerhalb einer `array` Umgebung.
`\arrayrulewidth`	legt die Dicke von vertikalen und horizontalen Strichen in einer Tabelle fest.
`\doublerulesep`	bestimmt den Abstand von Doppellinien.
`\arraystrech`	legt einen Faktor fest, mit dem der normale Zeilenabstand in einer Tabelle multipliziert wird. Standardmäßig beträgt er 1. Eine Änderung z. B. auf 1.5 erfolgt durch den Befehl `\renewcommand{\arraystrech}{1.5}` Damit wäre der Zeilenabstand in der Tabelle 1.5mal größer als außerhalb.

7.1.2 Beispiele für Tabellenkonstruktionen

Im Folgenden soll anhand eines immer wiederkehrenden Beispieles gezeigt werden, wie einfach die Erzeugung einer Tabelle mit LaTeX ist. Die Tabelle enthält sieben Spalten, davon sind die erste und letzte rechtsbündig, die zweite ist linksbündig, danach folgen vier zentrierte Spalten. Die Tabelle enthält keine horizontalen oder vertikalen Linien. Die Spaltendeklaration lautet demnach `{rlccccr}` und die Tabelle wird erzeugt mit:

Tabelle 7.2: Basketball Bundesliga, Beispiel 1

Platz	Verein	Sp	S	N	K	P
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S. Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14
5.	TV AXA DIREKT Langen	16	8	8	1357:1346	16:16
6.	USC Heidelberg	16	8	8	1443:1461	16:16
7.	TSV Breitengüßbach	16	8	8	1310:1265	16:16
8.	TSV Speyer	16	5	11	1249:1282	10:22
9.	BG Koblenz	15	4	11	1244:1352	8:22
10.	DJK Falke Nürnberg	15	2	13	1185:1386	4:26

```
\begin{tabular}{rlccccr}
Platz & Verein               & Sp & S  & N  & K         & P     \\[1pt]
1. & USC Freiburg            & 15 & 12 &  3 & 1336:1227 & 24:6  \\
2. & DJK S. Oliver Würzburg  & 16 & 12 &  4 & 1534:1369 & 24:8  \\
...
10. & DJK Falke Nürnberg     & 15 &  2 & 13 & 1185:1386 &  4:26
\end{tabular}
```

Die Angabe `[1pt]` am Ende der ersten Zeile bewirkt, daß ein zusätzlicher Zwischenraum von 1pt Größe zwischen der ersten und zweiten Zeile eingefügt wird. In der letzten Zeile kann das Zeilenendezeichen entfallen.

Im nächsten Schritt soll die Tabelle mit vertikalen und horizontalen Linien versehen werden. Die Spaltendeklaration ist entsprechend zu ergänzen und zwischen den Zeilen müssen die Linien mit `\hline` erzeugt werden:

```
\begin{tabular}{|r|l||c|c|c|c|r|}
\hline
Platz & Verein               & Sp & S  & N  & K         & P     \\[1pt]
\hline\hline
1. & USC Freiburg            & 15 & 12 &  3 & 1336:1227 & 24:6  \\ \hline
2. & DJK S. Oliver Würzburg  & 16 & 12 &  4 & 1534:1369 & 24:8  \\ \hline
...
10. & DJK Falke Nürnberg     & 15 &  2 & 13 & 1185:1386 &  4:26 \\ \hline
\end{tabular}
```

In diesem Fall muß am Ende der letzten Zeile noch ein Zeilenumbruch stehen, da unter der Zeile ja noch eine Linie folgt.

Tabelle 7.3: Basketball Bundesliga, Beispiel 2

Platz	Verein	Sp	S	N	K	P
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S. Oliver Würzburg	16	12	4	1534:1369	24:8
	⋮					
10.	DJK Falke Nürnberg	15	2	13	1185:1386	4:26

Im letzten Schritt soll die Tabelle eine Überschrift bekommen, die über alle Spalten reicht. Außerdem soll eine weitere Spalte eingeführt werden, deren Zellen über mehrere Zeilen reichen. Schließlich werden die beiden Spalten, die bislang das Korb- und das Punkteverhältnis enthielten, aufgeteilt und der Doppelpunkt als Spaltentrennung eingesetzt. Die erste Zeile wird mit einem `\multicolumn` Befehl erzeugt.

```
\fbox{\begin{tabular}{|r|l||c|c|c|r@{:}l|r@{:}l|c|}
\hline
\multicolumn{10}{|c|}{2. Basketball Bundesliga} \\ \hline
Platz & Verein  & Sp & S & N & \multicolumn{2}{c|}{Körbe}
                     & \multicolumn{2}{c|}{Punkte} & \\ \hline\hline
1.  & USC Freiburg       & 15 & 12 &  3 & 1336 & 1227 & 24 & 6 & \\ \cline{1-7}
...
10. & DJK Falke Nürnberg & 15 &  2 & 13 & 1185 & 1386
                                          &  4 & 26 & \rz{runde} \\ \hline
\end{tabular}}
```

Das Hochsetzen der Einträge innerhalb der Zellen erfolgt mit den in Abschnitt 4.7.5 auf Seite 70 vorgestellten `\raisebox` Befehlen. Zur Vereinfachung wurde hierfür der Befehl `\rz` definiert:
`\newcommand{\rz}[1]{\raisebox{1.5ex}[-15.ex]{#1}}`

Tabelle 7.4: Basketball Bundesliga, Beispiel 3

2. Basketball Bundesliga							
Platz	Verein	Sp	S	N	Körbe	Punkte	
1.	USC Freiburg	15	12	3	1336:1227	24:6	Aufstiegs- runde
2.	DJK S. Oliver Würzburg	16	12	4	1534:1369	24:8	
3.	TV Lich	14	10	4	1243:1369	20:8	
4.	SV03 Tübingen	15	8	7	1292:1322	16:14	
5.	TV AXA DIREKT Langen	16	8	8	1357:1346	16:16	
6.	USC Heidelberg	16	8	8	1443:1461	16:16	
7.	TSV Breitengüßbach	16	8	8	1310:1265	16:16	
8.	TSV Speyer	16	5	11	1249:1282	10:22	Abstiegs- runde
9.	BG Koblenz	15	4	11	1244:1352	8:22	
10.	DJK Falke Nürnberg	15	2	13	1185:1386	4:26	

7.2 Erweiterte Tabellenformen

Möchte man mit den Grundfunktionen der `tabular` Umgebung kompliziertere Tabellen erzeugen, bei denen man größeren Einfluß z. B. auf die Spaltendeklaration ausüben möchte, so wird man recht schnell die Grenzen dieser Umgebung erkennen. Mit Hilfe verschiedener Zusatzpakete stehen hier jedoch eine Reihe umfangreicher Funktionen zur Verfügung.

7.2.1 `array`

Das umfangreichste Zusatzpaket für Tabellen stellt das `array` Paket von Frank Mittelbach und David Carlisle dar (Mittelbach & Carlisle, 1998). Es bietet zunächst eine Reihe weiterer oder geänderter Spaltenoptionen (siehe Tabelle 7.5). Anhand der neuen Optionen ist es zum Beispiel möglich, zusätzlichen Text vor oder nach jedem Eintrag einer Spalte einzusetzen, außerdem kann der Zwischenraum zwischen den Spalten genauer bestimmt werden.

Tabelle 7.5: Spaltenoptionen des `array` Paketes

<table>
<tr><td><code>|</code></td><td>Erzeugt eine vertikale Linie zwischen zwei Spalten, wobei der Spaltenabstand um die Breite der Linie vergrößert wird.</td></tr>
<tr><td><code>m{</code>Breite<code>}</code></td><td>Erzeugt eine Spalte mit der angegebenen Breite, deren Inhalt vertikal zentriert ist (<code>\parbox{</code>Breite<code>}</code>).</td></tr>
<tr><td><code>b{</code>Breite<code>}</code></td><td>entspricht <code>\parbox[b]{</code>Breite<code>}</code>.</td></tr>
<tr><td><code>>{</code>Dekl.<code>}</code></td><td>Kann vor den Spaltenoptionen verwendet werden und fügt Dekl. direkt vor jedem Eintrag ein.</td></tr>
<tr><td><code><{</code>Dekl.<code>}</code></td><td>fügt Dekl. direkt hinter jedem Eintrag ein.</td></tr>
<tr><td><code>!{</code>Dekl.<code>}</code></td><td>ähnlich wie <code>|</code>, allerdings wird Dekl. anstelle der Linie eingesetzt. Anders als bei <code>@{</code>Dekl.<code>}</code> wird der normale Spaltenabstand nicht unterdrückt.</td></tr>
</table>

Die Optionen `>` und `<` wurden ursprünglich entwickelt, um auf einfache Weise den Text einer Spalte automatisch im mathematischen Modus zu setzen. Man muß nun nicht mehr in jeder Zeile die Umschaltbefehle verwenden. Die Spaltendeklaration `>{$}c<{$}` setzt automatisch den Text der Spalte zentriert im mathematischen Modus. Man kann die Optionen aber auch verwenden, um in einer Spalte die Schriftart umzustellen. Mit `>{\bfseries}` sorgt man dafür, daß der Text dieser Spalte fett gedruckt wird.

Das `array` Paket führt einen weiteren Parameter ein, mit dem das generelle Aussehen der Tabelle beeinflußt werden kann. Der Wert `\extrarowheight` wird zur normalen Höhe einer Tabellenzeile addiert, ohne die Tiefe der Zeile zu beeinflussen. Er ist besonders wichtig bei Tabellen mit horizontalen Linien, da hier der Abstand oberhalb des Textes oft zu eng ist.

7.2.2 tabularx

Das `tabularx` Paket stammt von David Carlisle. Es bietet eine Weiterentwicklung der `tabular*` Umgebung. In beiden Fällen kann die Gesamtbreite der Tabelle festgelegt werden. Bei der `tabularx` Umgebung kann man jedoch recht einfach festlegen, welche Spalten verbreitert werden sollen. Hierzu dient die Spaltenoption `X`, die der Option `p` entspricht. Die Breite wird jedoch nicht angegeben, sondern automatisch berechnet.

Normalerweise sind die X-Spalten in einer Tabelle immer gleich breit. Ist dies nicht erwünscht, so kann man das Verhältnis der Spalten zueinander in der Präambel der Tabelle angeben. Eine Tabelle mit zwei breiten und einer schmalen Spalte könnte z. B. so aussehen:

```
|>{\setlenght{\hsize}{0.6\hsize}}X|
  >{\setlength{\hsize}{1.2\hsize}}X|
  >{\setlength{\hsize}{1.2\hsize}}X|
```

Dies erzeugt eine Tabelle, in der die erste Spalte genau halb so breit ist wie die beiden anderen Spalten.

Tabelle 7.6: Beispiel einer tabularx Umgebung

Dies ist die schmale	Hier die zweite Zelle. Sie ist doppelt so groß wie die erste.	Und hier die dritte und letzte Zelle. Die Tabelle reicht insgesamt über die ganze Zeilenbreite.

Bei der Verwendung dieser Modifikationen sind ein paar Dinge zu beachten. `\hsize` bezeichnet die jeweilige Breite der Spalten. Die Summe der Spaltenbreite sollte konstant bleiben. In dem oberen Fall muß die Summe der drei Spalten also weiterhin das Dreifache der von LaTeX 2_ε bestimmten Spaltenbreite betragen. Es geht also nicht darum, das Verhältnis der Breiten zueinander anzugeben, sondern die genaue Breite der Spalten. In dem obigen Beispiel werden die beiden breiten Spalten jeweils 20% breiter, daher muß die linke Spalte um insgesamt 40% schmaler werden.

Darüberhinaus sollte man möglichst keinen `\multicolumn` Befehl über modifizierte Spalten reichen lassen.

Die `tabularx` Umgebung unterscheidet sich in einigen Punkten von der `tabular*` Umgebung:

- Die `tabular*` Umgebung ändert die Spalten*abstände*, um die Breite der Tabelle zu erreichen, `tabularx` dagegen ändert die Spalten*breite*.

- Der Inhalt der `tabularx` Umgebung entspricht einem Befehlsargument. Dies führt zu bestimmten Einschränkungen. `\verb` und `\verb*` Befehle können zwar verwendet werden, jedoch kann es sein, daß Leerzeichen falsch verarbeitet werden. Außerdem dürfen keine Kommentare in der Tabelle erscheinen.

- Normale `tabular` oder `tabular*` Umgebungen können geschachtelt werden. Möchte man eine `tabularx` Umgebung schachteln, so muß sie in Klammern `{...}` eingeschlossen werden.

- Die `tabularx` Umgebung wird intern mehrfach formatiert, bis die optimale Spaltenbreite gefunden ist. Dieser Vorgang ist langsamer als die normalen Tabellen. Außerdem kann das mehrmalige Formatieren manche TEX Strukturen beschädigen.

7.2.3 Zeilenumbruch in schmalen Spalten

Besonders bei schmalen Spalten kann es vorkommen, daß der Text nicht vernünftig umgebrochen wird. Oftmals ist es besser, den Text in Flattersatz zu setzen. Fügt man nun in der Spaltendeklaration `>{\raggedright}` ein, so wird man unweigerlich eine Reihe zunächst unverständlicher Fehlermeldungen erhalten. Dies liegt daran, daß die Befehle `\raggedright`, `\raggedleft` und `\centering` den Zeilenumbruch (`\\`) neu definieren, was jedoch innerhalb einer Tabelle nicht erlaubt ist. Bei Verwendung des `tabularx` Paketes kann man dies bei der Spaltendeklaration mit dem Befehl `\arraybackslash` verhindern. `>{\raggedright\arraybackslash}X` führt zu einer korrekten Bearbeitung der Tabelle. Verwendet man das Paket nicht, so kann man das Problem z. B. mit folgenden Befehlen beheben:

```
\newcomand{\PBS}[1]{\let\temp=\\#1\let\\=\temp}
\begin{tabular}{|>{\PBS\raggedright\hspace{0pt}}p{45mm}|}
```

Hier wird eine Spalte von 45mm Breite erzeugt, deren Textinhalt linksbündig gesetzt wird. Ein manueller Zeilenumbruch innerhalb der Spalte ist aber weiterhin nur mit `\newline` möglich!

7.2.4 Definition neuer Spaltenoptionen

Wenn man in einer Tabelle nur eine einzelne Spalte mit zusätzlichen Deklarationen verwendet, so ist es meist das einfachste, die Änderungen direkt in der Tabellenpräambel festzulegen. Tritt dagegen eine bestimmte Spaltendeklaration in einer Tabelle oder sogar in einem Dokument häufiger auf, so empfiehlt es sich, eine neue Spaltenoption zu definieren. Dies geschieht mit dem Befehl

```
\newcolumntype{Name}[Num]{Deklaration}
```

Name bezeichnet einen *einzelnen* Buchstabe, der als Bezeichnung für den neuen Spaltentyp dient. ***Num*** legt die Anzahl der Argumente fest, die der Spaltenoption übergeben wird. ***Deklaration*** schließlich ist die eigentliche Deklaration der neuen Spalte. Hierbei können alle Befehle und Optionen verwendet werden, die auch normal in der Tabellendeklaration verwendet werden dürfen. Verwendet man in vielen Tabellen z. B. Spalten mit fettgedrucktem Text, so kann man einen entsprechenden Spaltentyp definieren:

```
\newcolumntype{B}{>{\bfseries}c}
```

Man kann aber nicht nur neue Spaltentypen erzeugen, sondern auch mehrere Spaltendeklarationen zusammenfassen.

```
\newcolumntype{M}{llr}
\begin{tabular}{M}
  ...
\end{tabular}
```

Hierbei wird der neue Spaltentyp `M` durch die Deklaration `llr` ersetzt. Besonders bei mehreren Tabellen mit jeweils gleicher Spaltenform kann man damit die Eingabe vereinfachen. Möchte man eine Liste aller bekannten neuen Spaltentypen bekommen, so verwendet man in der Präambel den Befehl

```
\showcols
```

7.2.5 `dcolumn`

Betrachtet man die Tabelle in Abbildung 7.1 auf Seite 100, so fällt auf, daß die Preise in der rechten Spalte nicht korrekt untereinander stehen. Der Text der Spalte ist einfach zentriert. Ein ähnliches Problem stellt sich bei der Spalte mit den Punkten in Tabelle 7.3. Hier wäre es besser, wenn die Doppelpunkte jeweils untereinander ständen. In Tabelle 7.4 wurde dies zum Teil erreicht, in dem der Doppelpunkt als Spaltentrennung eingefügt wurde. Man sieht jedoch, daß auch dieses Ergebnis nicht voll befriedigend ist.

Bessere Ergebnisse kann man mit dem `dcolumn` Paket von David Carlisle erreichen. Hiermit ist es möglich, Spalteneinträge am Dezimalpunkt oder einem anderen Zeichen zu orientieren. Das Paket definiert die neue Spaltenoption `D`, die drei Argumente benötigt:

`D{`*`Eingabe-Trenn`*`}{`*`Ausgabe-Trenn`*`}{`*`Kommastellen`*`}`

Mit ***`Eingabe-Trenn`*** wird das Trenzeichen bei der Eingabe der Daten festgelegt, es darf in jeder Zelle nur einmal verwendet werden. ***`Ausgabe-Trenn`*** bestimmt, welches Zeichen bei der Ausgabe verwendet wird. Dieses Zeichen kann das gleiche wie bei der Eingabe sein. Es ist aber auch jedes andere mathematische Zeichen zugelassen. ***`Kommastellen`*** schließlich bestimmt die Anzahl der Nachkommastellen, für die in der Spalte Platz gelassen wird. Werden mehr Stellen eingegeben als in der Präambel der Tabelle definiert, so reichen sie über den Spaltenrand hinaus.

Tabelle		
Butter	5 kg	12,– DM
Milch	3 l	7,80 DM
Äpfel	4 kg	32,– DM
	8 kg	40,– DM
Gesamt:		91,80 DM

```
\fbox{\begin{tabular}
  {|c|r@{\,}l||D{,}{,}{2}@{\,}l|}
\hline
\multicolumn{5}{|c|}{Tabelle} \\
\hline \hline
Butter & 5 & kg & 12,- & DM \\
...
\end{tabular}}
```

Das Beispiel zeigt nun die veränderte Preisliste. Außer der Ausrichtung am Komma wurde in der Deklaration noch eine weitere Spalte eingefügt, in der die Währung steht. Dies wurde getan, da eine mit der Option `D` erzeugte Spalte im mathematischen Modus ist und der Text `DM` andernfalls in einer anderen Schrift auftreten würde.

7.2.6 `hhline`

Falls die Möglichkeiten, horizontale oder vertikale Linien in einer Tabelle zu verwenden, nicht ausreichen, kann man das `hhline` Paket verwenden. Das Paket definiert den Befehl

`\hline{`*`Deklarationen`*`}`

mit dem das Aussehen der Linie sehr leicht festgelegt werden kann.

`Deklaration` besteht dabei aus einer Liste der folgenden Zeichen:

`-`	eine einfache `\hline` von der Breite der Spalte
`=`	eine doppelte `\hline` von der Breite der Spalte
`~`	ein Leerraum von der Breite einer Spalte
`|`	eine `\vline`, die eventuelle horizontale Linien durchkreuzt
`:`	eine `\vline`, die von einer `\hline` durchbrochen wird
`#`	Segment einer doppelten `\hline` und einer doppelten `\vline`
`t`	obere Linie einer doppelten `\hline`
`b`	untere Linie einer doppelten `\hline`
`*`	ähnlich der Spaltenoption `*`: `*{4}{#=}` wird zu `#=#=#=#=` ersetzt

Die beiden Zeichen `t` und `b` können verwendet werden, um die Ecken einzelner Kästchen zu erzeugen. Die horizontalen Linien werden normalerweise durch vertikale Doppellinien unterbrochen. Die einzelnen Funktionen werden am Beispiel der Preisliste kurz dargestellt.

Tabelle		
Butter	5kg	12,– DM
Milch	3l	7,80 DM
Äpfel	4kg	32,– DM
	8kg	40,– DM
Gesamt:		91,80 DM

```
\begin{tabular}{...}
\hhline{|t:====:t|}
\multicolumn ...\\\hhline{|:==:t:==:|}
Butter ...\\\hhline{||-|-||--||}
Milch  ...\\\hhline{||-|-||--||}
 & 4kg ...\\\hhline{||~|-||--||}
... & 8kg ...\\\hhline{|:==::==:|}
... & 91,80 ...\\\hhline{|b:==:b:==:b|}
\end{tabular}
```

Während bei einer normalen `\hline` die Linie intern von einem `\hrule` Befehl erzeugt wird, wird bei einer `\hhline` die Linie aus vielen einzelnen Elementen zusammengesetzt. Die Stücke werden in der `dvi` Datei exakt aneinander gesetzt, jedoch kann es bei verschiedenen Druckertreibern oder Previewern zu Problemen kommen. In diesem Fall sollte das Problem vermindert werden, wenn man die Strichstärke ein wenig erhöht.

7.2.7 Abschließendes Beispiel

Zum Schluß soll noch einmal das Beispiel der Basketballtabelle aufgegriffen werden. Diesmal werden jedoch die in den vorherigen Abschnitten beschriebenen Werkzeuge einbezogen.

Zunächst werden drei neue Spaltentypen und ein weiterer Befehl definiert. Anschließend folgt die Beschreibung der eigentlichen Tabelle.

```
\newcolumntype{K}{D{:}{:}{4}}
\newcolumntype{P}{D{:}{:}{2}}
\newcolumntype{S}{D{.}{.}{0}}
\newcommand{\sptext}[1]{\multicolumn{1}{c|}{#1}}

\fbox{\begin{tabular}{|r|l||S|S|S|K|P|c|}
  \hline
  \multicolumn{8}{|c|}{2. Basketball Bundesliga} \\ \hline
  Platz & Verein  & \sptext{Sp} & \sptext{S} & \sptext{N} &
                      \sptext{Körbe} & \sptext{Punkte} & \\ \hline\hline
  1.  & USC Freiburg  & 15 & 12 &  3 & 1336:1227 & 24:6  & \\ \cline{1-7}
  ...
\end{tabular}}
```

Tabelle 7.7: Basketball Bundesliga, Beispiel 4

2. Basketball Bundesliga							
Platz	Verein	Sp	S	N	Körbe	Punkte	
1.	USC Freiburg	15	12	3	1336:1227	24:6	Aufstiegs-runde
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8	
3.	TV Lich	14	10	4	1243:1369	20:8	
4.	SV03 Tübingen	15	8	7	1292:1322	16:14	
5.	TV AXA DIREKT Langen	16	8	8	1357:1346	16:16	
6.	USC Heidelberg	16	8	8	1443:1461	16:16	
7.	TSV Breitengüßbach	16	8	8	1310:1265	16:16	
8.	TSV Speyer	16	5	11	1249:1282	10:22	Abstiegs-runde
9.	BG Koblenz	15	4	11	1244:1352	8:22	
10.	DJK Falke Nürnberg	15	2	13	1185:1386	4:26	

7.3 Farbige Tabellen

In vielen Fällen möchte man bestimmte Spalten, Zeilen oder sogar einzelne Zellen farbig hinterlegen. Diese Aufgabe ist jedoch nicht gerade trivial. Es reicht dabei in den meisten Fällen nicht aus, einfach nur eine farbige Box in einer Zelle zu erzeugen. Werden die Spalten und Zeilen durch Striche getrennt, so würde eine farbige Box in einer Zelle nicht bis zu diesen Strichen reichen. Das `colortbl`-Paket von David Carlisle bietet hier Abhilfe (Carlisle, 1998a).

Das Paket stellt für das Einfärben von Zeilen den Befehl

`\columncolor[`***`Modell`***`]{`***`Farbe`***`}[`***`links`***`][`***`rechts`***`]`

zur Verfügung. Der Befehl darf *nur* in der Tabellendefinition bzw. in der Deklaration eines neuen Spaltetyps in einem >-Ausdruck verwendet werden (vgl. Tabelle 7.5 auf Seite 105). ***`Farbe`*** definiert die Farbe der Zelle. Dazu muß entweder vorher der Farbname mit dem Befehl `\definecolor` aus dem `color`-Paket erzeugt worden sein (vgl. Abschnitt 9.4.1 auf Seite 171), oder die Farbe wird direkt definiert. Dazu ist der Parameter ***`Modell`*** nötig, der das verwendete Farbmodell festlegt. Die Farbe selbst wird dann als Zahl bzw. Zahlenkombination angegeben. Die beiden optionalen Parameter ***`links`*** und ***`rechts`*** geben an, wie weit die Farbe nach links und recht über den Tabelleninhalt hinausreichen soll. Werden die beiden Werte weggelassen, so nimmt die Farbe den gesamten Raum der Spalte ein.

```
\definecolor{hell}{gray}{0.9}
\definecolor{mittel}{gray}{0.75}
\definecolor{dunkel}{gray}{0.55}
\begin{tabular}{|r|l|S|S|S|>{\columncolor{hell}}K|>{\columncolor{mittel}}P|}
  \hline
  \multicolumn{7}{|>{\columncolor{dunkel}}c|}
                {\Large\rule{0mm}{5mm}2. Basketball Bundesliga} \\
  Platz & Verein  & \multicolumn{1}{c|}{Sp} & \multicolumn{1}{c|}{S} &
             \multicolumn{1}{c|}{N} &
             \multicolumn{1}{>{\columncolor{hell}}c|}{Körbe} &
             \multicolumn{1}{>{\columncolor{mittel}}c|}{Punkte}  \\
  1.  & USC Freiburg           & 15 & 12 &  3 & 1336:1227 & 24:6  \\
  2.  & DJK S: Oliver Würzburg & 16 & 12 &  4 & 1534:1369 & 24:8  \\
  3.  & TV Lich                & 14 & 10 &  4 & 1243:1369 & 20:8  \\
  4.  & SV03 Tübingen          & 15 &  8 &  7 & 1292:1322 & 16:14 \\
\end{tabular}
```

Das Beispiel in Tabelle 7.8 zeigt die bereits bekannte Basketball-Tabelle. Die letzte Spalte wurde weggelassen, ebenfalls einige Striche.[2] Die verwendeten Spaltendeklarationen sind bereits aus dem vorherigen Beispiel auf Seite 111 bekannt.

Tabelle 7.8: farbige Tabelle, Beispiel 1

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Die Bedeutung der beiden Parameter *links* und *rechts* wird im Vergleich zu der nächsten Tabelle deutlich. Hierfür wurde eine weitere Spalte eingefärbt, allerdings so, daß die Farbe jeweils 1pt weiter nach links und rechts reicht als der breiteste Text in der Spalte. Die Tabellendeklaration lautete also:

```
\begin{tabular}
  {|r|l|S|>{\columncolor{hell}[1pt][1pt]}S|S|
    >{\columncolor{hell}}K|>{\columncolor{mittel}}P|}
```

Tabelle 7.9: farbige Tabelle, Beispiel 2

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Es können aber nicht nur Spalten eingefärbt werden, sondern auch Zeilen. Hierfür bietet das `colortbl`-Paket den Befehl

`\rowcolor[`*`Modell`*`]{`*`Farbe`*`}[`*`links`*`][`*`rechts`*`]`

[2] Bei der Verwendung des `colortbl`-Paketes kann es zu Problemen mit doppelten Strichen in Tabellen kommen. Daher sollte man diese Kombination vermeiden.

Der Befehl färbt die gesamte Zeile, in der er steht, in der angegebenen Farbe ein. Er hat dabei Vorrang vor eventuellen `\columncolor`-Befehlen. Werden in der Tabelle also bereits einzelne Spalten eingefärbt, so wird diese Angabe in einer `\rowcolor`-Zeile ignoriert.

```
\begin{tabular}{|r|l|S|S|S|K|P|}
  \hline
  \multicolumn{7}{|>{\columncolor{dunkel}}c|}
                  {\Large\rule{0mm}{5mm}2. Basketball Bundesliga} \\
  Platz & Verein  & \sptext{Sp} & \sptext{S} & \sptext{N} &
     \sptext{Körbe} & \sptext{Punkte}  \\
  \rowcolor{mittel}
    1.  & USC Freiburg          & 15 & 12 &  3 & 1336:1227 & 24:6  \\
    2.  & DJK S: Oliver Würzburg & 16 & 12 &  4 & 1534:1369 & 24:8  \\
  \rowcolor{mittel}
    3.  & TV Lich               & 14 & 10 &  4 & 1243:1369 & 20:8  \\
    4.  & SV03 Tübingen         & 15 &  8 &  7 & 1292:1322 & 16:14 \\ \hline
\end{tabular}
```

Tabelle 7.10: farbige Tabelle, Beispiel 3

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Wie im Beispiel von Tabelle 7.10 erkennbar, muß das Einfärben der Titelzeile immer nocht mit einem `\columncolor`-Befehl in der Deklaration der `\multicolumn` erfolgen.

Das `colortbl`-Paket stellt keinen direkten Befehl zur Verfügung, um nur eine einzelne Zelle einzufärben. Mit einer Kombination der Befehle `\multicolumn` und `\columncolor` ist jedoch auch dieses problemlos lösbar. Die folgende Zeile definiert einen neuen Befehl `\FZ`, mit dem auch einzelne Zellen eingefärbt werden können:

```
\newcommand{\FZ}[2]{\multicolumn{1}{>{\columncolor{#1}}l}{#2}}
```

Der Befehl erhält zwei Parameter. Der erste enthält die Farbe der Zelle, der zweite den eigentlichen Zelleninhalt, der in diesem Fall in der Zelle linksbündig steht. Selbstverständlich wären auch alle anderen Formatierungen möglich.

```
\colorbox{hell}{
  \begin{tabular}{rlSSSKP}
    \multicolumn{7}{c}{\Large\rule{0mm}{5mm}2. Basketball Bundesliga}  \\[1pt]
    Platz & Verein  & \multicolumn{1}{c}{Sp} &
      \multicolumn{1}{c}{S} & \multicolumn{1}{c}{N} &
      \multicolumn{1}{c}{Körbe} & \multicolumn{1}{c}{Punkte}           \\
    1.  & \FZ{white}{USC Freiburg} & 15 & 12 &  3 & 1336:1227 & 24:6   \\
    2.  & DJK S: Oliver Würzburg   & 16 & 12 &  4 & 1534:1369 & 24:8   \\
    3.  & \FZ{white}{TV Lich}      & 14 & 10 &  4 & 1243:1369 & 20:8   \\
    4.  & SV03 Tübingen            & 15 &  8 &  7 & 1292:1322 & 16:14  \\
  \end{tabular}}
```

Um die gesamte Tabelle in Beispiel 7.11 einzufärben, wurde sie in eine `\colorbox` gesetzt. Die beiden farbigen Zelle wurden anschließend mit dem zuvor definierten Befehl `\FZ` eingefärbt.

Tabelle 7.11: farbige Tabelle, Beispiel 4

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Ds `colortbl`-Paket stellt nicht nur Funktionen bereit, um Spalten oder Zeilen einer Tabelle einzufärben. Auch farbige Striche können mit Hilfe des Paketes realisiert werden. Der Befehl

`\arrayrulecolor[`*`Modell`*`]{`*`Farbe`*`}`

legt die Farbe aller Striche (`\cline`, `\hline` und `\vline`) in der Tabelle fest. Der Befehl wirkt dabei von seinen ersten Auftraten an auf alle nachfolgenden Striche. Steht der Befehl also *vor* der Tabelle, so gilt er für alle Striche der Tabelle, tritt er *innerhalb* der Tabelle auf, so wirkt er nur auf die noch folgenden Striche. Dabei ist zu beachten, daß die vertikalen Striche *in der Tabellendeklaration* festgelegt werden. Sie haben also immer die Farbe, die vor Tabellendeklaration definiert ist. Man kann also den vertikalen Strichen nicht unterschiedliche Farben zuordnen.

Im folgenden Beispiel sollen die vertikalen Striche dunkel erscheinen, ebenso der oberste und der unterste horizontale Strich. Die übrigen horizontalen Striche sollen dagegen hell werden.

```
\setlength{\arrayrulewidth}{2pt}
\arrayrulecolor{dunkel}
\begin{tabular}{|r|l|S|S|S|K|P|}
  \hline
\arrayrulecolor{hell}
  \multicolumn{7}{|c|}{\Large\rule{0mm}{5mm}2. Basketball Bundesliga} \\[1pt]
  \hline
  Platz & Verein  & \multicolumn{1}{c|}{Sp} &
    \multicolumn{1}{c|}{S} & \multicolumn{1}{c|}{N} &
    \multicolumn{1}{c|}{Körbe} & \multicolumn{1}{c|}{Punkte}        \\ \hline
  1.  & USC Freiburg          & 15 & 12 &  3 & 1336:1227 & 24:6    \\ \hline
  2.  & DJK S: Oliver Würzburg & 16 & 12 &  4 & 1534:1369 & 24:8    \\ \hline
  3.  & TV Lich               & 14 & 10 &  4 & 1243:1369 & 20:8    \\ \hline
\arrayrulecolor{dunkel}
  4.  & SV03 Tübingen         & 15 &  8 &  7 & 1292:1322 & 16:14   \\ \hline
\end{tabular}
```

Tabelle 7.12: farbige Tabelle, Beispiel 5

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Das Beispiel aus Tabelle 7.12 zeigt noch einige Schwächen. Das Ändern der Strichfarbe wirkt auch auf die Spaltendeklaration in `\multicolumn`-Befehlen, was jedoch in diesem Beispiel nicht gewollt ist. Um auch diese vertikalen Striche in dunkler Farbe zu erhalten, muß man vor den `\multicolumn`-Befehlen wieder die dunkle Farbe einstellen. Das Ergebnis ist in Tabelle 7.13 zu sehen.

```
\arrayrulecolor{dunkel}
\begin{tabular}{|r|l|S|S|S|K|P|}
  \hline
  \multicolumn{7}{|c|}{\Large\rule{0mm}{5mm}2. Basketball Bundesliga} \\[1pt]
\arrayrulecolor{hell}
  \hline
\arrayrulecolor{dunkel}
  Platz & Verein  & \multicolumn{1}{c|}{Sp} &
    \multicolumn{1}{c|}{S} & \multicolumn{1}{c|}{N} &
    \multicolumn{1}{c|}{Körbe} & \multicolumn{1}{c|}{Punkte}        \\
\arrayrulecolor{hell} \hline
  1.  & USC Freiburg          & 15 & 12 &  3 & 1336:1227 & 24:6    \\ \hline
  2.  & DJK S: Oliver Würzburg & 16 & 12 &  4 & 1534:1369 & 24:8    \\ \hline
  3.  & TV Lich               & 14 & 10 &  4 & 1243:1369 & 20:8    \\ \hline
\arrayrulecolor{dunkel}
  4.  & SV03 Tübingen         & 15 &  8 &  7 & 1292:1322 & 16:14   \\ \hline
\end{tabular}
```

Tabelle 7.13: farbige Tabelle, Beispiel 6

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Auch diese Tabelle zeigt noch ein Problem. Die horizontalen Striche reichen auch in die vertikalen Striche am Rand der Tabelle hinein. Soll dieser Effekt verhindert werden, so dürfen die Linien nicht mit `\hline` erzeugt werden, sondern mit `\cline`. Diese Linien liegen immer nur zwischen den vertikalen Linien.

```
\arrayrulecolor{dunkel}
\begin{tabular}{|r|l|S|S|S|K|P|}
  \hline
  \multicolumn{7}{|c|}{\Large\rule{0mm}{5mm}2. Basketball Bundesliga} \\[1pt]
\arrayrulecolor{hell}
  \cline{1-7}
\arrayrulecolor{dunkel}
  Platz & Verein  & \multicolumn{1}{c|}{Sp} &
    \multicolumn{1}{c|}{S} & \multicolumn{1}{c|}{N} &
    \multicolumn{1}{c|}{Körbe} & \multicolumn{1}{c|}{Punkte}      \\
\arrayrulecolor{hell} \cline{1-7}
  1.  & USC Freiburg          & 15 & 12 & 3 & 1336:1227 & 24:6  \\ \cline{1-7}
  2.  & DJK S: Oliver Würzburg & 16 & 12 & 4 & 1534:1369 & 24:8  \\ \cline{1-7}
  3.  & TV Lich               & 14 & 10 & 4 & 1243:1369 & 20:8  \\ \cline{1-7}
\arrayrulecolor{dunkel}
  4.  & SV03 Tübingen         & 15 &  8 & 7 & 1292:1322 & 16:14 \\ \hline
\end{tabular}
```

Tabelle 7.14: farbige Tabelle, Beispiel 7

2. Basketball Bundesliga						
Platz	Verein	Sp	S	N	Körbe	Punkte
1.	USC Freiburg	15	12	3	1336:1227	24:6
2.	DJK S: Oliver Würzburg	16	12	4	1534:1369	24:8
3.	TV Lich	14	10	4	1243:1369	20:8
4.	SV03 Tübingen	15	8	7	1292:1322	16:14

Die Kombination von farbigen Linien und farbigen Zellen ist leider nur eingeschränkt möglich. Farbige Zellen ragen stets ein wenig in die Striche hinein, insbesondere, wenn diese verbreitert wurden. Verhindert werden kann dies nur mit Hilfe des `hhline`-Paketes.

An dieser Stelle soll anhand eines durchgehenden Beispiels die Bedeutung von Farbe und die unterschiedliche Wirkung farbiger Elemente in Tabellen erörtert werden. Weiterführende Überlegungen hierzu finden sich in „*The LaTeX Graphics Companion*“ (Goossens et al., 1997) und in „*Color for the Electronic Age*“ (White, 1990).

Zunächst einmal wird die Tabelle ohne jegliche farbige Elemente erzeugt.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\begin{tabular}{lrr}
  \multicolumn{3}{c}
    {\Large\bfseries Verkauf} \\
  Kunde   & Produkt 1 & Produkt 2 \\
  Kunde 1 & 15 &  9 \\
  Kunde 2 &  0 &  3 \\
  Kunde 3 &  7 & 11 \\ \cline{2-3}
  Gesamt  & 22 & 23
\end{tabular}
```

Als nächstes soll die gesamte Tabelle farbig hinterlegt werden. Zu diesem Zweck wird sie wie bereits die Tabelle in Beispiel 7.11 auf Seite 115 in eine `\colorbox` gesetzt.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\colorbox{dunkel}{\begin{tabular}{lrr}
  \multicolumn{3}{c}
    {\Large\bfseries Verkauf} \\
  Kunde   & Produkt 1 & Produkt 2 \\
  Kunde 1 & 15 &  9 \\
  Kunde 2 &  0 &  3 \\
  Kunde 3 &  7 & 11 \\ \cline{2-3}
  Gesamt  & 22 & 23
\end{tabular}}
```

Die hier verwendete Farbe läßt die Schrift nicht sehr gut erkennbar erscheinen, daher soll diese im nächsten Schritt weiß werden.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\colorbox{dunkel}{\color{white}
 \begin{tabular}{lrr}
  \multicolumn{3}{c}
    {\Large\bfseries Verkauf} \\
  Kunde   & Produkt 1 & Produkt 2 \\
  Kunde 1 & 15 &  9 \\
  Kunde 2 &  0 &  3 \\
  Kunde 3 &  7 & 11 \\ \cline{2-3}
  Gesamt  & 22 & 23
\end{tabular}}
```

Wie man sieht, wirkt der Befehl `\color` nicht auf die mit `\cline` erzeugt Linie. Nun wirkt die Schrift noch ein wenig dünn. Daher wird sie als nächstes in der gesamten Tabelle fett geschrieben.

Verkauf		
Kunde	**Produkt 1**	**Produkt 2**
Kunde 1	**15**	**9**
Kunde 2	**0**	**3**
Kunde 3	**7**	**11**
Gesamt	**22**	**23**

```
\colorbox{dunkel}{\color{white}
 \bfseries
 \begin{tabular}{lrr}
   \multicolumn{3}{c}
   {\Large Verkauf} \\
   Kunde   & Produkt 1 & Produkt 2 \\
   Kunde 1 & 15 &  9 \\
   Kunde 2 &  0 &  3 \\
   Kunde 3 &  7 & 11 \\ \cline{2-3}
   Gesamt  & 22 & 23
 \end{tabular}}
```

Auch wenn die Tabelle farbig ist, wird die Farbe bislang nicht eingesetzt, um einzelne Teile der Tabelle hervorzuheben. Die Farbe ist also noch keine funktionelles Element, mit dem die Aufmerksamkeit des Lesers auf bestimmte Teile gerichtet wird, um so die Aussage der Tabelle zu unterstützen.

Im nächsten Beispiel sollen daher wichtige Teile der Tabelle gezielt eingefärbt werden. Der farbige Hintergrund wird hierfür zunächst weggelassen.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\begin{tabular}{lrr}
  \multicolumn{3}{c}{\color{mittel}
    \Large Verkauf} \\
  Kunde   & Produkt 1 & Produkt 2 \\
  Kunde 1 & 15 &  9 \\
  Kunde 2 &  0 &  3 \\
  Kunde 3 &  7 & 11 \\ \cline{2-3}
  \color{mittel}Gesamt &
    \color{mittel}22 &
      \color{mittel}23
\end{tabular}
```

Man könnte alternativ auch die weniger wichtigen Teile der Tabelle einfärben und somit die wichtigen durch die Farbe schwarz hervorheben.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\color{mittel}
\begin{tabular}{lrr}
  \multicolumn{3}{c}{\color{black}
    \Large Verkauf} \\
  ...
  \color{black}Gesamt &
    \color{black}22 &
      \color{black}23
\end{tabular}
```

In den bisherigen Beispielen wurde jeweils nur Teile des Textes eingefärbt. Im nächsten Schritt soll dagegen zur Hervorhebung ein Teil der Tabelle farbig hinterlegt werden.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\begin{tabular}{lrr}
  \multicolumn{3}{c}
    {\Large Verkauf} \\
  ...
  \rowcolor{mittel}
  Gesamt  & 22 & 23
\end{tabular}
```

Auch hier kann man statt die wichtigen Teile zu hinterlegen, diese als einzige weiß lassen. Dies verstärkt die Hervorhebung oft sogar noch, gerade bei dunkleren Hintergrundfarben.

Verkauf		
Kunde	Produkt 1	Produkt 2
Kunde 1	15	9
Kunde 2	0	3
Kunde 3	7	11
Gesamt	22	23

```
\colorbox{mittel}{%
  \begin{tabular}{lrr}
    \multicolumn{3}{c}
      {\Large Verkauf} \\
    Kunde  & Produkt 1 & Produkt 2 \\
    ...
    \rowcolor{white}
    Gesamt & 22 & 23
  \end{tabular}}
```

7.4 Mehrseitige Tabellen

LaTeX 2_ε setzt intern jede Tabelle in eine Box. Dies hat zur Folge, daß innerhalb einer Tabelle kein Seitenumbruch erfolgen kann. Ist eine Tabelle zu groß für den restlichen Platz auf einer Seite, so wandert sie komplett auf die nächste Seite. Spätestens wenn man lange Datensätze hat, die in einer Tabelle gesetzt werden sollen, bekommt man Probleme. Sind die Daten recht einfach, so könnte man die `tabbing` Umgebung verwenden, die einen Seitenumbruch zuläßt. Komfortabler geschieht dies jedoch mit den beiden Paketen `longtable` und `supertabular`, die im folgenden beschrieben werden.

7.4.1 supertabular

Das `supertabular` Paket stammt von Theo Jurriens und Johannes Braams. Es stellt die Umgebung `supertabular` bereit, die ganz ähnlich zur normalen `tabular` Umgebung verwendet wird. Intern wird die Tabelle auch in mehrere `tabular` Umgebungen aufgeteilt. `supertabular` berechnet bei jedem `\\` Befehl die benötigte Höhe. Ist der Wert von `\textheight` erreicht, so wird der Befehl `\end{tabular}` eingefügt und die Tabelle auf der nächsten Seite fortgesetzt. Der Tabellenkopf wird dabei übernommen. Da dabei die Tabelle aber intern in mehrere aufgeteilt wird, kann es passieren, daß die einzelnen Teile unterschiedlich breit sind. Dies kann man nur beheben, indem man bei der Spaltendeklaration allen Spalten eine Breite zuweist.

Die Deklaration mehrseitiger Tabellen ist etwas aufwendiger, da der Kopf und der Fuß er einzelnenen Seiten festgelegt werden muß. Hierzu dienen die Befehle aus Tabelle 7.15 auf der nächsten Seite.[3]

Die `supertabular` Umgebung kann genauso wie eine normale `tabular` Umgebung verwendet werden. Es können sämtliche Spaltenoptionen gewählt werden. Nur die optionalen Positionsparameter `t` und `b` sind verständlicherweise wirkungslos.

Es ist sogar möglich, die Gesamtbreite der Tabelle anzugeben. Hierzu verwendet man die `supertabular*` Umgebung. In diesem Fall hat zwar die Tabelle auf allen Seiten die gleiche Breite, das bedeutet aber nicht, daß auch die einzelnen Spalten überall die gleiche Breite haben.

Bei der Vewendung der `supertabular` Umgebung kann es ein paar Schwierigkeiten geben. Falls auf der ersten Seite der Tabelle ein Gleitobjekt erscheint, kann es passieren, daß die Tabelle auf die nächste Seite rutscht. Eventuell noch nicht platzierte Gleitobjekte werden ebenfalls noch vor der Tabelle gesetzt. Au-

[3] Da LaTeX intern in jedem Fall einen Tabellenfuß setzt, *muß* die letzte Zeile mit einem Zeilenende (`\\`) beendet werden. Anderfalls werden Fehler ausgegeben.

Tabelle 7.15: Übersicht über die Parameter der `supertabular` Umgebung

`\tablehead{...}`	bestimmt den Kopf der einzelnen Tabellenseiten. Das Argument kann Text, Linien (`\hline`) und Zeilenumbrüche (`\\`) enthalten.
`\tablefirsthead{...}`	bestimmt den Kopf der ersten Tabellenseite. Dieser Befehl ist nur nötig, wenn der Kopf der ersten Seite von den übrigen abweichen soll.
`\tabletail{...}`	legt den Text fest, der am Ende jeder Seite ausgegeben wird.
`\tablelasttail{..}`	bestimmt das Aussehen des Endes der gesamten Tabelle. Diese Befehl ist nur nötig, wenn das Ende der letzten Tabellenseite von den vorherigen abweichen soll.
`\topcaption{...}`	legt die Überschrift der Tabelle fest, wenn diese über der Tabelle erscheinen soll.
`\bottomcaption{...}`	leget die Überschrift der Tabelle fest, wenn sie unter der Tabelle erscheinen soll.

ßerdem sollte die `supertabular` Umgebung nicht innerhalb einer Gleitumgebung verwendet werden. In diesem Fall würde LaTeX 2_ε versuchen, die Tabelle komplett auf eine Seite zu setzen.

7.4.2 `longtable`

Möchte man komplexe mehrseitige Tabellen erzeugen, so sollte man besser das `longtable` Paket von David Carlisle verwenden. Es hat – wie der Name bereits andeutet – einige Ähnlichkeiten mit der `table` Umgebung. So lassen sich z. B. Überschriften mit dem `caption` Befehl erzeugen. Die Tabellen werden wie gewohnt durchnummeriert und in das Tabellenverzeichnis eingetragen. Die Umgebung `longtable` hat jedoch einige Unterschiede zur `supertabular` Umgebung. Vor allem ist die `longtable` Umgebung in der Lage, die Spaltenbreiten auf allen Seiten gleich groß zu setzen. Dazu wird die Breite des jeweils breitesten Spalteneintrags in der `.aux`-Datei gespeichert und beim nächsten LaTeX-Lauf übernommen. Um diese Funktion zu ermöglichen, muß vor dem Aufruf der ersten `longtable` der Befehl

```
\setlongtables
```

stehen. Am besten setzt man den Befehl direkt an den Anfang des Dokumentes. Da LaTeX 2_ε immer den breitesten Eintrag speichert, ist es nicht möglich, die Spalten nach einem Durchlauf wieder zu verkleinern. Dazu muß zunächst der `\setlongtables` Befehl wieder entfernt oder auskommentiert werden. Außerdem erhalten, wenn eine weitere `longtable` eingefügt wird, alle weiteren

Tabelle 7.16: Übersicht über die Parameter der `longtable` Umgebung

`\LTleft`	Länge links neben der Tabelle
`\LTright`	Länge rechts neben der Tabelle
`\LTpre`	Länge vor der Tabelle (voreingestellt auf `\bigskipamount`)
`\LTpost`	Länge nach der Tabelle (voreingestellt auf `\bigskipamount`)
`\LTcapwidth`	Breite der Tabellenüberschrift
`LTchunksize`	Anzahl der Zeilen pro Tabellenteil
`\endhead`	Ende der Zeilen, die am Kopf der Tabelle ausgegeben werden
`\endfirsthead`	Ende der Zeilen, die auf der ersten Seite am Kopf der Tabelle ausgegeben werden. Wird nur benötigt, wenn der erste Kopf sich von den weiteren Seiten unterscheiden soll.
`\endfoot`	Ende der Zeilen, die am Fuß der Tabelle ausgegeben werden.
`\endlastfoot`	beendet die Zeilen, die am Ende der gesamten Tabelle ausgegeben werden. Wird nur verwendet, wenn sich diese Zeilen von den Fußzeilen auf den übrigen Seiten unterscheiden.

Tabellen eine völlig falsche Spaltenbreite, da die Zählung verschoben wurde. Aus diesem Grund sollte man den `\setlongtables` erst kurz vor Ende der Arbeiten an einem Dokument einsetzen.

Formatiert man das Dokument ohne den `\setlongtables` Befehl, so führt dies dazu, daß auf den verschiedenen Seiten und selbst innerhalb einer Seite der Tabelle die Spalten unterschiedliche Breite haben. Daran sollte man sich zunächst nicht stören.

Das `longtable` Paket stellt eine Reihe Parameter bereit, mit denen das Aussehen der Tabellen gesteuert werden kann. Einen Überblick über die Parameter bietet die Tabelle 7.16. Einige der Parameter sollen an dieser Stelle etwas näher beschrieben werden.

Da mehrseitige Tabellen unter Umständen sehr lang werden können, müssen sie intern in kleinere Teile aufgeteilt werden. Auf diese Weise muß TEX nicht die gesamte Tabelle gleichzeitig im Arbeitsspeicher halten. Die Zahl der Zeilen pro Teil wird von dem Wert

```
LTchunksize
```

festgelegt. Dieser LATEX-Zähler ist standardmäßig auf 20 eingestellt. Er kann mit dem Befehl

```
\setcounter{LTchunksize}{Zeilenzahl}
```

beliebig geändert werden, allerdings muß er mindestens so groß sein, wie die Anzahl der Zeilen im Tabellenkopf oder -fuß. Der Wert hat keinerlei Einfluß auf die endgültige Form der Ausgabe, wenn der Befehl `\setlongtables` gesetzt wurde. Steht mehr Arbeitsspeicher zur Verfügung, so kann man den Wert problemlos erhöhen. Das Formatieren geht damit schneller.

Überschriften kann man wie gewohnt mit dem `\caption` oder `\caption*` Befehl setzen. Intern entspricht dem im Prinzip ein `\multicolumn` Befehl

```
\multicolumn{N}{p{\LTcapwidth}{Überschrift}
```

N entspricht dabei der Spaltenzahl. Die Breite der Überschrift wird mit dem Wert `\LTcapwidth` bestimmt. Standardmäßig beträgt er 4in. Er kann aber mit

```
\setlength{LTcapwidth}{Länge}
```

beliebig verändert werden. Die Überschrift wird zentriert auf jeder Seite über der Tabelle ausgegeben. Es ist auch möglich, auf späteren Seiten eine andere Überschrift zu setzen. Hierzu wird der normale `\caption` Befehl in den ersten Tabellenkopf übernommen. In dem weiteren Kopf wird nur der Befehl `\caption[`*Überschrift*`]` verwendet. Soll auf den weiteren Seiten die Tabellennummer nicht erscheinen, so kann man dies mit `\caption*[`*Überschrift*`]` erreichen.

Die Tabelle wird standardmaßig horizontal zentriert. Dafür sind die Befehle

```
\LTleft
\LTright
```

auf `\fill` eingestellt. Man kann jedoch mit `\setlength` auch eine Festlänge angeben. Geschieht dies mit beiden Größen, so ist damit die Breite der gesamten Tabelle festgelegt auf:

```
\columnwidth - \LTleft - \LTright
```

7.4.3 Vergleich zwischen `supertabular` und `longtable`

Zum Schluß sollen die beiden Möglichkeiten, mehrseitige Tabellen zu erzeugen, miteinander verglichen werden. Hierzu wird die gleiche Tabelle einmal als `supertabular` und einmal als `longtable` gesetzt. Bei der Tabelle handelt es sich um einen Bestandteil des Rollenspielsystems Rolemaster. Es ist ein Auszug aus der „Preisliste für magische Heilkräuter, Brote und Gifte“ und wurde einzig und allein der Länge wegen gewählt.

Die `supertabular` Version wurde mit den Befehlen aus Abbildung 7.2 erzeugt.

```
\tablefirsthead{
   \hline
   \multicolumn{5}{|c|}{\large\bfseries Magische Heilkräuter} \\
   Nr. & Name & Code & Form/Zubereitung & Preis \\ \hline}
\tablehead{
   \hline
   Nr. & Name & Code & Form/Zubereitung & Preis \\ \hline}
\tabletail{
   \hline
   \multicolumn{5}{|r|}{
      \small \textit{Fortsetzung nächste Seite \dots}} \\ \hline}
\tablelasttail{\hline}

\begin{supertabular}{|lllll|}
  \hline
  \multicolumn{5}{|c|}{\textit{Allgemeine Heilkräuter}} \\
  601 & Arkasu & m-L-4 & Saft/Einreiben   & 12 GS \\
  602 & Arlan  & k-B-2 & Wurzel/Einnehmen &  1 BS \\
  ...
\end{supertabular}
```

Abbildung 7.2: Erzeugen einer `supertabular` Tabelle

Die `longtable` Version ist praktisch genauso lang. Sie benötigt die in Abbildung 7.3 dargestellten Befehle zur Erzeugung.

Die Ausgabe der beiden Tabellen ist in den Abbildungen 7.4 und 7.5 auf den Seiten 128 und 129 zu sehen.

Um die unterschiedliche Behandlung der Spaltenbreiten zu zeigen, wurden einige Spalteneinträge absichtlich verbreitert und Abkürzungen ausgeschrieben. Deutlich zu erkennen sind aber nicht nur die bei der `supertabular` variierenden Spaltenbreiten, sondern auch die unterschiedliche Länge der Tabellen.

```
\setlongtables
\begin{longtable}{|lllll|}
  \hline
  \multicolumn{5}{|c|}{\large\bfseries Magische Heilkräuter} \\
  Nr. & Name & Code & Form/Zubereitung & Preis \\ \hline\hline
\endfirsthead
  \hline
  Nr. & Name & Code & Form/Zubereitung & Preis \\ \hline
\endhead
  \hline
  \multicolumn{5}{|r|}{\small
      \textit{Fortsetzung nächste Seite \dots}} \\ \hline
\endfoot
  \hline
\endlastfoot
  \multicolumn{5}{|c|}{\textit{Allgemeine Heilkräuter}}\\
  601 & Arkasu & m-L-4 & Saft/Einreiben   & 12 GS \\
  602 & Arlan  & k-B-2 & Wurzel/Einnehmen &  1 BS \\
  ...
\end{longtable}
```

Abbildung 7.3: Erzeugen einer `longtable` Tabelle

Magische Heilkräuter

Nr.	Name	Code	Form/Zubereitung	Preis
	Allgemeine Heilkräuter			
601	Arkasu	m-L-4	Saft/Einreiben	12 GS
602	Arlan	k-B-2	Wurzel/Einnehmen	1 BS
603	Athelas	n-N-5	Blatt/Kochen	300 GS
604	Attanar	n-F-4	Moos/Einreiben	8 GS
605	Delrean	k-N-2	Rinde/Einreiben	3 SS
606	Felmather	m-S-5	Blatt/Einnehmen	105 GS
607	Latha	n-F-4	Stiel/Kochen	9 SS
608	Trudurs	k-F-4	Moos/Kochen	12 SS
609	Ukur	s-H-4	Nuß/Einnehmen	34 SS
	Eigenschaftsmodifikationen			
610	Lestagii	t-Ö-9	Kristalle/Einnehmen	520 GS
611	Merrig	r-K-8	Dorn/Kochen	90 GS
	Gegengifte			
612	Argsbargies	t-W-5	Blume/Kochen	38 GS
613	Eldaana	k-S-4	Blatt/Kochen	99 GS
614	Menelar	s-N-5	Zapfen/Kochen	65 GS
615	Mook	n-B-3	Beere/Einnehmen	30 GS
616	Quilmufur	m-N-7	Wurzel/Kochen	49 GS
617	Shen	n-F-6	Blatt/Einnehmen	27 GS
618	Ul-Naza	r-Ö-8	Blatt/Einnehmen	430 GS
	Knochenheilung			
619	Arfandas	k-F-6	Stiel/Einreiben	2 SS
620	Baalak	r-S-8	Schilf/Kochen	160 GS
621	Bursthelas	n-K-8	Stengel/Kochen	110 GS
622	Edram	k-F-8	Moos/Einreiben	31 GS
623	Gursamel	n-K-7	Stengel/Einreiben	30 GS
	Körperliche Veränderungen & Verbesserungen			
624	Agaath	f-G-2	Beere/Einnehmen	5 GS
625	Ankii	r-T-7	Beere/Einnehmen	100 GS
626	Atigax	s-H-4	Wurzel/Kochen	40 GS
627	Blauaug	m-K-7	Blume/Kochen	15 GS
628	Breldiar	m-V-4	Blume/Einnehmen	25 GS
629	Elbens Korb	n-K-7	Wurzel/Kochen	10 GS
630	Gylvir	m-S-6	Alge/Einnehmen	45 GS
631	Joef	n-T-3	Pulver/Einnehmen	35 GS
632	Kathlusa	s-Ö-3	Blatt/Einnehmen	50 GS
633	Kilmakur	h-K-7	Wurzel/Kochen	65 GS
634	Klagul	r-K-3	Knospe/Kochen	27 GS

Fortsetzung nächste Seite ...

Nr.	Name	Code	Form/Zubereitung	Preis
635	Marku	r-H-6	Nuß/Einnehmen	30 GS
636	Megillos	k-B-3	Blatt/Einnehmen	12 GS
637	Rud-tekma	h-D-6	Frucht/Einnehmen	25 GS
638	Spreizfuß	n-F-4	Samen/Kochen	23 GS
639	Traubenblatt	m-M-6	Nektar/Einnehmen	7 GS
640	Yaran	n-K-2	Pollen/Einnehmen	9 SS
641	Zulsendura	t-U-4	Pilz/Einnehmen	70 GS
642	Zur	k-U-4	Schimmel/Kochen	12 GS
	Kreislaufmittel			
643	Anserke	h-S-6	Wurzel/Einreiben	75 GS
644	Fek	h-S-6	Nuß/Kochen	50 GS
645	Harfy	r-K-6	Harz/Einreiben	127 GS
646	Hugburtun	r-W-6	Frucht/Einreiben	180 GS
	Lebenserhaltende Mittel			
647	Carcatu	h-S-7	Gras/Einreiben	89 Goldstücke
648	Degiîk	h-S-5	Blatt/Einnehmen	100 GS
649	Laurelin	m-S-9	Blatt/Einnehmen	999 GS
650	Nur-oiolosse	s-F-8	Zwiebel/Einnehmen	200 GS
651	Oiolosse	s-F-8	Zwiebel/Einnehmen	600 GS
652	Olvar	s-S-6	Blume/Einnehmen	200 GS
653	Panthur	t-H-4	Knötchen/Kochen	35 GS
654	Tyr-fira	s-A-9	Blatt/Einreiben	1200 GS
655	Vulcurax	h-D-9	Beere/Einreiben	1000 GS
	Muskel-, Knorpel- und Sehnenheilung			
656	Arnuminas	m-K-2	Blatt/Einreiben	6 BS
657	Arpsusar	n-F-5	Stengel/Kochen	30 GS
658	Curfalaka	h-D-7	Frucht/Einnehmen	40 GS
659	Dagmather	r-K-5	Stachel/Kochen	28 GS
660	Ebur	m-S-4	Blume/Einnehmen	22 GS
661	Hegheg	h-K-8	Wurzel/Paste	25 GS
662	Tarfeg	h-S-7	Blume/Einnehmen	23 GS
	Nervenheilung			
663	Belramba	r-N-6	Flechte/Kochen	60 GS
664	Terbas	m-M-3	Blatt/Einreiben	2 GS
665	Wifurwif	n-B-7	Flechte/Einnehmen	55 GS
666	Yuth	h-D-8	Blume/Einnehmen	29 GS

Abbildung 7.4: Beispiel einer `supertabular` Tabelle

Magische Heilkräuter

Nr.	Name	Code	Form/Zubereitung	Preis
		Allgemeine Heilkräuter		
601	Arkasu	m-L-4	Saft/Einreiben	12 GS
602	Arlan	k-B-2	Wurzel/Einnehmen	1 BS
603	Athelas	n-N-5	Blatt/Kochen	300 GS
604	Attanar	n-F-4	Moos/Einreiben	8 GS
605	Delrean	k-N-2	Rinde/Einreiben	3 SS
606	Felmather	m-S-5	Blatt/Einnehmen	105 GS
607	Latha	n-F-4	Stiel/Kochen	9 SS
608	Trudurs	k-F-4	Moos/Kochen	12 SS
609	Ukur	s-H-4	Nuß/Einnehmen	34 SS
		Eigenschaftsmodifikationen		
610	Lestagii	t-Ö-9	Kristalle/Einnehmen	520 GS
611	Merrig	r-K-8	Dorn/Kochen	90 GS
		Gegengifte		
612	Argsbargies	t-W-5	Blume/Kochen	38 GS
613	Eldaana	k-S-4	Blatt/Kochen	99 GS
614	Menelar	s-N-5	Zapfen/Kochen	65 GS
615	Mook	n-B-3	Beere/Einnehmen	30 GS
616	Quilmufur	m-N-7	Wurzel/Kochen	49 GS
617	Shen	n-F-6	Blatt/Einnehmen	27 GS
618	Ul-Naza	r-Ö-8	Blatt/Einnehmen	430 GS
		Knochenheilung		
619	Arfandas	k-F-6	Stiel/Einreiben	2 SS
620	Baalak	r-S-8	Schilf/Kochen	160 GS
621	Bursthelas	n-K-8	Stengel/Kochen	110 GS
622	Edram	k-F-8	Moos/Einreiben	31 GS
623	Gursamel	n-K-7	Stengel/Einreiben	30 GS
		Körperliche Veränderungen & Verbesserungen		
624	Agaath	f-G-2	Beere/Einnehmen	5 GS
625	Ankii	r-T-7	Beere/Einnehmen	100 GS
626	Atigax	s-H-4	Wurzel/Kochen	40 GS
627	Blauaug	m-K-7	Blume/Kochen	15 GS
628	Breldiar	m-V-4	Blume/Einnehmen	25 GS
629	Elbens Korb	n-K-7	Wurzel/Kochen	10 GS
630	Gylvir	m-S-6	Alge/Einnehmen	45 GS
631	Joef	n-T-3	Pulver/Einnehmen	35 GS
632	Kathlusa	s-Ö-3	Blatt/Einnehmen	50 GS
633	Kilmakur	h-K-7	Wurzel/Kochen	65 GS
634	Klagul	r-K-3	Knospe/Kochen	27 GS
635	Marku	r-H-6	Nuß/Einnehmen	30 GS
636	Megillos	k-B-3	Blatt/Einnehmen	12 GS

Fortsetzung nächste Seite ...

Nr.	Name	Code	Form/Zubereitung	Preis
637	Rud-tekma	h-D-6	Frucht/Einnehmen	25 GS
638	Spreizfuß	n-F-4	Samen/Kochen	23 GS
639	Traubenblatt	m-M-6	Nektar/Einnehmen	7 GS
640	Yaran	n-K-2	Pollen/Einnehmen	9 SS
641	Zulsendura	t-U-4	Pilz/Einnehmen	70 GS
642	Zur	k-U-4	Schimmel/Kochen	12 GS
		Kreislaufmittel		
643	Anserke	h-S-6	Wurzel/Einreiben	75 GS
644	Fek	h-S-6	Nuß/Kochen	50 GS
645	Harfy	r-K-6	Harz/Einreiben	127 GS
646	Hugburtun	r-W-6	Frucht/Einreiben	180 GS
		Lebenserhaltende Mittel		
647	Carcatu	h-S-7	Gras/Einreiben	89 Goldstücke
648	Degiïk	h-S-5	Blatt/Einnehmen	100 GS
649	Laurelin	m-S-9	Blatt/Einnehmen	999 GS
650	Nur-oiolosse	s-F-8	Zwiebel/Einnehmen	200 GS
651	Oiolosse	s-F-8	Zwiebel/Einnehmen	600 GS
652	Olvar	s-S-6	Blume/Einnehmen	200 GS
653	Panthur	t-H-4	Knötchen/Kochen	35 GS
654	Tyr-fira	s-A-9	Blatt/Einreiben	1200 GS
655	Vulcurax	h-D-9	Beere/Einreiben	1000 GS
		Muskel-, Knorpel- und Sehnenheilung		
656	Arnuminas	m-K-2	Blatt/Einreiben	6 BS
657	Arpsusar	n-F-5	Stengel/Kochen	30 GS
658	Curfalaka	h-D-7	Frucht/Einnehmen	40 GS
659	Dagmather	r-K-5	Stachel/Kochen	28 GS
660	Ebur	m-S-4	Blume/Einnehmen	22 GS
661	Hegheg	h-K-8	Wurzel/Paste	25 GS
662	Tarfeg	h-S-7	Blume/Einnehmen	23 GS
		Nervenheilung		
663	Belramba	r-N-6	Flechte/Kochen	60 GS
664	Terbas	m-M-3	Blatt/Einreiben	2 GS
665	Wifurwif	n-B-7	Flechte/Einnehmen	55 GS
666	Yuth	h-D-8	Blume/Einnehmen	29 GS

Abbildung 7.5: Beispiel einer `longtable` Tabelle

Kapitel 8

Setzen von mathematischen Formeln

TEX ist vor allem dazu konzipiert worden, mathematische Texte zu setzen. LATEX stellt dem Nutzer dafür eine große Zahl wichtiger und hilfreicher Werkzeuge an die Hand. Diese erschöpfend zu erläutern, würde den Umfang dieser Einführung sprengen, darum soll hier nur ein kleiner Einblick in die umfangreichen Möglichkeiten, die LATEX bietet, gegeben werden.

Hingewiesen sein hier außerdem auf das AMSTEX-Paket (American Mathematical Society, 1997), daß die mathematischen Funktionen von LATEX erheblich erweitert.

8.1 Allgemeines

In LATEX stehen zwei verschiedene Möglichkeiten zum Setzen von Formeln und mathematischen Ausdrücken zur Verfügung: ein Textmodus, mit dem man mathematische Ausdrücke in einer Textzeile ausgeben kann, und eine Umgebung, mit der man abgesetzte Zeilen für die mathematischen Ausdrücke erzeugt. Dabei kann man noch entscheiden, ob die Zeilen (Formeln) numeriert werden sollen oder nicht.

Als mathematische Texte gelten hierbei sowohl komplette mathematische Formeln als auch einzelne Variablennamen, griechische Buchstaben oder auch das Hoch- und Tiefstellen von Texten sowie diverse Sonderzeichen.

Textmodus: Mathematische Texte innerhalb eines Absatzes werden zwischen `\(` und `\)` (Kurzform: `$` und `$`) eingeschlossen.

mathematischer Modus: Hier werden die Formeln zwischen den Befehlen `\begin{displaymath}` ... `\end{displaymath}` (Kurzform: `\[...\]`)

eingeschlossen. Dabei werden die Zeilen nicht numeriert.

Möchte man nummerierte Zeilen haben, so muß man die `equation`-Umgebung verwenden. In diesem Fall ist es dann auch möglich, mit `\label` und `\ref` im Text auf die Nummern verweisen.

Der mathematische Modus unterscheidet sich zu dem normalen Text-Modus in verschiedenen Punkten:

1. Leerstellen und Zeilenumbrüche bei der Eingabe haben keine Bedeutung, alle Abstände werden nach der Logik der mathematischen Ausdrücke automatisch bestimmt oder müssen durch spezielle Befehle wie `\,` oder `\qquad` angegeben werden.

2. Leerzeilen sind wirkungslos. Vertikale Abstände können nur mit `\vspace` oder mit `\rule` erzeugt werden.

3. Jeder einzelne Buchstabe wird als Name einer Variablen betrachtet und entsprechend gesetzt (kursiv mit zusätzlichem Abstand). Will man innerhalb eines mathematischen Textes normalen Text (in aufrechter Schrift, mit Wortabständen) setzen, muß man diesen z. B. in `\mathrm{...}` einschließen. Steht AMS-TeX zur Verfügung, so kann man auch den Befehl `\text{...}` verwenden.

8.2 Elemente in mathematischen Formeln

Im folgenden sollen kurz die wichtigsten Elemente mathematischer Formeln beschrieben werden. Anhang B enthält eine Liste aller verfügbaren Sonderzeichen. Bei Verwendung von AMS-TeX stehen darüberhinaus noch eine Reihe anderer Symbole und Befehle zur Verfügung, auf die an dieser Stelle jedoch nicht weiter eingegangen werden soll.

Exponenten und Indizes können mit den Zeichen `^` und `_` hoch- bzw. tiefergestellt werden. Dabei wird nur das nachfolgende Zeichen hoch- bzw. tiefergestellt. Möchte man mehrere Zeichen verschieben, so müssen sie in geschweifte Klammern gesetzt werden. Diese Befehle wird man sicherlich auch häufig in normalen Texten verwenden.

$E = mc^2$ $\qquad$ 12°C $\qquad$ H$_2$O

```
$E = mc^2$ \qquad
$12^\circ$C \qquad H$_2$O
```

Mit `\sqrt` erhält man ein Wurzelzeichen. LaTeX bestimmt dabei die Größe des Zeichens automatisch. Die n-te Wurzeln erhält man mit `\sqrt[n]`.

$\sqrt{x} \qquad \sqrt{\sqrt{x^{2}+\sqrt{y}}+x^\alpha}$	`$\sqrt{x}$ \qquad` `$\sqrt{\sqrt{x^{2}+\sqrt{y}}` `+x^\alpha}$`

Die Befehle `\overline` und `\underline` bewirken **waagrechte Striche** direkt über bzw. unter einem Ausdruck. **Waagerechte Klammern** dagegen erhält man mit den Befehlen `\overbrace` und `\underbrace`. Diese Befehle können selbstverständlich auch geschachtelt werden.

$\overline{A+B}$	`$\overline{A+B}$ \\`
$\underbrace{\overbrace{A+B}^\alpha-\overbrace{C+D}^\gamma}_{\chi}$	`$\underbrace{\overbrace{A+B}^\alpha-` `\overbrace{C+D}^\gamma}_{\chi}$`

Möchte man sich überlappende Klammern haben, so hat man zunächst das Problem, daß diese Überlappung durch die Klammer-Struktur verhindert wird. Um den Text dennoch setzen zu können, muß man den TeX-Befehl `\phantom` verwenden. Der Befehl setzt den in seinem Argument angegebenen Text nicht, sondern hält genau so viel Platz frei, wie dieser Text benötigt. Der Befehl `\hphantom` betrachtet dabei nur den benötigten horizontalen Platz, der Befehl `\vphantom` nur den vertikalen.

$\overbrace{\hphantom{A+B}}^\alpha\hphantom{-}\overbrace{\hphantom{C+D}}^\gamma$ $A+\underbrace{B-C}_{\chi}+D$	`$\overbrace{\hphantom{A+B}}^\alpha%` ` \hphantom{-}` ` \overbrace{\hphantom{C+D}}^\gamma$` `$A+\underbrace{B-C}_{\chi}+D$`

Runde und eckige **Klammern** kann man direkt eingeben, geschweifte erhält man mit `\{` und `\}`. Setzt man ein `\left` vor die öffnende Klammer und `\right` vor die schließende, so wird automatisch die richtige Größe gewählt. Dies gilt auch für senkrechte Striche (`\left|`). Will man dagegen die Größe selbst bestimmen, so kann man das mit den Befehlen `\big`, `\Big`, `\bigg` und `\Bigg` anstelle von `\left` tun.[1]

[1] Diese Befehle gibt es auch noch in einer zweiten Version. Dort wird den öffnenden Befehlen ein `l`, den schließenden ein `r` angehängt (`\Biggl`). In diesem Fall werden die Klammern

$1+\left(\frac{1}{1-x^{2}}\right)^{3}$ $\Bigl((x+1)(x-1)\Bigr)^{2}$	`$ 1 + \left( \frac{1}{ 1-x^{2} }` `    \right) ^3 \\` `\Bigl( (x+1) (x-1) \Bigr)^{2} $`

Für die **Modulo-Funktion** gibt es zwei verschiedene Befehle: `\bmod` für den binären Operator $a \bmod b$ und `\pmod{...}` für die Form $x \equiv a \pmod{b}$.

Brüche werden mit dem Befehl `\frac{`*Zähler*`}{`*Nenner*`}` gesetzt. Die Länge des Bruchstrichs wird dabei automatisch bestimmt. Für einfache Brüche kann man aber auch das Zeichen `/` verwenden.

$$\frac{x^{2}}{a-1}\qquad x^{\frac{2}{a+b}}\qquad d^{1/2}$$

```
\begin{displaymath}
\frac{ x^{2} }{ a-1 }\qquad
  x^{ \frac{2}{a+b} }\qquad d^{ 1/2 }
\end{displaymath}
```

Das **Integralzeichen** wird mit `\int` erzeugt, das **Summenzeichen** mit `\sum`. Die Grenzen werden mit `^` bzw. `_` oder mit dem Befehl `\limits` angegeben.

$$\sum_{i=1}^{n}\qquad\int_{0}^{\frac{\pi}{2}}\qquad\int\limits_{-\infty}^{+\infty}$$

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\int \limits_{-\infty}^{+\infty}
\end{displaymath}
```

Für drei hintereinander liegende **Punkte** (z. B. für $1, 2, \ldots, n$) gibt es die Befehle `\ldots` und `\cdots`. Dabei setzt `\ldots` die Punkte auf die Grundlinie (low) und `\cdots` in die Mitte der Zeilenhöhe (centered). Darüberhinaus gibt es die Befehle `\vdots` für vertikale und `\ddots` für diagonale Punkte ($\ddots$).

$x_{1},\ldots,x_{n}\qquad x_{1}+\cdots+x_{n}$	`$x_{1},\ldots,x_{n} \qquad` `  x_{1}+\cdots+x_{n}$`

bei der internen Verarbeitung von TEX als funktionell öffnende bzw. schließende Klammern interpretiert. Der Unterschied ist jedoch bei der Verwendung aus LATEX heraus zu vernachlassigen.

Hängt man den Klammerbefehlen ein `m` (`\biggm`) an, so werden die Klammern mit einem zusätzlichen horizontalen Abstand gesetzt.

Die Namen mathematischer Funktionen werden nicht kursiv gedruckt, sondern aufrecht. Auf diese Weise sind sie besser von den Variablen zu unterscheiden.

```
\arccos  \cos   \csc   \exp   \ker   \limsup  \min   \sinh
\arcsin  \sin   \cosh  \deg   \gcd   \lg      \ln    \Pr
\acrtan  \sup   \tanh  \tan   \cot   \det     \hom   \lim
\liminf  \sec   \arg   \coth  \dim   \inf     \log   \max
```

$\lim_{x \to 0} \frac{1-\cos x}{x^2} = \frac{1}{2}$

```
\[ \lim_{x \to 0}
   \frac{1-\cos x}{x^2}
   = \frac{1}{2} \]
```

8.2.1 Texte, Buchstaben und Akzente

Da Buchstaben im mathematischen Modus von LaTeX 2ε als Variablennamen interpretiert werden, muß man **normalen Text** mit den üblichen Schriftbefehlen erzeugen. Darüberhinaus stehen die `\mathXX`-Befehle zur Verfügung, mit denen andere Zeichensätze für die Variablen verwendet werden können.

$\mathcal{G} = k$
$\mathtt{a \neq b}$
$\mathbf{Ergebnis:}\ d = 89$
$\mathrm{Folge:}\ s > 73$
$\sum y = \mathit{k}$
$\mathsf{F}_{(x)} = x^2 + 56$
$ax + by = \mathnormal{dz}$

```
$ \mathcal{G} = k \\
\mathtt{a \neq b} \\
\mathbf{Ergebnis:} d = 89 \\
\mathrm{Folge:} s > 73 \\
\sum y = \mathit{k} \\
\mathsf{F}_{(x)} = x^2 + 56 \\
ax + by = \mathnormal{dz} $
```

Griechische Buchstaben können ebenfalls gesetzt werden. Kleinbuchstaben erhält man mit `\alpha`, `\beta`, `\gamma`, usw., große griechische Buchstaben mit `\mathrm{A}`, `\mathrm{B}`, `\Gamma`, `\Delta`, usw.

$\lambda, \xi, \gamma, \mu, \Phi, \Omega$

$\sin \lambda = 1.3\pi$

```
$\lambda, \xi, \pi, \mu,
  \Phi, \Omega $ \\
$\sin{\lambda} = 1.3\pi$
```

Außer den auch in normalem Texten verfügbaren Akzenten (vgl. Tabelle 3.6 auf Seite 29) stehen im mathematischen Modus eine Reihe weiterer Zeichen zur Verfügung.

$\hat{x}$	`\hat{x}`	$\breve{x}$	`\breve{x}`	$\check{x}$	`\check{x}`	$\tilde{x}$	`\tilde{x}`
$\grave{x}$	`\grave{x}`	$\acute{x}$	`\acute{x}`	$\dot{x}$	`\dot{x}`	$\widehat{abc}$	`\widehat{abc}`
$\bar{x}$	`\bar{x}`	$\vec{x}$	`\vec{x}`	$\ddot{x}$	`\ddot{x}`	$\widetilde{xyz}$	`\widetilde{xyz}`

Man sollte daran denken, daß die beiden Buchstaben `i`, `j` ohne Punkte verwendet werden sollen, wenn sie ein Akzent bekommen (`\i` und `\j`).

Die beiden Befehle `\widehat` und `\widetilde` können sich über mehrere Zeichen erstrecken, alle anderen nur über eins.

8.2.2 Ändern von Abständen

Es kann manchmal vorkommen, daß man mit den von LaTeX erzeugten Abständen in den Formeln nicht zufrieden ist. Zu diesem Zweck kann man die Abstände von Hand verändern.

Die wichtigsten Befehle dazu sind `\,` für einen kleinen Abstand, `\␣` für einen mittleren, `\quad` und `\qquad` für große Abstände sowie `\!` für die Verkleinerung eines Abstands.

Diese Befehle kann man mit Ausnahme von `\!` auch im normalen Text direkt (ohne Voranstellen eines `$`) benutzen. Größere Abstände oder genauere Angaben kann man mit `\hspace{...}` und `\rule` realisieren. Vertikale Abstände ändert man am besten mit den gewohnten Befehlen `\\[...]`, `\vspace{...}`, oder `\rule`.

$$F_n = F_{n-1} + F_{n-2} \quad n \ge 2 \tag{8.1}$$

```
\begin{equation}
F_{n} = F_{n-1} + F_{n-2}
  \quad n \ge 2
\end{equation}
```

$$\int\!\!\!\int_{D} dx\,dy \quad \textrm{statt} \quad \int\int_{D} dx dy \tag{8.2}$$

```
\begin{equation}
  \int\!\!\!\int_{D} dx\,dy
  \quad \textrm{statt}
  \quad \int\int_{D} dx dy
\end{equation}
```

8.3 Vektoren, Matrizen und Felder

Zur Erzeugung z. B. von Determinanten, Matrizen und Vektoren hat man verschiedene Möglichkeiten. Zunächst kann dazu die `array`-Umgebung verwendet werden. Sie funktioniert ganz ähnlich der `tabular`-Umgebung.

$$\mathbf{X} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

```
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
```

Auch kompliziertere Ausdrücke wie z. B. Gleichungssysteme können auf diese Weise gesetzt werden. Dabei kann man sich auch der verschiedenen Elemente der Tabellendefinition bedienen (siehe Abschnitt 7.1 auf Seite 99).

$$\begin{array}{c@{\ =\ }c}
7x + 15y^2 - (11b + x) & 9b \\[2pt]
3x + \frac{37}{49}y^2 + 13\frac{5}{9}b & 0 \\ \hline
8x + 15y^5 - 20b & 0 \\
3x + \frac{37}{49}y^2 + 13\frac{5}{9}b & 0
\end{array}$$

```
$ \begin{array}{c@{\ =\ }c}
7x + 15 y^2 - (11b + x) & 9b \\[2pt]
3x + \frac{37}{49}y^2
  + 13\frac{5}{9}b & 0 \\ \hline
...
\end{array} $
```

Möchte man **Vektoren** setzen, so wäre die Verwendung der `array`-Umgebung sehr mühselig, wenngleich natürlich möglich. Wesentlich einfacher jedoch sind die beiden Befehle `\matrix{...}` und `\pmatrix`. Der Unterschied zwischen den beiden Befehlen liegt darin, daß die Matrix im zweiten Fall eine Klammer erhält („parenthesized matrix"). Bei der Verwendung der beiden Befehle muß man beachten, daß es sich um TEX-Befehle handelt. Daher erfolgt der Zeilenumbruch *nicht* wie gewohnt mit `\\`, sondern mit `\cr` !

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix}$$

```
$ \pmatrix{a & b & c \cr
           d & e & f \cr
           g & h & i} *
\pmatrix{x \cr y \cr z} =
\pmatrix{\alpha \cr \beta \cr \gamma} $
```

Kleinere Ausdrücke können auch mit den TEX-Befehlen `\atop` und `\choose`

übereinander gesetzt werden. Beide Befehle erzeugen quasi einen Bruch ohne Bruchstrich. Sie sind also nur für zweizeilige Ausdrücke gedacht. Die Syntax ist in beiden Fällen gleich, der Unterschied liegt darin, daß `\choose` den Ausdruck in runde Klammern setzt (Binominalkoeffizient). Die letzte Möglichkeit, Formelteile übereinander zu setzen, bietet der Befehl `\stackrel{`*`Oben`*`}{`*`Unten`*`}`. Dieser setzt den Text der beiden Parameter zentriert übereinander. Dabei wird für den oberen Text eine kleinere Schriftart gewählt.

$$\binom{x+1}{n} = \binom{x}{n} + \binom{x}{n-1}$$

$$A \stackrel{\alpha'}{\longrightarrow} B$$

```
\[  {x+1 \choose n} =
 {x \choose n} + {x \choose n-1} \]

\[  A \stackrel{\alpha'}
 {\longrightarrow} B            \]
```

8.4 Mehrzeilige Formeln

Für **mehrzeilige** Formeln oder Gleichungssysteme werden die Umgebungen `eqnarray` und `eqnarray*` statt `array` verwendet. Bei `eqnarray` erhält jede Zeile eine eigene Gleichungsnummer, bei `eqnarray*` wird wie bei `displaymath` *keine* Nummer hinzugefügt. Sollen Gleichungssysteme eine gemeinsame Nummer erhalten, so kann man alle Zeilen bis auf eine mit dem Befehl `\nonumber` vor dem Zeilenumbruch `\\` beenden.

Die Umgebungen `eqnarray` und `eqnarray*` arbeiten wie eine 3-spaltige Tabelle der Form `{rcl}`, wobei die mittlere Spalte für das Gleichheits- oder Ungleichheitszeichen verwendet wird, nach dem die Zeilen ausgerichtet werden sollen.

$$f(x) = \cos x \qquad (8.3)$$

$$f'(x) = -\sin x \qquad (8.4)$$

$$\int_0^x f(y)dy \neq \sin x \qquad (8.5)$$

```
\begin{eqnarray}
f(x) & = & \cos x        \\
f'(x) & = & -\sin x      \\
\int_{0}^{x} f(y)dy &
 \not= & \sin x
\end{eqnarray}
```

Prinzipiell könnte man natürlich mehrzeilige Formeln auch mit der `array`-Umgebung erzeugen. Der Unterschied zu `eqnarray` besteht darin, daß bei diesen die Teile in der rechten und linken Spalte als *Formelteile* angesehen werden und nicht als *Textteile*. Dies führt dazu, daß bei `eqnarray` die Variablennamen und Symbole in einer angepaßten Größe erscheinen, Zähler und Nenner von Brüchen werden nicht verkleinert. Außerdem wird zusätzlicher Zwischenraum zwischen den Spalten eingefügt. Man vergleiche das folgende Beispiel mit dem auf Seite 137.

$$\begin{array}{rcl} 7x + 15y^2 - (11b + x) & = & 9b \\ 3x + \dfrac{37}{49}y^2 + 13\dfrac{5}{9}b & = & 0 \\ \hline 8x + 15y^5 - 20b & = & 0 \\ 3x + \dfrac{37}{49}y^2 + 13\dfrac{5}{9}b & = & 0 \end{array}$$

```
\begin{eqnarray*}
7x + 15 y^2 - (11b + x) & = & 9b \\
3x + \frac{37}{49}y^2
  + 13\frac{5}{9}b & = & 0 \\
...
\end{eqnarray*}
```

Zu lange Formeln werden von LaTeX *nicht* automatisch getrennt. Dies muß der Autor übernehmen. Er muß also bestimmen, wo die Zeilen umgebrochen werden und wieweit die folgenden Zeilen eingerückt werden sollen. In der Regel verwendet man dafür eine der beiden folgenden Varianten:

$$\begin{array}{rcl} \sin x & = & x - \dfrac{x^3}{3!} + \dfrac{x^5}{5!} - \\ & & {} - \dfrac{x^7}{7!} + \cdots \qquad (8.6) \end{array}$$

$$\begin{array}{l} \cos x = 1 - \dfrac{x^2}{2!} + \\ \qquad + \dfrac{x^4}{4!} - \dfrac{x^6}{6!} + \cdots \qquad (8.7) \end{array}$$

```
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
     +\frac{x^{5}}{5!} - {}
                      \nonumber\\
 & & {} -\frac{x^{7}}{7!} + \cdots \\
\lefteqn{
  \cos x = 1
     -\frac{x^{2}}{2!} + {} }
                      \nonumber\\
 & & {} +\frac{x^{4}}{4!}
     -\frac{x^{6}}{6!} + \cdots
\end{eqnarray}
```

Der in der zweiten Version verwendete Befehl `\lefteqn` bewirkt, daß der in Klammern angegebene Text zwar ausgedruck wird, LaTeX aber annimmt, daß dieser die Breite *Null* hat. Daher wird die zweite Zeile in diesem Beispiel nicht so weit eingerückt wie die des ersten Beispiels.

8.5 Weitere Elemente

8.5.1 Einrahmen von Formeln

Formeln lassen sich problemlos in vertikale Boxen (`parbox`, `minipage`; vgl. 4.7) setzen. Innerhalb dieser Boxen erscheinen die Formeln je nach Art der mathematischen Umgebung horizontal zentriert oder linksbündig mit der von `\mathindent` bestimmten Einrücktiefe. Auf diese Weise ist es selbstverständlich möglich, Formeln nebeneinander zu setzen.

Auch kann man erreichen, daß die Formelnummer horizontal zentriert zur Formel erscheint und nicht in einer bestimmten Zeile. Zu diesem Zweck werden zwei Absatzboxen erzeugt, von denen die eine die Formel und die andere die Nummer enthält. Alternativ kann man sich dafür auch eine neue Umgebung definieren.

$$\begin{aligned} f(x) &= \cos^2 x \\ \int f(x) &= \frac{1}{2}\left(x+\sin x \cos x\right) \end{aligned} \qquad (8.8)$$

```
\parbox{60mm}
  {\begin{eqnarray*}
    f(x)      & = & \cos^2 x \\
    \int f(x) & = & \frac{1}{2}
       \left(x+\sin x \cos x\right)
\end{eqnarray*}} \hfill
\parbox{7.5mm}
  {\begin{eqnarray}\end{eqnarray}}
```

Will man eine Formel hervorheben, indem man sie einrahmt, so ist dies mit einer `\fbox` ebenfalls möglich. Im Textmodus macht dies in der Regel überhaupt keine Probleme. Der Befehl `\fbox{$f(x)=x^2+7$}` erzeugt: $\boxed{f(x) = x^2 + 7}$. Möchte man einen größeren Abstand des Rahmens zu Formel haben, so benötigt man den Befehl `\fboxsep={`*Wert*`}`. In diesem Beispiel wurde der Abstand auf `8mm` gesetzt:

$$\boxed{\quad f(x) = x^2 + 7 \quad}$$

Sehr viel komplexer wird die Angelegenheit, wenn man numerierte Formeln einrahmen möchte. Eine Rahmenbox um die gesamte `eqnarray`-Umgebung würde die Nummer mit einrahmen, was normalerweise nicht gewünscht ist.

$$\boxed{\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots} \qquad (8.9)$$

```
\begin{eqnarray}
\fboxsep=12pt
\mbox{\fbox{$\sin x = x -
  \frac{x^3}{3!} +
  \frac{x^5}{5!} - \dots$}}
\end{eqnarray}
```

Auch hier kann es sinnvoll sein, sich dafür eine eigene Umgebung zu definieren, wenn man derartige Konstruktionen häufiger verwendet.

Beim Einrahmen von mehrzeiligen Formeln, die zusammen eine Nummer erhalten sollen, empfiehlt es sich, genauso vorzugehen, wie oben, also zwei Absatzboxen nebeneinander zu setzen. Dabei muß man jedoch die Breite der ersten Box durch Ausprobieren finden. Dies soll am Anfang der Herleitung für die Sinusreihe aus Formel 8.9 gezeigt werden.

$$\boxed{\begin{aligned} \sin x &= x + \left(\left[\frac{(t-x)^2}{2} \sin t\ dt \right]_0^x - \int_0^x \frac{(t-x)^2}{2} \cos t\ dt \right) \\ &= x - \int_0^x \frac{(t-x)^2}{2} \cos t\ dt \end{aligned}} \qquad (8.10)$$

Das Beispiel wurde erzeugt mit folgendem Quelltext:

```
\hspace*{\fill}\fbox{\parbox{295pt}
{\begin{eqnarray*}
  \sin x & = & x + \left( \left[ \frac{(t-x)^2}{2}
      \sin t\ dt \right]_0^x - \int_0^x
      \frac{(t-x)^2}{2} \cos t\ dt \right) \\
         & = & x - \int_0^x \frac{(t-x)^2}{2} \cos t\ dt
\end{eqnarray*}}} \hfill
\parbox{28pt}{\begin{eqnarray}\end{eqnarray}}
```

8.5.2 Chemische Formeln

Chemische Formeln werden nach etwas anderen Regeln gesetzt als mathematische. Der Hauptunterschied ist, daß der Text nicht kursiv erscheint. Außerdem werden einige Zeichen benötigt, die nicht direkt zur Verfügung stehen, so zum

Beispiel manche Pfeilsymbole. Diese kann man sich jedoch recht einfach selber definieren, wie das Beispiel zeigt.

$$\mathrm{A} + \mathrm{B} \mathrel{\mathop{\rightleftharpoons}\limits^{K_1}_{K_2}} \mathrm{C}$$

```
\newcommand\chemrel[2]
  {\mathrel{\mathop
    {\rightleftharpoons}
    \limits^{#1}_{#2}}}
\[  \mbox{A} + \mbox{B}
    \chemrel{K_1}{K_2} \mbox{C}  \]
```

Kapitel 9

LaTeX 2ε und Graphik

In diesem Kapitel sollen die verschiedenen Möglichkeiten zur Erzeugung oder Einbindung von Graphiken in LaTeX-Dokumente erläutert werden. Dabei kann jedoch nur ein erster oberflächlicher Einblick gegeben werden. Ausführlich wird das Thema zum Beispiel in „The LaTeX Graphics Companion“ (Goossens et al., 1997) behandelt.

Im ersten Abschnitt wird die `picture` Umgebung von LaTeX behandelt. Diese bietet nur wenige, einfache Zeichenelemente. Wesentlich mehr Möglichkeiten stehen durch die zahlreichen Zusatzpakete zur Verfügung, von denen einige im zweiten Abschnitt vorgestellt werden. Im dritten Abschnitt sollen verschiedene Werkzeuge erläutert werden, die eine Verknüpfung der Graphikfähigkeiten von PostScript und LaTeX bieten. Im letzten Teil des Kapitels soll dann noch auf Verwendung farbiger Elemente in Dokumenten eingegangen werden. Doch zunächst folgt ein kurzer Überblick über die verschiedenen Ansätz der Graphikerstellung.

TeX selbst bietet gar keine Möglichkeiten, Zeichnungen zu erstellen oder einzubinden. Es gibt lediglich den `\special`-Befehl, mit dem Befehle direkt an den `dvi`-Treiber weitergegeben werden können. Der Befehl selbst hat keinen Einfluß auf die Formatierung[1] Es bleibt dem `dvi`-Treiber überlassen, die durch `\special` eingefügten Informationen zu verarbeiten. Im Abschnitt über PostScript auf Seite 163 wird der Befehl erneut erläutert werden.

Insgesamt gibt es sechs verschiedene Ansätze, Graphiken mit TeX und LaTeX zu erzeugen (Goossens et al., 1994, S. 283).

1. Zeichensätze mit Bildelementen. Auf diesem Prinzip basiert beispielsweise die `picture`-Umgebung von LaTeX (siehe Abschnitt 9.1). Die Elemente

[1] Dies ist nicht ganz korrekt: In bestimmten Fällen hindert die Existenz des Befehls in der Eingabedatei LaTeX daran, einen Leerraum korrekt als solchen zu erkennen. Die Folge ist, daß ein zusätzlicher Leerraum eingefügt wird.

der Zeichnung werden dabei aus einzelnen Zeichen zusammengesetzt.

2. Bildmakropakete, die auf der LaTeX `picture`-Umgebung oder auf den Linienbefehlen von TeX basieren (siehe Abschnitt 9.2).

3. Bildzeichensätze, in denen jedes Zeichen, das gesetzt werden soll, einem Zeichen aus einem speziell dafür erzeugten Zeichensatz entspricht. Diese Zeichensätze werden zum Beispiel mit den Programmen pbmtopk oder BM2FONT erzeugt. Beide Programme können aus Graphikdateien TeX-Fonts erstellen, in denen die Graphiken gerastert und Buchstaben zugeordnet werden.

4. Halbton-Zeichensätze bestehen aus kleinen Flächen mit unterschiedlichen Graustufen, die auf herkömmliche TeX-Weise zu Bildern zusammengesetzt werden können.

5. Bilder aus ASCII-Zeichen. Hier bietet zum Beispiel PICTeX einen Umfangreichen Befehlssatz zur Erzeugung komplexer Zeichnungen. Kurven werden dabei beispielsweise aus vielen kleinen Punkten zusammengesetzt. Dabei wird jedoch eine relativ hohe Anforderung an die Hardware, insbesondere den zur Verfügung stehenden Arbeitsspeicher, gestellt.[2]

6. Verwendung der `\special`-Befehle von TeX. Damit können zum Beispiel externe Graphikdateien eingebunden werden (siehe Abschnitt 9.3.1).

   Dieser Ansatz ist geräteabhängig, da die `\special`-Befehle vom jeweiligen `dvi`-Treiber verarbeitet werden. Aufgrund der starken Verbreitung von PostScript-Previewern und -Druckern gewinnt dieser Ansatz mehr und mehr an Bedeutung. Für PostScript gibt es außerdem verschiedene Makropakete, die eine Erzeugung selbst komplexer Zeichnungen ermöglichen.

9.1 Die `picture` Umgebung

Mit der `picture` Umgebung von LaTeX können relativ schnell einfache Zeichnungen erzeugt werden. Es stehen als Grundelemente hierfür gerade Linien verschiedener Neidungen, Pfeile, Kreise, Bezierkurven und Text zur Verfügung, die frei in dem Bild plaziert werden können.

Da alle Elemente intern aus Zeichen verschiedener Zeichensätze zusammengesetzt werden, stehen immer nur eine begrenzte Anzahl von Elementen zur

[2] Bei komplexeren Zeichnungen kann die Bearbeitung einer PICTeX-Seite durchaus 25 bis 30 mal länger dauern als die einer normalen Textseite.

Verfügung. So können beispielsweise für Linien nur 48 und für Pfeile nur 24 verschiedene Neigungen verwendet werden.

9.1.1 Koordinatensystem

Damit die einzelnen Bildelemente in dem Bild plaziert werden können, muß zunächst ein Koordinatensystem erzeugt werden. Als erstes wird dazu die Einheitslänge mit dem Befehl

```
\setlength{\unitlength}{Länge}
```

festgelegt. Dies muß vor der `picture`-Umgebung erfolgen. Alle Positions- und Größenangaben innerhalb eines Bildes erfolgen somit ohne Maßangabe als reine Dezimalzahl. Somit kann das Bild leicht durch Änderung der Einheitslänge skaliert werden. Als Zahlen dürfen auch Kommazahlen angegeben werden. Hierbei ist jedoch zu beachten, daß das Komma in der `picture`-Umgebung Zahlenpaare trennt. Es darf daher nur der Dezimalpunkt innerhalb einer Zahl verwendet werden.

Das Koordinatensystem wird mit Erzeugung der `picture`-Umgebung automatisch angelegt.

```
\begin{picture}(Breite,Höhe)(ΔX,ΔY)
\end{picture}
```

Breite und *Höhe* geben dabei die Größe des Bildes in Vielfachen der Einheitslänge an. Diese Angabe wird nur benötigt, um ein entsprechend großes Rechteck für das Bild freizuhalten. Die Angabe hat keinerlei Einfluß darauf, wo Bildobjekte plaziert werden können. Man kann also auch ohne Probleme Objekte *außerhalb* dieses Rechtecks plazieren.

Normalerweise hat die linke untere Ecke des Bildes die Koordinaten `(0,0)`, das Koordinatensystem kann jedoch durch die optionale Angabe von ΔX und ΔY verschoben werden.

Zur Positionierung von Bildelementen stehen dem Anwender zwei Befehle zur Verfügung.

```
\put(x,y){Bildobjekt}
\multiput(x,y)(Δx,Δy){Anzahl}{Bildobjekt}
```

Die Parameter *x* und *y* legen die Position fest, an der *Bildobjekt* erscheinen soll. *Bildobjekt* kann eines der im Folgenden vorgestellten Zeichenelemente von LaTeX oder auch eine LR- bzw. Absatz-Box sein. Auch weitere `picture`-Umgebungen sind möglich.

Mit dem zweiten Bildbefehl kann man ein Objekt mehrfach setzen. Das Bildobjekt wird so oft erzeugt, wie mit dem Parameter ***Anzahl*** angegeben. $\Delta \boldsymbol{x}$ und $\Delta \boldsymbol{y}$ legen die Inkremetierungswerte fest, um die die Koordinaten $\boldsymbol{x}$ und $\boldsymbol{y}$ mit jedem nochmaligen Auftreten verändert werden. Das erste Objekt erscheint also bei $(\boldsymbol{x},\boldsymbol{y})$, das letzte bei

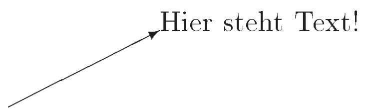
$$\Big((\boldsymbol{x} + [\textit{\textbf{Anzahl}} - 1] * \Delta \boldsymbol{x}),\ (\boldsymbol{y} + [\textit{\textbf{Anzahl}} - 1] * \Delta \boldsymbol{y}) \Big)$$

9.1.2 Standard LaTeX-Bildobjekte

LaTeX stellt nur relativ wenige Grundelemente für Zeichnungen bereit: Texte, Linien und Pfeile, Kreise, Ovale und Kreissegmente, Rechtecke sowie quadratische Bezier-Kurven.

Text

Das einfachste Bildelement ist ein einzeiliger Text. Dieser kann direkt in den `\put`-Befehl eingetragen werden. Die Koordinate des Befehls gibt dabei die linke untere Ecke des Textes an.




```
\setlength{1mm}
\begin{picture}(50,15)
  \put(0,0){\vector(2,1){20}}
  \put(20,10){Hier steht Text!}
\end{picture}
```

Möchte man mehrzeiligen Text erzeugen, so muß dieser in eine `\parbox` oder eine `minipage` gesetzt werden. Dabei kann der Parameter zur Positionierung wie gewohnt verwendet werden, um die genaue Lage der Box festzulegen. Die drei Boxen im folgenden Beispiel wurden jeweils mit der gleichen y-Koordinate erzeugt.

`\parbox{35mm}{...}`	`\parbox[t]{35mm}{...}`	`\parbox[b]{35mm}{...}`
Bezugspunkt dieser Box ist die vertikale Mitte des linken Randes	Bezugspunkt dieser Box ist die Grundlinie der ersten Zeile	Bezugspunkt dieser Box ist die Grundlinie der letzten Zeile

Linien und Pfeile

Linien und Pfeile werden mit den beiden Befehlen

```
\line(Δx,Δy){Länge}
\vector(Δx,Δy){Länge}
```

erzeugt. Der Strich beginnt bei den Koordinaten des `\put`-Befehls. Die Neigung wird durch die beiden Parameter Δx und Δy angegeben. Die Angabe `(0,1)` erzeugt beispielsweise eine vertikale Linie, die vom Fußpunkt nach oben reicht. Der Parameter ***Länge*** gibt bei vertikalen oder horizontalen Strichen die tatsächliche Länge an, bei geneigten Strichen jedoch die Länge der *Projektion des Striches auf die x-Achse*. Die Bedeutung der Parameter kann man in der nebenstehenden Abbildung leicht nachvollziehen.

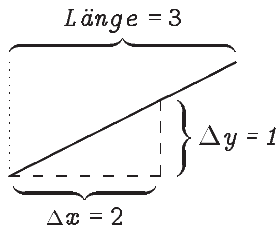


Abb. 9.1: Verwendung des `\line`-Befehls

Wie eingangs bereits erwähnt, steht nur eine begrenzte Anzahl von Neigungen zur Verfügung. Die Parameter Δx und Δy müssen drei Bedingungen erfüllen:

1. Es sind nur ganze Zahlen zugelassen.
2. Das Zahlenpaar darf keinen gemeinsamen Teiler haben.[3] `(2,4)` ist also nicht erlaubt, stattdessen muß `(1,2)` geschrieben werden.
3. Bei Linien sind nur Werte zwischen -6 und 6, bei Pfeilen zwischen -4 und 4 erlaubt.

Damit können insgesamt 48 verschiedene Steigungen bei Linien und 24 bei Pfeilen verwendet werden. Außerdem stehen für Striche nur zwei verschiedenen Strichstärken zur Verfügung. Ausgewählt werden sie mit den Befehlen

```
\thinlines
\thickline
```

In der Standardeinstellung werden dünne Striche verwendet. Ein Umschalten mit `\thicklines` wählt eine um den Faktor 1,5 dickere Strichstärke aus. Für horizontale und vertikale Striche kann außerdem die Strichstärke direkt angegeben werden.

```
\linethickness{Stärke}
```

[3] Ausnahme: 1 bzw. -1

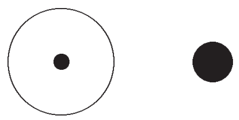
Für ***Stärke*** ist jede positive, feste Längenangabe erlaubt. Die Wirkung des Befehls endet, sobald entweder ein weiterer `\linethickness`-Befehl ein andere Strichstärke auswählt oder einer der beiden Strichstärkenbefehle auftritt.

Kreise, Ovale und Kreissegmente

Mit den Befehlen

```
\circle{Durchmesser}
\circle*{Durchmesser}
```

können Kreise erzeugt werden. Auch hier stehen nur bestimmte Durchmesser zur Verfügung. Es wird jeweils der Kreis verwendet, dessen Durchmesser dem angegebenen ***Durchmesser*** am nächsten kommt. Die zweite Form des Befehls füllt den Kreis komplett mit Farbe. Der Bezugspunkt der Befehle ist jeweils der Kreismittelpunkt.

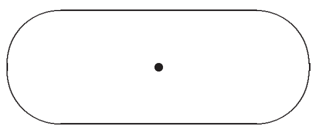


```
\setlength{\unitlength}{1mm}
\begin{picture}(50,25)
  \put(10,10){\circle{15}}
  \put(10,10){\circle*{2}}
  \put(30,10){\circle*{10}}
\end{picture}
```

Ovale und Kreissegmente werden mit dem Befehl

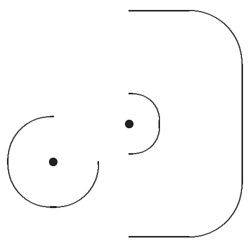
```
\oval(Breite,Höhe)[Segment]
```

erzeugt. Ein Oval ist dabei ein Rechteck mit der angegebenen ***Breite*** und ***Höhe***, das abgerundete Ecken erhält. Es werden hierfür die größtmöglichen Kreisradien verwendet, mit denen die aneinanderstoßenden Seiten verbunden werden können. Der Bezugspunkt des Befehls ist wiederum der Mittelpunkt.



```
\begin{picture}(50,20)
  \put(25,10){\oval(40,15)}
  \put(25,10){\circle*{1}}
\end{picture}
```

Mit dem Parameter *Segment* kann optional ein bestimmtes Segment des Ovals erzeugt werden. Für *Segment* kann man `t`, `b`. `l` oder `r` bzw. eine Kombination davon angeben. Auf diese Weise lassen sich auch Halb- oder Viertelkreise erzeugen. Da jedoch auch bei Ovalen nur die für Kreise möglichen Radien zur Verfügung stehen, kann es passieren, daß die Halbkreise nicht korrekt dargestellt werden. Dies gilt vor allem für große Radien.



```
\begin{picture}(40,30)
  \put(20,25){\oval(8,8)}
  \put(20,25){\oval(30,30)[r]}
  \put(20,25){\circle*{1}}
  \put(10,20){\oval(12,12)[tl]}
  \put(10,20){\oval(12,12)[b]}
  \put(10,20){\circle*{1}}
\end{picture}
```

Rechtecke

Die bereits vorgestellten Box-Befehle (siehe 4.7.1 auf Seite 65) können in einer `picture`-Umgebung problemlos weiterverwendet werden. Gleichzeitig stehen weitere LR-Boxen zur Verfügung

```
\framebox(Breite,Höhe)[Pos]{Text}
\makebox(Breite,Höhe)[Pos]{Text}
\dashbox{Strich}(Breite,Höhe)[Pos]{Text}
```

Die bekannten Befehle `\framebox` und `\makebox` können in einer erweiterten Form verwendet werden, zusätzlich kann eine Box mit gestrichelten Rahmen erzeugt werden. *Strich* legt die Länge der einzelnen Striche fest.

Alle diese Boxen haben die angegebene *Breite* und *Höhe*. Wird eine `\dashbox` verwendet, so sollten diese Werte möglichst ganzzahlige Vielfache der Strichlänge sein. Der Bezugspunkt der Boxen ist immer die linke untere Ecke. Der Inhalt der Box erscheint horizontal und vertikal zentriert. Der optionale Parameter *Pos* läßt mit `l`, `r`, `t`, `b` oder einer Paarkombination (z.B. `rt`) eine Änderung zu.

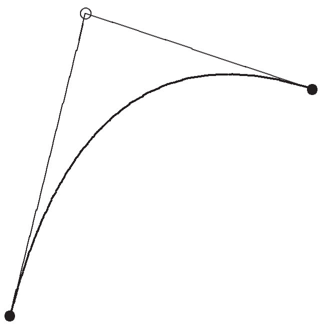
Bezier-Kurven

Als letztes Zeichenelement bietet LaTeX 2_ε die Möglichkeit von quadratischen Bezier-Kurven. Hierzu dient der Befehl

`\qbezier[n](AX,AY)(BX,BY)(CX,CY)`

Der Befehl erzeugt eine Kurve, mit den beiden Endpunkte `(AX,AY)` und `(CX,CY)`. Der Punkt `(BX,BY)` dient als Kontrollpunkt, wobei die Linen $\overline{AB}$ und $\overline{BC}$ die Kurventangenten in den Punkte A und C bildet.

Mit dem optionalen Parameter *n* kann die Anzahl der Punkte festgelegt werden, mit denen die Kurve gezeichnet wird. Wird die Angabe weggelassen, so werden so viele Punkte gezeichnet, daß eine durchgezogenene Linie entsteht.



```
\setlength{\unitlength}{1mm}
\begin{picture}(60,50)
  \put(5,5){\line(1,4){10}}
  \put(15,45){\line(3,-1){30}}
  \put(5,5){\circle*{1.5}}
  \put(15,45){\circle{1.5}}
  \put(45,35){\circle*{1.5}}
  \thicklines
  \qbezier(5,5)(15,45)(45,35)
\end{picture}
```

9.2 Erweiterungen durch Zusatzpakete

Die Grundfunktionen der `picture`-Umgebung von LATEX haben zahlreiche Beschränkungen und sind zum Teil nur wenig intuitiv zu bedienen. Besonders bei der Verwendung von `\line` oder `\vector` Befehlen muß man häufig sehr viel rechnen oder ausprobieren, bis das gewünschte Ergebnis erreicht ist.

Es gibt zahlreiche Ansätze, diese Nachteile zu beheben und die Zeichenfunktionen zu erweitern. In diesem Abschnitt sollen einige dieser Zusatzpakete und ihre Werkzeuge kurz vorgestellt werden.

9.2.1 `epic` -Paket

Das Paket `epic` wurde von Sunil Podar entwickelt (PODAR, 1986). Es stellt dem Benutzer eine leistungsfähige High-Level-Schnittstelle zur LATEX-`picture`-Umgebung zur Verfügung. In vielen Fällen entfällt dabei ein umständliches Berechnen von Positionen und Längen.

Positionierungsbefehle

Der LaTeX-Befehl `\multiput` ist nur in der Lage, *dasselbe* Objekt mehrfach in bestimmten Abständen zu erzeugen. Möchte man jedoch verschiedene Objekte haben, so müßte man dazu entsprechend viele `\put`-Befehle verwenden. `epic` bietet hier den Befehl

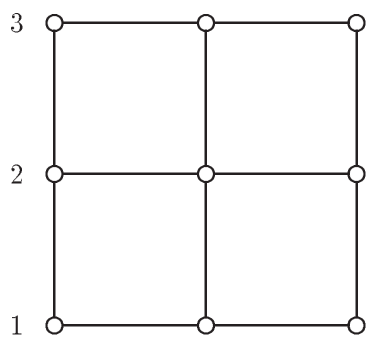
`\multiputlist(`x`,`y`)(`Δx`,`Δy`)[`*Pos*`]{`*Objekt1,Objekt2,...*`}`

Die Verwendung des Befehls ist ähnlich zum `\multiput`-Befehl. Allerdings werden die Objekte der Liste nacheinander plaziert. Diese Objekte können praktisch alles sein. Allerdings muß man beachten, daß die Objekte mit Kommas getrennt werden. Daher sollte man Objekte, die nicht nur aus einem einzelnen Befehl bestehen, immer in geschweifte Klammern stellen und ohne Leerzeichen zu den Kommas in die Liste setzen. Enthalten die Befehle selbst Kommas, so müssen sie auf jeden Fall in Klammern gesetzt werden.

Die Objekte werden intern in eine `\makebox(0,0)[`*Pos*`]` gesetzt. Der optionale Parameter *Pos* des `\multiputlist`-Befehls wird an die `\makebox` weitergereicht. Somit kann die Lage der Objekte genauer definiert werden.

`\matrixput(`x`,`y`)(`Δx_1`,`Δy_1`){`n_1`}(`Δx_2`,`Δy_2`){`n_2`}{`*Objekt*`}`

Der Befehl `\matrixput` ist ein zweidimensionaler `\multiput`-Befehl. Er eignet sich immer dann, wenn ein Objekt in regelmäßigen Abständen in zwei Dimensionen gesetzt werden soll.



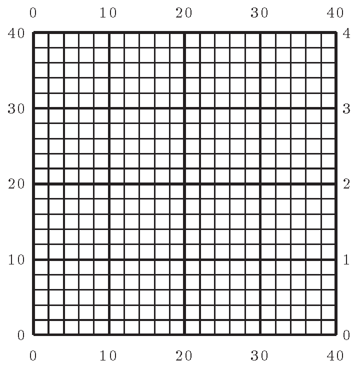

```
\setlength{\unitlength}{1mm}
\begin{picture}(60,50)
  \thicklines
  \matrixput(6,5)(20,0){2}
    (0,20){3}{\line(1,0){8}}
  \matrixput(5,6)(20,0){3}
    (0,20){2}{\line(0,1){8}}
  \matrixput(5,5)(20,0){3}
    (0,20){3}{\circle{2}}
  \multiputlist(1,5)(0,20){1,2,3}
\end{picture}
```

Zeichenbefehle

Das `epic`-Paket stellt auch verschiedene neue Befehle zum Zeichnen bereit. Mit dem Befehl

```
\grid(Breite,Höhe)(ΔBreite,ΔHöhe)[X-Start,Y-Start]
```

kann ein Raster mit der angegebenen *Breite* und *Höhe* erzeugt werden. Dabei werden horizontale Linien im Abstand Δ*Höhe* und vertikale im Abstand Δ*Breite* gezeichnet. Sinnvollerweise sollten *Breite* und *Höhe* jeweils ganzzahlige Vielfache von Δ*Breite* und Δ*Höhe* sein. Werden die optionalen Parameter *X-Start* und *Y-Start* angegeben, so werden die Linien zusätzlich mit Nummern in Schritten von Δ*Breite* und Δ*Höhe* versehen. Sollen nicht alle Linien durchnummeriert werden, so benutzt man einfach zwei `\grid`-Befehle.




```
\setlength{\unitlength}{1mm}
\begin{picture}(50,50)
  \put(5,5){\grid(40,40)(2,2)}
  \thicklines
  \put(5,5){\tiny
    \grid(40,40)(10,10)[0,0]}
\end{picture}
```

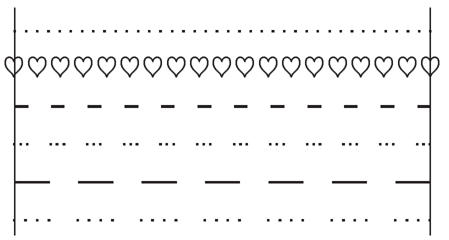
Das `epic`-Paket stellt insgesamt drei Befehle zum Zeichnen von Linien. Hauptunterschied zum LaTeX-Befehl `\line` ist zum einen, daß die Befehle beliebig viele Punkte verbinden können, und zum anderen, daß eine Linie über die Punkte, durch die sie hindurch geht, definiert wird und nicht durch Anfangspunkt, Steigung und Länge.

```
\dottedline[Zeichen]{Lücke}(x1,y1)(x2,y2)...(xn,yn)
```

Der Befehl `\dottedline` zieht eine gepunktete Linie durch die angegebenen Koordinatenpaare. Hierzu müssen mindestens zwei Punkte angegeben werden. Der *Lücke* legt, in `\unitlength`-Einheiten, den Abstand der einzelnen Punkte fest. Da jedoch die Anzahl der zu setzenden Punkte ganzzahlig sein muß, kann es sein, daß der tatsächliche Punktabstand etwas von der Vorgabe abweicht. Standardmäßig wird ein kleines Quadrat als Punkt verwendet. In diesem Fall wird die Strichstärke mit `\thicklines` und `\thinlines` eingestellt. Mit dem optionalen Parameter ***Zeichen*** kann jedoch auch ein anderes Zeichen gewählt werden. Dabei sollte beachtet werden, daß nicht alle Zeichen in allen Zeichensätze vertikal zentriert werden; so zum Beispiel »*«.

`\dashline`[*Dehn*]{*Länge*}[*Lücke*](x_1,y_1)(x_2,y_2)...(x_n,y_n)

Mit dem `\dashline`-Befehl wird eine gestrichelte Linie zwischen den angegebenen Punkten gezogen. Der optionale Parameter ***Dehn*** legt fest, wie stark die Striche gedehnt oder geschrumpft werden. Der Parameter muß zwischen -100 und $+\infty$ liegen. ***Länge*** legt die Länge der verwendeten Striche fest, ***Lücke*** definiert den Abstand zwischen den Punkten, die für einen Strich verwendet werden. Standardmäßig werden durchgezogene Striche verwendet.

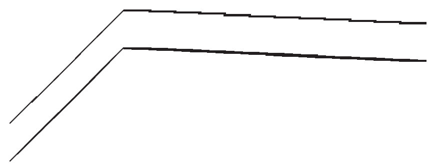


```
\setlength{\unitlength}{1mm}
\begin{picture}(60,35)
  \thicklines
  \dottedline{1.5}(5,35)(60,35)
  \dottedline[$\heartsuit$]{3}
    (5,30)(60,30)
  \dashline{2}(5,25)(60,25)
  \dashline{2}[1.0](5,20)(60,20)
  \dashline[+20]{5}(5,15)(60,15)
  \dashline[+20]{5}[1.5](5,10)(60,10)
\end{picture}
```

`\drawline`[*Dehn*](x_1,y_1)(x_2,y_2)...(x_n,y_n)

Der `\drawline`-Befehl zieht eine geschlossene Linie durch die angegebenen Punkte. Dabei verwendet er aus den Zeichensätzen die verfügbaren Strichsegmente, deren Steigung der gewählten am nächsten kommt. Daher kann es sein, daß die Linien gezackt erscheinen. Die Strichstärke kann mit `\thicklines` und `\thinlines` eingestellt werden.

Die Anzahl der Strichsegmente, aus denen die Linie zusammengesetzt wird, kann mit dem optionalen Parameter ***Dehn*** eingestellt werden. Wenn ***Dehn*** größer als Null ist, werden mehr Striche verwendet. Dadurch entsteht eine weniger gezackte Linie. Standardmäßig werden so viele Striche verwendet, wie nötig ist, um die Linie durchgezogen erscheinen zu lassen.



```
\setlength{\unitlength}{1mm}
\begin{picture}(60,30)
  \drawline(5,10)(20,25)(60,23.3)
  \drawline[200](5,5)(20,15)(60,18.3)
\end{picture}
```

Das `epic`-Paket bietet drei verschiedene Umgebungen zum Zeichnen von Linien. Sie entsprechen in ihrer Wirkung den bereits vorgestellten Zeichenbefehlen.

```
\jput(x,y){Objekt}

\begin{dottedjoin}[Zeichen]{Lücke}
\end{dottedjoin}

\begin{dashjoin}[Dehn]{Länge}[Lücke]
\end{dashjoin}

\begin{drawjoin}[Dehn]
\end{drawjoin}
```

Die Parameter der Umgebungen sind identisch mit den jeweiligen Zeichenbefehlen. Die Punkte innerhalb der Umgebungen werden mit dem Befehl `\jput` (für: „join and put") eingegeben. Die diesem Befehl übergebenen Objekte werden dabei an den angegebenen Koordinaten ausgegeben und miteinander verbunden.

Möchte man umfangreichere Diagramme erstellen, so kann es sehr aufwendig werden, die einzelnen Punkte einzugeben. Das `epic`-Paket stellt hierfür den Befehl

```
\putfile{Dateiname}{Objekt}
```

bereit. Mit diesem Befehl ist es möglich, Daten, die von einem anderen Programm berechnet wurden, von LaTeX setzen zu lassen. Der Befehl liest hierzu die externe Datei *`Dateiname`* ein. Diese ASCII-Datei muß in jeder Zeile ein Koordinatenpaar der Form „x y" (mit Leerzeichen) enthalten.

Die Datei darf Kommentare enthalten. Diese werden mit `%` eingeleitet. Steht ein Kommentar in der gleichen Zeile wie ein Koordinatenpaar, so sollte mindestens ein Leerzeichen hinter dem Zahlenpaar stehen.

9.2.2 `eepic`-Paket

Das `epic`-Paket stellt zwar eine deutliche Erweiterung der Zeichenfähigkeiten von LaTeX dar, unterliegt jedoch immer noch einzelnen Beschränkungen. Diese Beschränkungen versucht das `eepic`-Paket von Conrad Kwok aufzuheben. Dabei greift das Paket auf die TeX-Fähigkeit zurück, `\special`-Befehle zu erzeugen. Um das Paket nutzen zu können, muß also ein `dvi`-Treiber vorhanden sein, der diese Befehle verstehen kann. Die meisten Treiber sind dazu heute jedoch in der Lage. Trotzdem geht dieser Ansatz auf Kosten der Geräteunabhängigkeit der `dvi`-Datei.

Neudefinition bestehender Befehle

Das `eepic`-Paket implementiert einige der Zeichenbefehle von LaTeX neu. Diese Befehle ermöglichen es dem Nutzer, Linien in beliebiger Steigung und Kreise mit beliebigem Durchmesser zu zeichnen. Die Beschränkungen für die Vektoren bleiben jedoch bestehen, da für die Pfeilspitzen die selben Zeichen verwendet werden.

```
\line(Δx,Δy){Länge}
```

Der Befehl entspricht dem Originalbefehl von LaTeX. Allerdings kann für Δx und Δy jetzt jede beliebige ganze Zahl angegeben werden.

```
\circle{Durchmesser}
\circle*{Durchmesser}
```

Auch diese Befehle entsprechen den Originalbefehlen. Für ***Durchmesser*** kann jedoch jede von TeX akzeptierte Länge angegeben werden. Außerdem zeichnet der Befehl den Kreis *exakt* mit dem angegebenen Durchmesser.

```
\oval(Breite,Höhe)[Segment]
```

Dieser Befehl wurde so verändert, daß für den Durchmesser der Kreise an den Ecken jede Größe verwendet werden kann, die von TeX akzeptiert wird. Dieser Wert wird durch Verändern der Variablen `\maxovaldiam` eingestellt (Voreinstellung: 40pt).

Darüberhinaus definiert das `eepic`-Paket alle Zeichenbefehle und -umgebungen des `epic`-Paketes neu. Die Bedienung dieser Befehle ist jedoch identisch mit den Originalversionen. Das Paket ermöglicht nur ein schöneres Aussehen des Ergebnisses.

Neue Befehle des eepic-Paketes

Das `eepic`-Paket stellt verschiedene neue Befehle zur Verfügung.

`\allinethicknes{`*Stärke*`}`

Der `\allinethicknes`-Befehl legt die Strichstärke aller Strichzeichenbefehle fest. Dies betrifft gerade Linien, Kurven, Kreise, Ellipsen, Bögen, Ovale und Splines.

`\Thicklines`

Der Befehl `\Thicklines` stellt die Strichstärke auf das 1.5-fache des Wertes von `\thicklines`.

`\path(`x_1`,`y_1`)(`x_2`,`y_2`)...(`x_n`,`y_n`)`

Der Befehl `\path` ist eine schnelle Variante des `\drawline`-Befehls. Es ist kein optionaler Parameter *Dehn* möglich, daher wird immer eine durchgezogene Linie erzeugt.

`\spline(`x_1`,`y_1`)(`x_2`,`y_2`)...(`x_n`,`y_n`)`

Der Befehl `\spline` zeichnet eine Chaikins-Kurve, die durch den ersten und letzten Punkt geht. Die übrigen Punkte sind nur Kontrollpunkte.

`\ellipse{`*x-Durchmesser*`}{`*y-Durchmesser*`}`
`\ellipse*{`*x-Durchmesser*`}{`*y-Durchmesser*`}`

Analog zu den `\circle`- und `\circle*`-Befehlen. Erzeugt eine leere oder gefüllte Ellipse mit den angegebenen *x-Durchmesser* und *y-Durchmesser*.

`\arc{`*Durchmesser*`}{`*Anfangswinkel*`}{`*Endwinkel*`}`

Der Befehl zeichnet einen Kreisbogen mit dem angegebenen *Durchmesser*. Sowohl *Anfangswinkel*, als auch *Endwinkel* werden in Radianten angegeben. *Anfangswinkel* muß in dem Bereich $[0,\frac{\pi}{2}]$ liegen, *Endwinkel* zwischen *Anfangswinkel* und *Anfangswinkel* $+2\pi$. Die Bögen werden im Uhrzeigersinn gezeichnet.

`\filltype{`*Fülltyp*`}`

Der Befehl legt die Bereichsfüllung von `\circle*`- und `\ellipse*`-Befehlen fest. Erlaubte Werte sind: `black` (Standardwert), `white` und `shade`.

9.2.3 Binäre und ternäre Bäume

Es gibt zwei verschiedene Zusatzpakete, mit denen binäre und ternäre Bäume gezeichnet werden können.

Das Paket `trees`

Das Paket `trees` von Peter Vosroose (VANROOSE, 1990) baut auf der `picture`-Umgebung von LaTeX auf. Mit ihm können Bäume in beliebiger Größe gezeichnet werden. Der Anwender muß dazu innerhalb einer `picture`-Umgebung nur die Knoten mit den Befehlen

```
\branch{Steilheit}{Text}{ID}:TocherA,TochterB.
\tbranch{Steilheit}{Text}{ID}:TocherA,TochterB,TochterC.
```

festlegen. Der Befehl `\branch` wird bei binären Bäumen verwendet, `\tbranch` bei ternären. Der Parameter *Steilheit* legt den Anstieg der Zweige fest. Er wird durch eine ganze Zahl zwischen 0 und 3 angegeben. *Text* wird über dem entsprechenden Punkt eingefügt. *ID* bezeichnet intern die gegenwärtige Verzweigung. *TochterA*, *TochterB* und, bei `\tbranch`, *TochterC* bezeichnen die zwei bzw. drei Tochterknoten. Doppelpunkt, Komma und Punkt sind obligatorisch.

```
\root(X,Y) WurzelID.
```

Der Befehl legt die absoluten Koordinaten der Wurzel des Baumes auf `(X,Y)` fest. Die Klammern, das Leerzeichen vor *WurzelID* und der Punkt sind obligatorisch.

```
\leaf{Kopftext}{Seitentext}{BlattID}
```

Die Parameter *Kopftext* und *Seitentext* werden über bzw. neben dem aktuellen Blatt ausgegeben, das den internen Bezeichner *BlattID* trägt.

Die Bezeichnungen der Zweige des Baumes werden mit dem Befehl

```
\branchlabels{LabelA}{LabelB}{LabelC}
```

festgelegt. Die Voreinstellung ist 0 für den ersten, 1 für den zweiten und 2 für den dritten Zweig.

Abbildung 9.2 auf der nächsten Seite zeigt ein Beispiel eines Baumes, der mit dem `trees`-Paket erstellt wurde. Das Beispiel ist der Dokumentation zu dem Paket entnommen, es wurde nur geringfügig verändert (VANROOSE, 1990).

Die Definition des Baumes lautete:

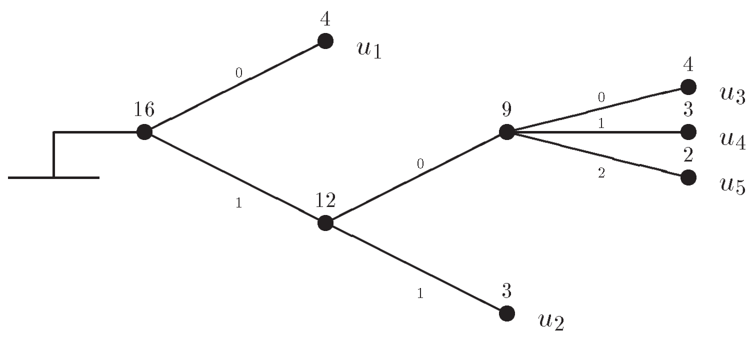


Abbildung 9.2: Beispiel eines Baumes mit `trees`

```
\setlength{\unitlength}{2mm}
\begin{picture}(50,25)
  \thicklines
  \root(5,15) 0.
    \branch{2}{16}{0}:1,2.
      \leaf{4}{$u_1$}{1}.
    \branch{2}{12}{2}:3,7.
      \tbranch{3}{9}{3}:4,5,6.    % Hier wurden flache Äste ausgewählt
          \leaf{4}{$u_3$}{4}.
          \leaf{3}{$u_4$}{5}.
          \leaf{2}{$u_5$}{6}.
      \leaf{3}{$u_2$}{7}.
\end{picture}
```

Das Paket `ecltree`

Das zweite Paket zum Zeichnen von Baumgraphiken ist das Paket `ecltree` von Hideki Isozaka. Das Paket basiert auf dem `epic`-Paket. Das Baumdiagramm wird in einer eigenen Umgebung definiert.

```
\begin{bundle}{Oberknoten}
\end{bundle}
```

Das Argument *Oberknoten* gibt den Text an, der über dem obersten Knoten erscheint. Innerhalb der Umgebung können Knoten mit dem Befehl

```
\chunk[Kantentext]{Knotentext}
```

angelegt werden. *Knotentext* ist der Label für diesen Knoten. Optional kann mit *Kantentext* ein Text angegeben werden, der neben der Kante erscheint. Es können beliebig viele `\chunk`-Befehle verwendet werden. Sie erscheinen alle in der gleichen Ebene. Die genaue Lage der Knoten hängt vom Text ab. Soll der Baum mehrere Ebenen bekommen, so müssen entsprechend viele `bundle`-Umgebungen geschachtelt werden (siehe Abbildung 9.3).

`\drawwith{`*Zeichenbefehl*`}`

Das Aussehen der Kanten kann mit dem Befehl `\drawwith` festgelegt werden. Standardmäßig wird eine durchgezogene Linie verwendet.

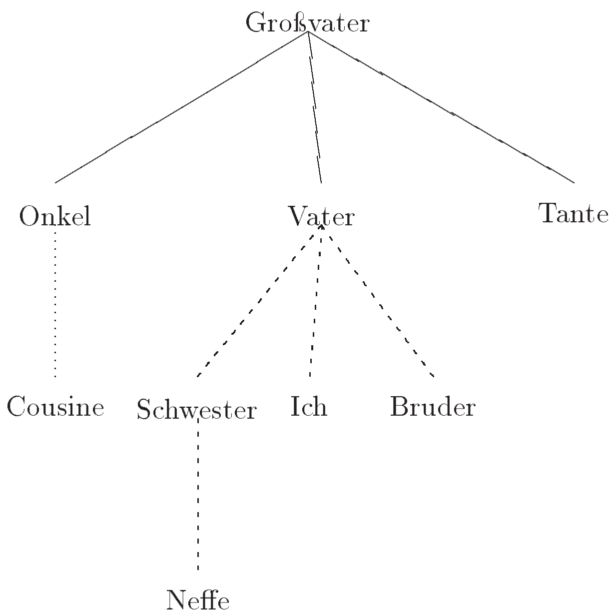


Abbildung 9.3: Beispiel eines Baumes mit `ecltree`

Das Beispiel aus Abbildung 9.3 zeigt ein Baumdiagramm mit mehreren Ebenen. Erzeugt wurde der Baum mit folgenden Befehlen:

```
\setlength{\GapDepth}{20mm}
\setlength{\GapWidth}{8mm}
\begin{bundle}{Großvater}
   \chunk{\begin{bundle}{Onkel}
             \drawwith{\dottedline}
             \chunk{Cousine}
          \end{bundle}}
```

```
    \chunk{\begin{bundle}{Vater}
              \drawwith{\dashline}
              \chunk{\begin{bundle}{Schwester}
                        \chunk{Neffe}
                     \end{bundle}}
              \chunk{Ich}
              \chunk{Bruder}
           \end{bundle}}
    \chunk{Tante}
 \end{bundle}
```

In dem Beispiel wurde zwei Parameter der `bundle`-Umgebung geändert, die die Abstände innerhalb der Umgebung bestimmen.

`\GapDepth`	der minimale vertikale Abstand zweier übereinander liegender Knoten
`\GapWidth`	der minimale horizontale Abstand zweier nebeneinander liegender Knoten
`\EdgeLabelSep`	der Abstand zwischen einem Kantenlabel und dem unteren Knoten einer Kante

9.2.4 Zeichnen beliebiger Kurven

Das `curves`-Paket von Ian Maclaine-cross ermöglicht es, beliebige Kurven zu zeichnen (Maclaine-cross, 1995). Dabei werden die Kurven durch sich überlagernde Punkte erzeugt, deren Größe und Anzahl so gewählt wird, daß optisch ein gleichmäßiges Bild entsteht. Die einzelnen Segmente zwischen den Koordinatenpunkten werden als Parabel angenähert.

```
\arc[NSymbol](x,y){Winkel}
```

Der Befehl zeichnet einen Kreisbogen, der an der aktuellen Position des `\put`-Befehls zentriert ist und bei `(`*x*`,`*y*`)` beginnt. Der Bogen wird gegen den Uhrzeigersinn gezeichnet. *Winkel* wird in Grad angegeben. Das optionale Argument *NSymbol*, das auch von den meisten anderen Befehlen des Paketes akzeptiert wird, beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

```
\bigcircle[NSymbol]{Durckmesser}
```

Der Befehl zeichnet einen Kreis mit dem angegebenen Durchmesser in Einheiten von `\unitlength`.

```
\curve[NSymbol](x1,y1,x2,y2,...xn,yn)
```

Der Befehl `\curve` zeichnet eine Kurve durch die angegebenen Punkte. Zwei Punkte ergeben eine Gerade, drei eine Parabel.

```
\closecurve[NSymbol](x1,y1,x2,y2,...,xn,yn)
```

Dieser Befehl erzeugt eine geschlossene Kurve. Es müssen dazu mindestens sechs Punkte angegeben werden.

```
\tagcurve[NSymbol](x1,y1,x2,y2,...,xn,yn)
```

Dieser Befehl zeichnet eine Kurve ohne das erste und letzte Segment. Werden nur drei Punkte angegeben, so wird das letzte Segment gezeichnet.

```
\scaleput(x1,y1){Bildobjekt}
```

Der Befehl zeichnet *Bildobjekt* an der Position (*x* ,*y*). Gleichzeitig nimmt er eine Parallelprojektion oder Drehung vor, die durch die vier Faktoren `\xscale`, `\xscaley`, `\yscale` und `\yscalex` bestimmt wird. Die Faktoren können mit dem Befehl `\renewcommand` neu festgelegt werden. Die Voreinstellung ist für `\xscale` und `\yscale` 1.0, für `\xscaley` und `\yscalex` 0.0.

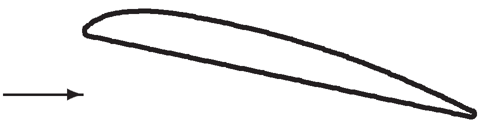
Die folgenden Beispiele wurden der Dokumentation des `curves`-Paketes entnommen (Maclaine-cross, 1995). In den ersten beiden Beispielen wird ein Flügenquerschnitt verwendet. Die Koordinaten des ersten `\curves`-Befehls wurden entsprechenden Strömungstabellen entnommen. Der Befehl `\RAFsixE` wurde folgendermaßen definiert:

```
\newcommand{\RAFsixE}{
  \scaleput(1.25,1.25){\arc(0,-1.25){-135}}
  \scaleput(0,0){\curve(0.366,2.133, 1.25,3.19, 2.5,4.42,
     5.0,6.10, 7.5,7.24, 10,8.09, 15,9.28, 20,9.90, 30,10.3,
     40,10.22, 50,9.80, 60,8.98, 70,7.70, 80,5.91, 90,3.79,
     95,2.58, 99.24,1.52)}
  \scaleput(99.24,0.76){\arc(0,-0.76){180}}
  \scaleput(0,0){\curve(1.25,0, 99.24,0)}
}
```

```
\begin{picture}(100,20)
  \RAFsixE
\end{picture}
```

Das zweite Beispiel zeigt den gleichen Flügel. Allerdings soll er diesmal gedreht werden. Hierzu werden die Skalierungsparameter so geändert, daß die diagonalen Elemente `\xscale` und `\yscale` $\cos(12°)$ entsprechen und die Elemente links und rechts von der Diagonalen $\pm\sin(12°)$:

```
\renewcommand{\xscale}{0.9781}
\renewcommand{\xscaley}{0.2079}
\renewcommand{\yscale}{0.9781}
\renewcommand{\yscalex}{-0.2079}
```



```
\begin{picture}(120,30)(-20,0)
  \put(0,20){\RAFsixE}
  \thicklines
  \put(-20,5){\vector(1,0){20}}
\end{picture}
```

Mit dem `curves`-Paket ist durch Kombination von Drehung und Scalierung auch eine Parallelprojektion möglich. Dabei werden Kreise in Ellipsen umgeformt. In dem Beispiel aus Abbildung 9.4 auf der nächsten Seite, das ebenfalls der Dokumentation entnommen wurde, mußten die Winkel für den `\arc`-Befehl jedoch durch Ausprobieren gefunden werden. Erzeugt wurde das Beispiel mit folgenden Befehlen:

```
\setlength\unitlength{2mm}
\begin{picture}(40,30)
  \thicklines
  \multiput(20,5)(20,12){2}{\line(0,-1){2}\line(-5,3){20}}
  \multiput(20,5)(-20,12){2}{\line(5,3){20}}
  \put(20,3){\line(5,3){20}}
  \put(20,3){\line(-5,3){20}}
  \put(0,15){\line(0,1){2}}
  \linethickness{1pt}
  \put(20,5){
    \renewcommand{\xscale}{1}
    \renewcommand{\xscaley}{-1}
    \renewcommand{\yscale}{0.6}
    \renewcommand{\yscalex}{0.6}
    \scaleput(10,10){\bigcircle{10}}
    \put(0,-2){
```

```
        \scaleput(10,10){\arc(5,0){121}}
        \scaleput(10,10){\arc(5,0){-31}}
        }
      }
  \end{picture}
```

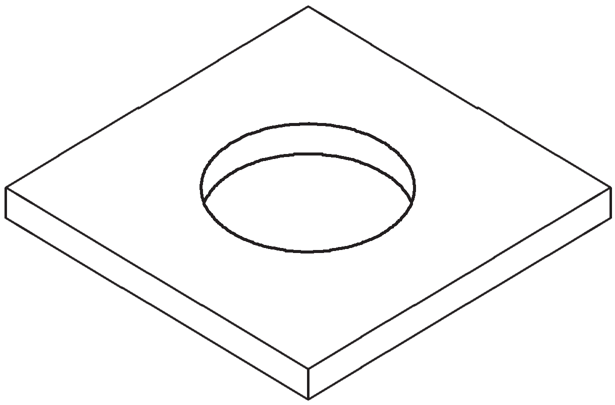


Abbildung 9.4: Parallelprojektion mit dem `curves`-Paket

9.3 LATEX und PostScript

In der Einleitung zu diesem Kapitel wurde bereits erwähnt, daß es mehrere Möglichkeiten gibt, die Fähigkeiten von LATEX und PostScript miteinander zu verbinden. PostScript ist eine sehr mächtige Seitenbeschreibungssprache, die dazu dient, eine ausdruckbare Seite inklusive Text, Linien und Graphiken zu beschreiben. Normalerweise kommt ein Benutzer jedoch mit der eigentlichen Sprache gar nicht in Berührung. Die PostScript-Dateien werden von entsprechenden Programmen automatisch erzeugt.

Eine so erzeugte Datei kann dann auf einem PostScript-fähigen Drucker ausgegeben werden. Alternativ kann die Datei auch mit verschiedenen Programmen angezeigt oder auf Druckern ausgegeben werden, die kein PostScript verstehen.

Um die Funktionen von PostScript für LATEX nutzen zu können, gibt es eine ganze Reihe von Zusatzpaketen. Allen gemein ist, daß sie die entsprechenden PostScript-Befehle in Form von `\special`-Befehlen (vgl. Seite 143) in die `dvi`-Datei schreiben. Diese Datei muß anschließend, zum Beispiel mit dem Programm dvips, in eine PostScript-Datei umgewandelt werden. dvips ist frei verfügbar und für vielen Plattformen erhältlich. Die in diesem Abschnitt beschriebenen Zusatzpakete können alle mit dvips verwendet werden.

9.3.1 Einbindung von PostScript-Dateien

Die einfachste Art, PostScript-Fähigkeiten in LATEX zu nutzen, besteht darin, PostScript-Dateien in das Dokument einzubinden. Die PostScript-Datei darf hierfür nur genau eine Seite haben. In der Regel wird außerdem das sog. „Encapsulated PostScript“ (eps) als Dateiformat verwendet. Dieses Format unterscheidet sich vom normalen PostScript dadurch, daß die sogenannte „BoundingBox“ angegeben wird. Diese Angaben legen fest, wo die Abbildung auf der Seite liegt und wie groß sie ist.

Zum Einbinden von PostScript-Dateien bietet das `epsfig`-Paket von Sebastian Rahtz den Befehl

```
\epsfig{file=Datei ,height=H ,width=B ,clip=,angle=W ,
        silent=,bbllx=blx ,bblly=bly ,bburx=brx ,bbury=bry }
```

Das Argument `file` ist zwingend erforderlich, alle anderen können weggelassen werden. Der Befehl sucht sich aus der angegebenen Datei die relevanten Daten über die „BoundingBox“ und positioniert die Abbildung automatisch. Stehen die Angaben über die „BoundingBox“ nicht in der Datei, oder soll nur ein Teil der Datei angezeigt werden, so kann mit den Argumenten `bbllx=`, `bblly=`, `bburx=` und `bbury=` die Position der linken unteren und der rechten oberen Ecke festgelegt werden. Die Werte werden in „großen“ PostScript-Punkten („bigpoints“) angegeben. Diese entsprechen 1/72 eines Zolls.

Mit den Argumenten `height=` und `width=` kann die Höhe und Breite der Ausgabe festgelegt werden. Werden die Angaben weggelassen, so wird die „natürliche“ Größe der Abbildung verwendet. Wird nur eines der Argumente verwendet, so wird die Abbildung entsprechend skaliert, das Längen-Breiten-Verhältnis bleibt also erhalten. Gibt man jedoch beide Werte an, so kann die Abbildung auch gestaucht oder gestreckt werden.

Die eingebundene Graphik kann gedreht werden. Hierfür wird das Argument `angle=` mit der Winkelangabe in Grad verwendet. Der Winkel wird entgegen dem Uhrzeigersinn gemessen.

Schließlich kennt der Befehl `\epsfig` noch zwei Schalter. Diesen wird zwar kein Wert übergeben, das „=“ ist jedoch zwingend erforderlich. Der Schalter `silent=` unterdrückt eventuelle Meldungen während der Ausführung des `\epsfig`-Befehls. Wird der Schalter `clip=` verwendet, so werden keine Teile, die außerhalb der Boundingbox liegen, ausgegeben. Dies ist insbesondere dann von Interesse, wenn nur ein Ausschnitt einer Datei ausgegeben werden soll.

9.3.2 Abbildungen unterdrücken

PostScript-Dateien können teilweise sehr groß werden. Dementsprechend lange dauert dann auch die Ausgabe über den Drucker oder auf dem Bildschirm. Um die Ausgabe dieser Abbildungen in Vorversionen eines Dokumentes zu unterdrücken, steht der Befehl

```
\psdraft
```

zur Verfügung. Der Befehl wird von dem Paket `graphics` bereitgestellt. Dieses Paket wird von praktisch allen anderen Paketen geladen, die PostScript-Funktionen bieten. Das Paket stellt eine allgemeine Schnittstelle zu PostScript bereit. Somit werden alle Pakete, die auf diesem aufbauen, automatisch auch von allen Druckertreibern unterstützt, die das `graphics`-Paket unterstützen.

Der Befehl schaltet den Draft-Modus ein. In diesem Modus wird zum Beispiel die Ausgabe aller mit `\epsfig` eingebundenen Abbildungen unterdrückt. Stattdessen wird ein Rahmen in der Größe der Abbildung gezogen, in dem der Dateiname steht. Der Befehl wirkt solange, bis der Draft-Modus mit dem Befehl

```
\psfull
```

wieder ausgeschaltet wird.

9.3.3 Rotierende Elemente mit `rotating`

Mit dem `epsfig`-Befehl ist es möglich, externe Abbildungen einzubinden und dabei zu drehen. Um einzelne Teile eines Dokumentes direkt rotieren zu lassen, stellt das `rotating`-Paket von Sebastian Rathz und Leonor Barroca verschiedene Befehle und Umgebungen bereit (Barroca, 1995). Dieses Paket basiert ebenfalls auf dem `graphics`-Paket.

Die Umgebung `rotate` dreht Text oder Graphiken um den als Parameter angegebenen Winkel in Grad gegen Uhrzeigersinn. Dabei wird jedoch keinerlei Platz für den gedrehten Inhalt gelassen.

Hier wird Text `rotating` gedreht.
Dies ist die zweite Zeile.

```
Hier wird Text
\begin{rotate}{-45}
  \texttt{rotating}
\end{rotate}
gedreht. \\
Dies ist die zweite Zeile.
```

Abbildung 9.5 zeigt eine Anwendung für diese Umgebung. Das Beispiel wurde der Dokumentation, die dem Paket beigelegt ist, entnommen (Rathz & Barroca, 1994).

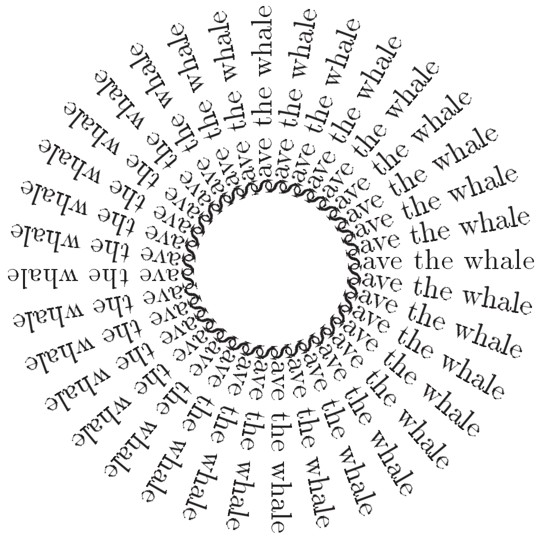


Abbildung 9.5: Mit `rotate` gedrehter Text

Die Befehle, mit denen dieses Beispiel erzeugt wurde, sollen hier nicht verschwiegen werden, wenn auch nicht weiter auf die Bedeutung der verschiedenen Befehle eingegangen werden soll. Sie zeigen jedoch sehr deutlich, daß man durchaus auch mit TeX rechnen kann.

```
\newcount\wang
\newsavebox{\wangtext}
\newdimen\wangspace
\def\wheel#1{\savebox{\wangtext}{#1}%
  \wangspace\wd\wangtext
  \advance\wangspace by 1cm%
  \centerline{%
    \rule{0pt}{\wangspace}%
    \rule[-\wangspace]{0pt}{\wangspace}%
    \wang=-180\loop\ifnum\wang<180
    \rlap{\begin{rotate}{\the\wang}%
        \rule{1cm}{0pt}#1\end{rotate}}%
    \advance\wang by 10\repeat}}
\wheel{Save the whale}
```

Soll LATEX für den gedrehten Inhalt ausreichend Platz lassen, so verwendet man die Umgebung `turn`.

Hier wird Text `turn` gedreht.
Dies ist die zweite Zeile.

```
Hier wird Text
\begin{turn}{-45}
  \texttt{turn}
\end{turn}
gedreht. \\
Dies ist die zweite Zeile.
```

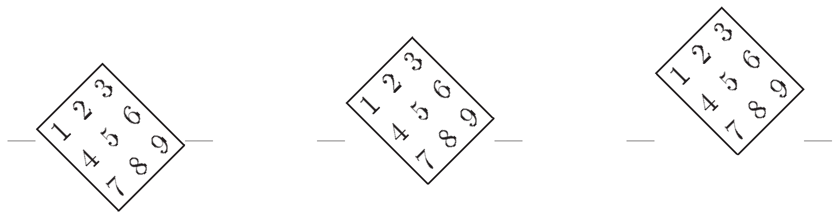
Die Umgebung `sideways` stellt einen Spezialfall dar. Der Umgebung kann kein Winkel angegeben werden, sie verwendet automatisch den Winkel 90°. Außerdem läßt sie ausreichend Platz für den Inhalt.

Dies ist die erste Zeile
Hier wird Text `sideways` gedreht.

```
Dies ist die erste Zeile \\
Hier wird Text
\begin{sideways}
  \texttt{sideways}
\end{sideways}
gedreht.
```

Innerhalb der drei Umgebungen können nahezu alle LATEX-Befehle verwendet werden. In vielen Fällen wird man neben reinem Text oder Graphikelementen auch Absatzboxen verwenden. Hierbei zeigt sich jedoch, daß die Verwendung von TEX-Boxen nicht immer so einfach ist, wie es auf den ersten Blick erscheint. Die Boxen haben eine Höhe *und* eine Tiefe. Die Drehungen erfolgen jedoch um den Punkt an der linken Kante, der auf der Grundlinie liegt (siehe Abb. 9.6 auf der nächsten Seite). Das folgende Beispiel ist jeweils mit der gleichen `turn`-Umgebung erzeugt worden, nur die Boxbefehle unterscheiden sich.

`\parbox[t]{1cm}{...}` `\parbox{1cm}{...}` `\parbox[b]{1cm}{...}`



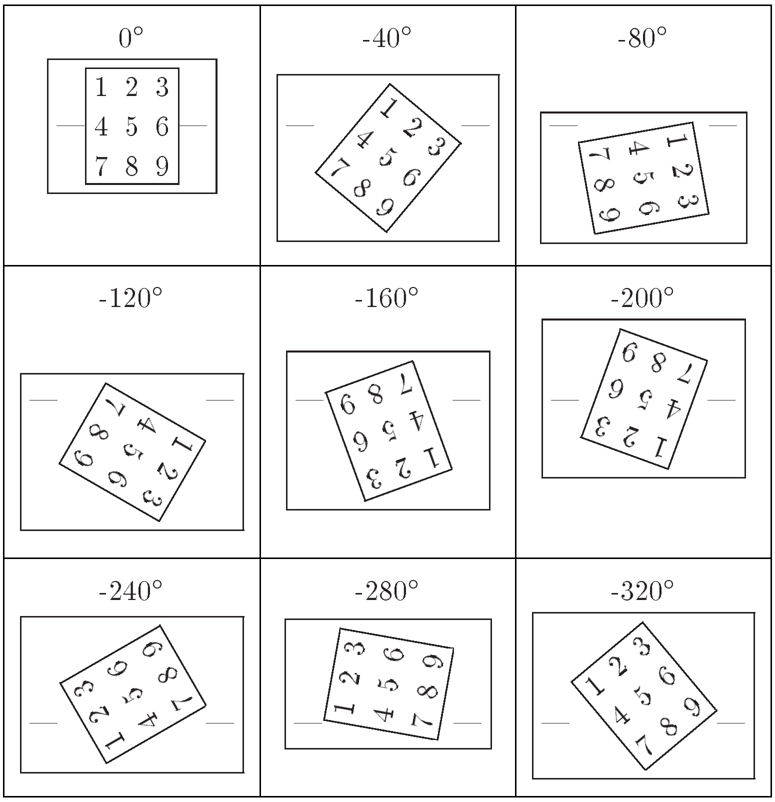


Abbildung 9.6: Gedrehte Absätze

Rotationsumgebungen können in vielen Situationen verwendet werden. So zum Beispiel, um lange Spaltenüberschriften in Tabellen zu drehen, damit die Spalten schmaler werden können.

Spalte 1	Spalte 2	Spalte 3
1	2	3
4	5	6
7	8	9

```
\begin{tabular}{rrr}
 \begin{rotate}{45}
    Spalte 1\end{rotate} &
 \begin{rotate}{45}
    Spalte 2\end{rotate} &
 \begin{rotate}{45}
    Spalte 3\end{rotate} \\
 \hline
 1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\\
 \hline
\end{tabular}
```

In den voranstehenden Beispiel wurde eine `rotate`-Umgebung verwendet. Da LATEX für den gedrehten Inhalt keinerlei Raum freihält, haben die Einträge auch keinen Einfluß auf die Spaltenbreite. Im folgenden Beispiel wird daher die `turn`-Umgebung verwendet. Gleichzeitig werden zentrierte Spalten verwendet.

Spalte 1	Spalte 2	Spalte 3
1	2	3
4	5	6
7	8	9

```
\begin{tabular}{ccc}
 \begin{turn}{45}
    Spalte 1\end{turn} &
 \begin{turn}{45}
    Spalte 2\end{turn} &
 \begin{turn}{45}
    Spalte 3\end{turn} \\
 \hline
 1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\\
 \hline
\end{tabular}
```

Nun erscheinen die Spalten etwas zu breit. Aus diesem Grund wird im nächsten Beispiel erneut die `rotate`-Umgebung verwendet und zusätzlich ein kleiner unsichtbarer Strich eingefügt, der nun die Spaltenbreite festlegt.

Spalte 1	Spalte 2	Spalte 3
1	2	3
4	5	6
7	8	9

```
\begin{tabular}{rrr}
 \begin{rotate}{45}
    Spalte 1\end{rotate}
  \rule{5mm}{0mm} &
 \begin{rotate}{45}
    Spalte 2\end{rotate}
  \rule{5mm}{0mm} &
 \begin{rotate}{45}
    Spalte 3\end{rotate}
  \rule{5mm}{0mm} \\
 \hline
 1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\\
 \hline
\end{tabular}
```

Sollen Gleitobjekte komplett gedreht werden, so kann dazu eine der Umgebungen `sidewaystable` und `sidewaysfigure` verwendet werden. Sie drehen jeweils das gesamte Gleitobjekt, inklusive einer eventuellen Unterschrift. Die gedrehten Objekte haben immer die Breite `\textheight`. Sie belegen also immer eine eigene Seite.

Da dies nicht in jedem Fall gewünscht ist, kann die Abbildung oder Tabelle natürlich auch in einer normalen Gleitumgebung gedreht werden. Soll gleichzeitig die Überschrift auch gedreht werden, so dient hierzu der Befehl

```
\rotcaption{Überschrift}
```

9.3.4 Skalieren von Text

Mit Hilfe des `graphics`-Paketes ist es möglich, Text zu skalieren und zu verzerren. Hierzu dient der Befehl

```
\scalebox{Faktor-Breite}[Faktor-Höhe]{Text}
```

Der Befehl skaliert ***Text*** um den angegebenen Faktor ***Faktor-Breite***. Wird zusätzlich das optionale Argument ***Faktor-Höhe*** angegeben, so wird der Text vertikal anders skaliert als horizontal und der Text demnach verzerrt. Die beiden Skalierungsfaktoren ***Faktor-Breite*** und ***Faktor-Höhe*** dürfen auch negative Werte erhalten, in diesem Fall wird der Text auch gespiegelt. Soll der Text nur um eine vertikale Achse gespiegelt werden, ohne ihn zu verzerren, so kann dafür der Befehl

```
\reflectbox{Text}
```

verwendet werden. Der Befehl entspricht `\scalebox{-1}[1]{Text}`.

---

breiter Text!!!

hoher Text!!!

gespiegelter Text!!!

4cm breit, 1cm hoch!!!

```
\scalebox{2}[1]{breiter Text!!!} \\
\scalebox{1}[2]{hoher Text!!!} \\

\reflectbox{gespiegelter Text!!!} \\

\resizebox{4cm}{1cm}
  {4cm breit, 1cm hoch!!!}
```

In bestimmten Fällen ist der genaue Skalierungsfaktor nicht wichtig, sondern vielmehr die absolute Breite oder Höhe des skalierten Textes. In diesen Fällen wird der Befehl

```
\resizebox*{Breite}{Höhe}{Text}
```

verwendet. Er skaliert `Text` auf die angegebene absolute `Breite` und `Höhe`. Wird für eine der Längenangaben `!` angegeben, so wird für beide Werte der gleiche Skalierungsfaktor verwendet, so daß das Längen-Breiten-Verhältnis erhalten bleibt.

9.4 Farbe in Dokumenten

Für viele Menschen ist Farbe ein wichtiger Bestandteil von Graphiken. Auf nahezu allen Computern finden sich Programme zum Erstellen farbiger Bilder und Abbildungen. Auch viele Textverarbeitungsprogramme bieten derartige Funktionen.

Eine farbige Abbildung in ein LaTeX-Dokument einzubinden, ist mit den in Abschnitt 9.3.1 vorgestellten Werkzeugen kein Problem. Um farbige Elemente direkt in einem Dokument zu erzeugen, stellt das Paket `color` eine allgemeine Schnittstelle zur Verfügung. Zahlreiche Pakete, so zum Beispiel das auf Seite 112 vorgestellte `colortbl`-Paket, basieren auf dieser Schnittstelle.

9.4.1 Farbmodelle und Farbdefinition

Die Farbdefinition, die das Paket verwendet, basiert auf verschiedenen Farbmodellen. Welche Farbmodelle von dem jeweiligen `dvi`-Treiber unterstützt werden, ist der entsprechenden Dokumentation zu entnehmen. Die Zahl der `dvi`-Treiber, die das `color`-Paket unterstützen, wächst ständig, daher kann hier nicht weiter darauf eingegangen werden. Die folgenden Farbmodelle werden jedoch von den meisten Treibern unterstützt.

`rgb` Das `rgb`-Farbmodell basiert auf der „additiven Mischung“ (genauer: der Überlagerung) der drei Farben *Rot*, *Grün* und *Blau*. *Additiv* bedeutet, daß volle Anteile der drei Grundfarben zusammen Weiß ergeben. Dies ist zum Beispiel der Fall, wenn drei Scheinwerfer mit den Farben auf einen gemeinsamen Punkt scheinen.

`cmyk` Dieses Farbmodell wird vor allem in der Druckindustrie verwendet. Es basiert auf der „subtraktiven Mischung“ der vier Farben *Cyan*, *Magenda*, *Yellow* und *Black*. Die ersten drei sind die jeweiligen Komplementärfarben zu den im RGB-Modell verwendeten Farben. *Cyan* entspricht also der Farbe „Weiß minus Rot“. Diese Betrachtung entspricht durchaus der Realität, denn eine Farbfläche der Farbe *Cyan* absorbiert aus dem einfallenden weißen Licht die Farbe *Rot*. Eine Überlagerung der beiden Farben *Cyan* und *Magenta* ergibt somit genaugenommen auch keine *Mischung* dieser Farben. *Cyan*, verstanden als „Weiß minus Rot“, überlagert von

Magenta, verstanden als „Weiß minus Grün", ergibt also „Weiß minus Rot minus Grün", also „Blau". Eine blaue Fläche absorbiert sowohl grünes, als auch rotes Licht.

`gray` Dieses Farbmodell dient nur dazu, auf vereinfachte Weise Graustufen zu erzeugen. Prinzipiell können Graustufen auch mit den beiden vorangestellten Farbmodellen erzeugt werden, so z.B. durch gleiche Anteile von Rot, Grün und Blau im RGB-Farbmodell.

Definition neuer Farben

Die Farbnamen `black`, `white`, `red`, `green`, `blue`, `cyan`, `magenta` und `yellow` sollten in einer Standdardinstallation bereits definiert worden sein. Neue Farben können jederzeit mit dem Befehl

```
\definecolor{Name}{Modell}{Farbdefinition}
```

erzeugt und verfügbar gemacht werden. ***Name*** ist der Name, unter dem die neue Farbe verwendet werden soll. Es kann dafür ein neuer oder auch ein bereits bestehender Name gewählt werden. ***Modell*** wählt das Farbmodell aus, mit dem die Farbe definiert werden soll. ***Definition*** schließlich ist eine Zahl bzw. eine durch Kommas getrennte Liste von Zahlen zwischen 0 und 1, die den Anteil der jeweiligen Farbkomponenten angibt.

9.4.2 Verwendung von Farbe

Der einfachste Befehl zum Nutzen von Farbe in Dokumenten ist der Befehl

```
\color{Farbe}
```

Von seinem Auftreten an wird sämtlicher nachfolgender Text bis zum Ende der aktuellen Umgebung in ***Farbe*** gedruckt. ***Farbe*** muß dabei, wie auch in allen nachfolgenden Befehlen, ein vorher definierter Farbname sein. Alternativ kann die Farbe auch direkt definiert werden.

```
\color[Modell]{Definition}
```

In Analogie zu den Zeichensatzbefehlen kann für einen kurzen farbigen Text auch der Befehl

```
\textcolor{Farbe}{Text}
```

verwendet werden. Der Befehl druckt ***Text*** in der angegebenen ***Farbe***. Auch dieser Befehl steht in einer zweiten Form zur Verfügung, bei der die Farbe im Befehlsaufruf definiert wird.

\textcolor[*Modell*]{*Definition*}

Dieser Text ist dunkel!
Dieser Text ist zweifarbig!

```
\definecolor{hell}{gray}{0.7}
\definecolor{dunkel}{gray}{0.4}

\textcolor{dunkel}
  {Dieser Text ist dunkel!} \\
{\color{hell}Dieser Text
  \color[gray]{0.2}ist zweifarbig!}
```

Die Hintergrundfarbe der gesamten aktuellen und aller nachfolgender Seiten wird mit dem Befehl

\pagecolor{*Farbe*}

umgestellt. Dieser Befehl ist global, das bedeutet, er wirkt auch über das Ende der aktuellen Umgebung hinaus und kann nur durch einen weiteren `\pagecolor`-Befehl aufgehoben werden.

Farbige Boxen können mit den Befehlen

\colorbox{*Farbe*}{*Text*}
\fcolorbox{*Rahmen*}{*Farbe*}{*Text*}

erzeugt werden. ***Text*** wird jeweils in der aktuellen Vordergrundfarbe besetzt. Bei Bedarf kann sie mit entsprechenden Befehlen auch in der Box verändert werden. Die Rahmenbreite der `\fcolorbox` entspricht `\fboxrule`, der Abstand zwischen Boxinhalt und Rahmen entspricht `\fboxsep`.

Weißer Text!

Farbige Rahmenbox mit dunklem Rahmen

```
\setlength{\fboxrule}{2mm}
\setlength{\fboxsep}{1.5mm}

\colorbox{dunkel}
  {\color{white} Weißer Text!} \\

\fcolorbox{dunkel}{hell}{%
  \parbox{4cm}{Farbige Rahmenbox
    mit dunklem Rahmen}}
```

9.4.3 Mögliche Probleme bei der Verwendung von Farbe

TEX selbst bietet keinerlei Möglichkeiten des Farbdrucks. Somit bleibt dies den `dvi`-Treibern überlassen. Dabei können jedoch Probleme Auftreten, da nicht alle diese Treiber die entsprechenden `\special`-Befehle in der gleichen Weise verarbeiten. Beispielsweise können nicht alle Treiber die Farbe bei einem Seitenumbruch beibehalten. Möchte man also einen Absatz farbig schreiben, kann es je nach Treiber sein, daß der Text nach einem Seitenumbruch fälschlicherweise wieder in Schwarz erscheint (Carlisle, 1998b).

Ein weiteres Problem, das bei allen Treibern besteht, ist, daß an den Stellen, an denen die Farbe wechselt, die Breite von Leerzeichen beeinflußt wird. Somit kann es passieren, daß ein Text *mit* farbigen Elementen anders gesetzt wird, als der selbe Text *ohne* diese Elemente.

Anhang A

Fehlermeldungen

Jeder LaTeX-Nutzer wird früher oder später einmal Fehler machen, schließlich ist niemand perfekt. Als Fehler können einfache Tippfehler auftreten oder auch größere Probleme in komplexeren Strukturen. LaTeX gibt, wenn es beim Setzen auf einen Fehler trifft, eine entsprechende Fehlermeldung aus.

Da diese besonders für den LaTeX-Neuling am Anfang sehr verwirrend aussehen, soll hier zunächst die allgemeine Struktur einer Fehlermeldung kurz erläutert werden. Im zweiten Teil des Kapitels werden dann die wichtigsten Fehlermeldungen behandelt. Der letzte Teil enthält eine Aufstellung der häufigsten LaTeX-Warnungen, die oftmals einen Hinweis auf eventuelle Probleme geben können.

A.1 Grundstruktur der Fehlermeldungen

Prinzipiell gibt es zwei Arten von Fehlermeldungen, diejenigen, die von LaTeX stammen, und die, die von TeX stammen. Häufig ziehen LaTeX-Fehler eine Reihe von TeX-Fehlern nach sich. Dies liegt daran, daß jeder LaTeX-Befehl letztlich aus einer Reihe von TeX-Befehlen aufgebaut ist. Der Unterschied soll hier jedoch nicht näher beleuchtet werden. Erkennbar sind die beiden Arten am unterschiedlichen Fehlerindikator: Eine TeX-Fehlermeldung beginnt mit einem `!`, eine Meldung von LaTeX beginnt mit `! LaTeX Error`.

Zunächst ein einfaches Beispiel: LaTeX stößt während des Setzens eines Textes auf folgende Zeile:

```
Wir wollen betonen, daß \enph{alle} Anwesenden essen!
```

In dieser Zeile ist der Befehl `\emph` aus Versehen als `\enph` geschrieben worden. Da LaTeX diesen Befehl nicht kennt wird er ohne Interpretation an TeX weitergereicht. Auch hier ist dieser Befehl unbekannt und es wird folgende Fehlermeldung ausgegeben:

```
! Undefined control sequence
l.74 Wir wollen betonen, daß \enph
                                  {alle} Anwesenden essen!
?
```

Eine solche Fehlermeldung ist sicherlich für jeden Anwender leicht verständlich. Die Meldung beginnt mit einem Fehlerindikator (`! Undefined control sequence`), der besagt, daß der verwendete Befehl nicht definiert worden ist. Die zweite Zeile der Meldung zeigt zunächst, in welche Zeile der Quelldatei den Fehler verursacht hat. In diesem Fall ist es Zeile 74. Bei der Fehlermeldung wird jedoch nicht angegeben, in welcher *Datei* der Fehler aufgetreten ist. Tritt der Fehler in einem Zusatzpaket auf, so wird die Zeile in diesem Paket angegeben. In diesem Fall kann es schwierig werden, den fehlerhaften Text zu finden. Ist ein Dokument auf mehrere Dateien verteilt, so wird beim Formatieren jeweils angezeigt, wenn eine weitere Datei gerade geladen wird. Ein Blick auf die Zeilen vor der Fehlermeldung geben darüber Auskunft.

An die Zahl schließt sich der Text der Zeile an bis einschließlich zu der Stelle, an der der Fehler aufgetreten ist. In der Zeile darunter folgt dann der restliche Text der Eingabezeile. Mit dem Fragezeichen am Schluß der Fehlermeldung und dem an dieser Stelle blinkenden Cursor zeigt TEX an, daß es auf eine Eingabe des Nutzers wartet.

Gibt man als Eingabe ein `?` ein, so erhält man eine Liste der möglichen Eingaben:

```
Type <return> to proceed, S to scroll future error messages,
R to run without stopping, Q to run quietly,
I to insert something, E to edit your file,
1 or ... or 9 to ignore next 1 to 9 tokens of input,
H for help, X to quit
?
```

Am Ende wartet TEX wieder auf eine Eingabe. Die Möglichkeiten bedeuten im Einzelnen:

`<return>`: Durch Drücken der Returntaste fährt das Programm mit der Bearbeitung des Textes fort. Eventuell wird versucht, den Fehler zum Beispiel durch Einsetzen von Text zu beheben. In dem gewählten Beispiel würde der Fehler einfach übergangen.

`S` „scroll mode“: TEX fährt mit der Bearbeitung des Dokumentes fort. Treten weitere Fehler auf, so werden entsprechende Meldungen ausgegeben, TEX hält jedoch nicht mehr an. Das Ergebnis ist das gleiche, als wenn bei jedem auftretenden Fehler <return> gedrückt würde.

`R` **„run mode“:** Dieser Fall ist ähnlich wie der „scroll mode“, allerdings stoppt TEX auch dann nicht, wenn eine nicht auffindbare Datei über einen `\include` oder `\input` Befehl eingelesen werden soll. Im „scroll mode“ würde bei einem solchen Fehler gestoppt, so daß man den richtigen Namen, eventuell mit Pfadangabe, eingeben kann.

`Q` **„quiet mode“:** Wie `R`, jedoch erscheinen keine weiteren Meldungen auf dem Bildschirm. Sie werden aber in der Protokolldatei abgespeichert.

`I` **„insert“:** Hier kann man Text eingeben. TEX fügt ihn vor das nächste noch nicht verarbeitete Zeichen der Quelldatei ein und fährt fort. Der Text wird jedoch *nicht* in der Quelldatei eingefügt. `I\stop` beendet die Bearbeitung, die laufende Seite erscheint in der Ausgabedatei.

`1...`**:** Mit Eingabe einer Zahl kleiner als 100 kann die Interpretation der nächsten Zeichen übersprungen werden. Danach stoppt LATEX erneut, um auf eine weitere Eingabe des Nutzers zu warten.

`H` **„help“:** TEX bzw. LATEX gibt eine ausführlichere Fehlerbeschreibung und eventuell eine Empfehlung, wie der Fehler behoben werden kann.

`X` **„exit“:** Das Formatieren wird abgebrochen, die laufende Seite erscheint nicht in der Ausgabedatei.

`E` **„edit“:** Das Formatieren wird abgebrochen wie bei `X`. Danach wird der Editor aufgerufen. Der Cursor steht am Anfang der Zeile, in der der Fehler aufgetreten ist. Diese Funktion ist nicht auf allen Systemen implementiert. Außerdem muß man aufpassen, wenn gleichzeitig ein anderer Editor mit der gleichen Datei aktiv ist. In diesem Fall kann es zu Inkonsistenzen kommen.

Bei der Eingabe spielt Groß- und Kleinschreibung keine Rolle.

Wird in unserem Beispiel ein `H`, gefolgt von der Returntaste, eingegeben, so erhält man als nächste Ausgabe:

```
The control sequence at the end of the top line of your error message
was never \def'ed. If you have misspelled it (e.g., `\hobx'), type `I'
and the correct spelling (e.g., `I\hbox'). Otherwise just continue
and I'll forget about whatever was undefined.

?
```

Auch diese Meldung ist leicht verständlich. In dem gewählten Beispiel kann man also `I\emph` eingeben, und das Ergebnis wird wie gewünscht gesetzt.

Drückt man nur Return, so wird der Befehl ignoriert und der Text genauso gesetzt wie wenn die Eingabezeile gelautet hätte:

```
Wir wollen betonen, daß alle Anwesenden essen!
```

Im Gegensatz zur alten Version LaTeX2.09 werden komplizierte Fehlermeldungen von LaTeX 2$_\varepsilon$ unterdrückt. Schreibt man im Quelltext z. B.

```
\begin{centre}
  Zentrierter Text
\end{centre}
```

so erzeugt dies folgende Fehlermeldung:

```
! LaTeX error. Environment centre undefinded

See LaTeX manual or LaTeX Companion for explanation.
Type H<return> for immediate help.
 ...

l.543 \begin{centre}

?
```

Die erste Zeile enthält auch hier die wichtigste Information. Die Punkte zeigen Text an, der für einen normalen LaTeX-Benutzer nicht nötig ist. Erhält man dennoch einmal eine Fehlermeldung, die unverständlichen Text erhält, so kann man ihn getrost übergehen.

A.2 Liste der häufigsten Fehlermeldungen

In diesem Abschnitt werden die wichtigsten Fehlermeldungen, die LaTeX und TeX ausgeben, in alphabetischer Reihenfolge aufgeführt. Es wird sicherlich noch weitere, seltenere Fehlermeldungen geben. Dies gilt vor allem beim Einbinden zusätzlicher Pakete.

`! Argument of \line has an extra }.`

Die Argumente für einen `\line` oder `\vector` Befehl sind nicht korrekt. In der Regel wurde nur eine Koordinate für die Richtung angegeben.

`! Double subscript.`

Im mathematischen Modus traten zwei Tiefstellungsbefehle hintereinander ohne eine korrekte Klammerung auf. Möchte man z. B. a_{b_c} erreichen, so muß man `$a_{b_c}$` oder `$a_{b_{c}}$` schreiben. `$a_b_c$` führt zu der angegebenen Fehlermeldung.

`! Double superscript.`

Diese Meldung ist analog zu der vorigen zu verstehen.

`! Extra alignment tab has been changen to \cr.`

In einer `tabular`- oder `array`-Umgebung wurden in einer Zeile zu viele `&`-Befehle entdeckt. Häufigste Ursache ist entweder, daß zu wenige Spalten definiert worden sind, oder daß ein Zeilenende vergessen wurde.

`! Extra }, or forgotten $.`

Im mathematischen Modus wurde entweder eine linke Klammer `{` vergessen oder eine rechte Klammer `}` zu viel gesetzt oder ein Umschaltzeichen in den mathematischen Modus (z. B. `[`, `$`) wurde vergessen.

`! File ended while scanning use of ...`

Es wurden Paramter für einen Befehl weggelassen, oder die schließenden Klammern für die Parameter fehlen.

`! Font ... not loaded: Not enough room left.`

In dem Dokument sollen mehr Zeichensätze verwendet werden, als TEX Speicherplatz dafür bereithält. In diesem Fall sollte man entweder das Dokument auf verschiedene einzelne Dateien verteilen, die dann getrennt verarbeitet werden, oder man sollte versuchen, die Zahl der verwendeten Schriften zu reduzieren. Computerfreaks bietet sich noch die Möglichkeit, TEX neu zu compilieren und vorher die maximale Zahl der Schriften zu erhöhen.

`! I can't find file ... .`

In dem Dokument sollte eine Datei geladen werden, die TEX nicht finden kann. Endet der Name auf `.tex`, so handelt es sich in der Regel um einen Teil des Dokumentes, der mit einem `\input` oder `\include` Befehl eingebunden werden sollte.

Endet der Name auf `.sty` oder `.cls`, so kann TEX ein zusätzliches Paket oder eine Klassendatei nicht finden. TEX sucht nach den Dateien in dem Verzeichnis, aus dem heraus es aufgerufen wurde, sowie in den im Suchpfad angegeben Dateien.[1]

[1] Je nach TEX-Installation kann dies unterschiedlich sein. Das gerade bei Linux weit verbreitete Paket teTEX verwendet der schnelleren Auffindbarkeit der Dateien wegen eine Indexdatei. Es können nur die Dateien gefunden werden, die in dieser Liste aufgeführt sind. Zu dem Paket gehört das Programm `texhash`, das diese Indexdatei jederzeit erneuern kann.

Am Ende der Fehlermeldung erscheint als weitere Zeile:

`Please type another input file name:`

An dieser Stelle wartet TEX auf die Eingabe des korrekten Namens einschließlich des (relativen oder absoluten) Pfades.

`! Illegal parameter number in definition of ... .`
In der Definition eines Befehls oder einer Umgebung wurde die Anzahl der Argumente falsch angegeben. Die vier Befehle `\newcommand`, `\renewcommand`, `\newenvironment` und `\renewenvironment` bekommen als Argument die Zahl der Argumente für den späteren Aufruf. Wenn in der Definition des Befehls oder der Umgebung mehr Argumente verwendet werden, als am Anfang angegeben wurden, erscheint obige Meldung.

`! Illegal unit of measure (pt inserted).`
Tritt dieser Fehler alleine auf, so bedeutet dies, daß an einer Stelle entweder eine Maßeinheit vergessen wurde, oder daß man eine falsche Maßeinheit verwendet hat. Am häufigsten wird dies bei einer Länge null auftreten. Auch hier erwartet TEX die Angabe einer Längeneinheit. In diesem Fall wird die Datei korrekt bearbeiten, wenn man die Returntaste drückt.

Der Fehler kann auch direkt nach der Meldung

`! Missing number, treated as zero.`

auftreten. In diesem Fall ist die Meldung eine direkte Folge der vorigen (siehe unten).

`! LaTeX Error: Bad \line or \vector argument.`
Das erste Argument eines `\line` oder `\vector`-Befehls enthält ein nicht erlaubtes Wertepaar.

`! LaTeX Error: Bad math environment delimiter.`
LATEX hat entweder in einem normalen Text einen Beendigungsbefehl für den mathematischen Modus wie z. B. `\]` oder `\)` gefunden, oder es wurde innerhalb des mathematischen Modus ein weiterer Anfangsbefehl wie `\begin{displaymath}` oder `\[` gefunden. Entweder liegt die Ursache in nicht zusammenpassenden Umschaltbefehlen für den mathematischen Modus oder in einer falschen Klammerstruktur.

`! LaTeX Error: \begin{...} on input line X ended by \end{...}.`
Eine begonnene Umgebung wurde nicht korrekt beendet. Entweder wur-

de der `\end{...}`-Befehl falsch geschrieben, weggelassen oder ein zusätzlicher eingegeben. Als Vorbeugung sollte man sich angewöhnen, bei jeder Umgebung zunächst das Befehlspaar `\begin` und `\end` zu schreiben und dann den Text der Umgebung. Bei der Verwendung des auctex-Paketes für die GNUemacs-Editor werden beide Befehle automatisch gesetzt, so daß hier Fehler nahezu ausgeschlossen sind.

`! LaTeX Error: Can be used only in preamble.`
Es wurde ein Befehl nach dem Befehl `\begin{document}` eingegeben, der nur im Vorspann erlaubt ist. Es kann auch der Fall sein, daß der Befehl `\begin{document}` in dem Dokument mehrmals auftritt.

`! LaTeX Error: Command ... already defined.`
Es sollte ein neuer Befehl, Zähler, o. ä. definiert werden, dessen Name bereits existiert. Entweder muß man sich einen neuen Namen suchen, oder ein bereits bestehender Befehl soll mit `\renewcommand` umdefiniert werden.

`! LaTeX Error: Counter too large!`
Ursache für diese LaTeX-Meldung ist, daß man einen Zähler mit Buchstaben numeriert und der Zähler größer als 26 geworden ist. In diesem Fall werden keine weiteren Nummern mehr ausgegeben. Der Fehler wiederholt sich jedesmal, wenn der entsprechende Zähler erhöht wird. Um den Fehler zu beheben, stellt man einfach die Art der Numerierung um.

`! LaTeX error: Environment ... undefined.`
Ein `\begin`-Befehl enthält einen unbekannten Namen für eine Umgebung. Wurde der Name nur falsch geschrieben, so kann man ihn mit der Eingabe von `I` *`Richtiger Name`* korrigieren. Eventuell wurde auch vergessen ein entsprechendes Paket, welches die Umgebung bereitstellt, zu laden. In diesem Fall kann es zu einer ganzen Reihe weiterer Fehlermeldungen kommen, so daß ein Abbruch der Bearbeitung sinnvoll sein kann.

`! LaTeX Error: Illegal character in array arg.`
Es wurde ein falsches Zeichen bei der Spaltendeklaration einer `tabular` oder `array`-Umgebung oder in einem `\multicolumn`-Befehl verwendet.

`! LaTeX error: Missing \begin{document}.`
Entweder wurde der Befehl `\begin{document}` tatsächlich vergessen, oder der Vorspann bzw. ein im Vorspann geladenes Paket enthält ausdruckbaren Text. In diesem Fall liegt die Ursache in der Regel in falschen Klammerstrukturen.

`! LaTeX error: No counter '...' defined.`

In einem `\setcounter` oder `\addtocounter` Befehl taucht ein nicht definierter Zähler auf. Dieser Fehler kann auch beim Lesen der `.aux`-Datei auftreten, wenn darin ein Zähler auftritt, der erst zu einem späteren Zeitpunkt im Dokument definiert wird. Deshalb sollten `\setcounter` Befehle nur im Vorspann auftreten.

`! LaTeX Error: Not in outer par mode.`

Eine `parbox` oder eine `minipage`-Umgebung enthält ein unerlaubtes Gleitobjekt (`figure`, `table`) oder einen `\marginpar`-Befehl. Dieser Fehler kann auch auftreten, wenn die genannten Umgebungen im mathematischen Modus auftreten.

`! LaTeX Error: Something's wrong-perhaps a missing \item.`

Diese Meldung kann verschiedene Ursachen haben. Die häufigste liegt darin, daß in einer Listenumgebung der Text nicht mit dem Befehl `\item` beginnt. Es kann auch sein, daß eine Liste nicht korrekt definiert wurde, oder daß bei einem Literaturverzeichnis die Mustermarke vergessen wurde.

`! LaTeX Error: Tab overflow.`

Es wurden mehr Tabulatorstopps gesetzt, als von LaTeX erlaubt.

`! LaTeX Error: There's no line here to end.`

Es wurde ein Zeilenendebefehl zwischen zwei Absätzen oder am Ende einer Umgebung, die automatisch einen Absatz beendet, entdeckt. Möchte man an dieser Stelle eine Leerzeile einfügen, so kann man neben den Platzbefehlen `\smallskip`, `\medskip`, `\bigskip`, `\vspace{...}` oder `\rule` auch ein Leerzeichen fordern: `~\\`.

`! LaTeX Error: This may be a LaTeX bug.`

Hier ist LaTeX vollständig durcheinander geraten. Die häufigste Ursache für eine solche Meldung ist die, daß bei vorigen Fehlermeldungen immer nur mit return reagiert wurde. In diesem Fall sollte die Bearbeitung des Textes abgebrochen und die früheren Fehler behoben werden. Tritt der Fehler dennoch auf, so handelt es sich um den recht unwahrscheinlichen Fall, daß ein wirklicher Fehler im Programm aufgetreten ist. In diesem Fall sollte man die Datei kopieren, soweit reduzieren wie möglich und dann zusammen mit der Protokolldatei möglichst an die Bezugsquelle, von der man LaTeX erhalten hat, weiterleiten.

`! LaTeX Error: Too deeply nested.`

Die Meldung besagt, daß zu viele Listenumgebungen ineinander verschachtelt wurden. Die Schachtelungstiefe ist abhängig von der Rechnerplattform. Vier Ebenen sind jedoch immer vorhanden.

`! LaTeX Error: Too many unprocessed floats.`

LaTeX hat mehr unbearbeitete Gleitobjekte, als dafür Speicher vorgesehen ist. Entweder wurden auf einer Seite zu viele `\marginpar`-Befehle verwendet, oder es wurden zu viele, zu große gleitende Bilder und Tabellen in zu dichtem Abstand verwendet.

Im ersten Fall muß man einige der Befehle streichen. Im zweiten sollte man an verschiedenen Stellen im Dokument die Ausgabe sämtlicher aufgelaufener Gleitobjekte mit einem der beiden Befehle `\clearpage` oder `\cleardoublepage` erzwingen.

`! LaTeX Error: Undefined tab position.`

In einer `tabbing`-Umgebung treten in einer Zeile zu viele `\>`-Befehle auf. Eventuell wurde ein Zeilenende vergessen.

`! LaTeX Error: \< in mid line.`

Dieser Befehl darf nur in einer `tabbing`-Umgebung am Anfang und nicht mitten in einer Zeile auftreten.

`! Misplaced alignment tab character &.`

Das Zeichen `&`, das nur in einer `tabular`- oder `array`-Umgebung als Steuerzeichen verwendet werden darf, trat in normalem Text auf. Wahrscheinlich wollte man ein & ausdrucken. Dies erreicht man mit dem Befehl `\&`. Ist dies der Fall, so kann man mit der Eingabe `I\&` das Dokument korrekt weiterbearbeiten lassen.

`! Missing number, treated as zero.`

Die häufigste Ursache für diese Meldung ist, daß bei einem LaTeX-Befehl eine Längen- oder Zahlangabe in einem Argument vergessen wurde. Der Fehler kann auch durch einen Befehl auftreten, dem man am Ende einen optionalen Parameter übergeben kann, und der von einem mit `[` beginnendem normalen Text gefolgt wird.

`! Missing { inserted.`
`! Missing } inserted.`

Tritt eine dieser beiden Meldungen auf, so stimmen die Klammerstrukturen im Dokument nicht. Die fehlenden Zeichen gehören in der Regel nicht in die Zeile, die in der Fehlermeldung angegeben wird. Findet man die Stelle nicht, an der die Klammern fehlen, so sollte man bei der Fehlermeldung die Weiterbearbeitung des Dokumentes mit Return erreichen und den Probeausdruck analysieren.

`! Not a letter.`

Diese Meldung erscheint, wenn in einem `\hyphenation`-Befehl ein verbotenes Zeichen verwendet wurde. In diesem Fall sollte man die Silbentrennung auf andere Art erreichen (siehe 3.2.5).

`! Paragraph ended before ... was complete.`

Ein Argument für einen Befehl enthält eine unerlaubte Leerzeile oder einen `\par`-Befehl. Wahrscheinlich wurde eine Klammer `}` vergessen.

`! Text line contains an invalid character.`

Die Eingabezeile des Dokumentes enthält ein Zeichen, daß TEX unbekannt ist. Meistens liegt die Ursache an einer Fehlbedienung des Editors. Kann man das fehlerhafte Zeichen nicht finden, so sollte man den Text einmal mit einem anderen Editor laden.

`! Too many }'s.`

TEX hat eine schließende Klammer `}` entdeckt, zu der keine öffnende Klammer (`{`) gehört. Entweder wurde die öffnende vergessen, was zu einer ganzen Reihe weiterer Fehlermeldungen oder Warnungen führen kann, oder es wurde einfach eine schließende zu viel gesetzt. In diesem Fall kann ohne Probleme die Return-Taste gedrückt werden und die Datei wird korrekt bearbeitet.

`! Undefined control sequence.`

Dies ist eine der häufigsten Fehlermeldungen. In der Regel wurde einfach nur ein Befehlsname falsch geschrieben oder ein Befehl, der in einem Zusatzpaket geladen wird, wurde verwendet ohne daß das Paket geladen wurde. Ist der Befehl nur falsch geschrieben worden, so kann man mit `I`*`Richtiger-Name`* die korrekte Weiterbearbeitung des Dokumentes erreichen.

Der Fehler kann auch auftreten, wenn Befehle, die nur innerhalb einer bestimmten Umgebung definiert sind, außerhalb der Umgebung verwendet wurden, oder wenn der Befehl `\documentclass` vergessen wurde.

`! Use of ... doesn't match its definition.`

Meistens tritt diese Meldung bei der falschen Verwendung von Bildbefehlen auf. Ist der Name des Befehls `@array`, dann liegt der Grund für den Fehler in einem falschen `@`-Ausdruck einer `tabular`- oder `array`-Umgebung. Eventuell wurde bei einem zerbrechlichen Befehl ein `\protect` vergessen.

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

TEX hat in normalem Text das Sonderzeichen `#` entdeckt, das nur in der

Definition von Befehlen oder Umgebungen erlaubt ist. Ein # erhält man mit `\#`.

A.3 Warnungen

Neben Fehlermeldungen, die immer eine Unterbrechung der Bearbeitung durch LaTeX hervorrufen, werden eine Reihe von Warnungen ausgegeben. Diese Warnungen deuten an, daß die Ausgabedatei eventuell noch Mängel, wie z. B. noch nicht aufgelöste Referenzen, enthält. Die Warnungen werden ebenfalls in der Protokolldatei abgelegt und können im Anschluß an die Bearbeitung für Feinarbeiten genutzt werden.

A.3.1 LaTeX-Warnungen

Die Warnungen, die von LaTeX ausgegeben werden, beginnen alle mit „`LaTeX Warning`". Anschließend erscheint eine der folgenden Meldungen:

`Citation ‘...’ on page ... undefined.`
: In einem `\cite`-Befehl wurde ein Schlüsselwort verwendet, daß nicht durch einen `\bibitem`-Befehl definiert wurde. Da LaTeX die Schlüsselworte in einer Hilfsdatei (`.aux`) speichert, die am Anfang der Formatierung gelesen wird, und sich damit die Angaben über Referenzen u. ä. auf die jeweils letzte Bearbeitung beziehen, kann es sein, daß die Warnung beim nächsten Durchlauf nicht mehr auftritt.

`Font shape ‘...’ undefined`
`using ‘...’ instead on input line ... .`
: Es wurde eine Schrift angefordert, die nicht definiert ist. Sie wurde durch eine andere ersetzt.

`Label ‘...’ multiply defined.`
: Bei zwei `\label`- oder `\bibitem`-Befehlen wurde dieselbe Markierung verwendet. Diese Warnung erscheint beim nächsten LaTeX-Lauf erneut, da die Markierungen in der `.aux`-Datei abgelegt wurden und somit am Anfang erneut gelesen werden. Wenn nötig, kann man auch die `.aux`-Datei löschen, wird dann jedoch Warnungen erhalten, daß bestimmte Verweise unbekannt sind.

`Label(s) may have changed. Rerun to get cross-references right.`
: Da die meisten Warnungen während des Formatierens ausgegeben werden und damit eventuell zu schnell wieder verschwinden, werden die wichtigsten Informationen am Ende wiederholt. Diese Meldung besagt, daß

die Verweise (`\cite`, `\ref`, `\pageref`, ...) falsch sein können, da sich die Werte bei der aktuellen Bearbeitung verändert haben. Ein nochmaliges Formatieren löst die Referenzen korrekt auf.

`Marginpar on page ... moved.`
Auf der angegebenen Seite wurde eine Randnotiz nach unten verschoben, da sie zu nah bei einer anderen lag. Die Notiz erscheint also nicht mehr in der Zeile, in der der `\marginpar`-Befehl steht. Eventuell muß das Problem behoben werden, indem man der Wert für `\marginparsep` verkleinert oder den Befehl in eine andere Zeile verschiebt.

`Oval too small.`
In einer `picture`-Umgebung wurde ein `\oval`-Befehl mit zu kleinen Abmessungen verwendet, so daß LaTeX dieses Oval nicht aus den zur Verfügung stehenden Zeichen erzeugen kann.

`Reference '...' on page ... undefined.`
In einem `\ref`- oder `\pageref`-Befehl wurde ein Schlüsselwort verwendet, daß (beim vorigen LaTeX-Lauf) nicht durch einen `\label`-Befehl definiert wurde. Da LaTeX die Schlüsselworte in einer Hilfsdatei (`.aux`) speichert, die am Anfang der Formatierung gelesen wird, und sich damit die Angaben über Referenzen u. ä. auf die jeweils letzte Bearbeitung beziehen, kann es sein, daß die Warnung beim nächsten Durchlauf nicht mehr auftritt.

`Some font shapes were not available, defaults substituted.`
Diese Schlußmeldung zeigt an, daß in dem Dokument Schriften angefordert wurden, die nicht zur Verfügung stehen. In der Regel beruht dies auf der falschen Kombination von Schriftbefehlen. Die Schriften wurden durch voreingestellte ersetzt. In einer früheren Warnung ist genauer angegeben, welche Schrift nicht verwendet werden konnte und durch welche sie ersetzt wurde.

`There were undefinded references.`
Auch diese Meldung erscheint am Ende der Bearbeitung. Sie besagt, daß einzelne Verweise nicht aufgelöst werden konnten, da die Markierungen beim vorigen LaTeX-Lauf noch nicht bekannt waren.

A.3.2 TeX-Warnungen

TeX-Warnungen enthalten im Gegensatz zu Fehlern und LaTeX-Warnungen keinerlei Indikator oder Einleitungstext. Sie werden direkt ausgegeben. Die häufigsten Meldungen sind:

`Overfull \hbox (...pt to wide) in paragraph at lines ... - ...`

TEX konnte an einer Stelle den Zeilenumbruch nicht richtig vornehmen. Die Zeile ragt um den in Klammern angegeben Betrag (1pt $\approx$ 0.35mm) über den rechten Rand hinaus. Die nächste Zeile der Warnung enthält nähere Informationen zu dem Problem. Es wird die gerade verwendete Schrift ausgegeben, gefolgt von einem kurzen Textausschnitt, innerhalb dessen der Zeilenumbruch stattgefunden hat. Die möglichen Trennungsstellen werden durch `--` gekennzeichnet.

Beheben kann man das Problem durch einen Zeilenumbruch oder durch Angeben der Trennung. Ragt der Text nur geringfügig über den Rand hinaus (1 – 2 Punkte), so ist dies für die meisten Dokumente noch akzeptabel. In diesem Fall sollte man das Ergebnis anhand des Probeausdrucks beurteilen.

`Overfull \vbox (... pt to high) ...`

Diese Meldung ist äußerst selten. TEX konnte auf einer Seite den Seitenumbruch nicht korrekt vornehmen. Der Text auf dieser Seite ragt über den unteren Seitenrand hinaus. Da TEX dazu neigt, eher zu wenig als zu viel Text auf einer Seite zu setzen, tritt diese Meldung eigentlich nur auf, wenn die Seite eine vertikale Box enthält, die höher als die Texthöhe (`\textheight`) ist. Auch hier können kleinere Abweichungen noch akzeptiert werden. Auf jeden Fall sollte man sich den Probeausdruck daraufhin anschauen.

`Underfull \hbox (badness ...) in paragraph at lines ... - ...`

Diese Meldung ist das Gegenteil zu `Overfull \hbox`. TEX hat in einer Zeile den Zeilenumbruch so vorgenommen, daß die Wortabstände zu groß für die recht hohen Anforderungen von TEX geworden sind. Der Wert in Klammern gibt Auskunft darüber, wie schlecht die Zeile gesetzt wurde. Der Zahlenwert kann zwischen 1000 und 10000 liegen[2]. In den meisten Fällen können Werte von weniger als 2000 toleriert werden.

`Underfull \vbox ...`

Analog zu `Underfull \hbox` bedeutet diese Meldung, daß beim Seitenumbruch die Absatzabstände zu sehr gedehnt wurden. Diese Meldung tritt besonders dann auf, wenn wie bei der Dokumentenklasse `book` die

[2] TEX vergibt intern für alle Aktionen gewichtete Strafpunkte. Beim Setzen des Textes werden verschiedene Möglichkeiten miteinander verglichen und die mit den wenigsten Punkten gewählt. Ein `badness`-Wert von 10000 zeigt dabei an, daß die tatsächliche Schrumpfung oder Dehnung in der Zeile mehr als das 4.5fache des normalen Wertes beträgt.

ersten und letzten Zeilen auf zwei gegenüberliegenden Seiten angeglichen werden.

A.4 Suche nach Fehlerursachen

Viele Ursachen von Fehlern, wie z. B. falsch geschriebene Befehlsnamen, vergessene oder zu viel gesetzte Klammern oder nicht beendete Umgebungen, sind in der Regel sehr leicht zu finden. Bei manchen anderen Fehlern kann es jedoch ausgesprochen mühsam werden, die eigentliche Ursache für einen Fehler zu finden. Als vorsorgende Maßnahme sollte man während der Erstellung eines Dokumentes zwischendurch immer mal wieder eine LaTeX-Bearbeitung durchführen. Dies gilt vor allem, wenn man kompliziertere Befehlsstrukturen verwendet. Tritt ein Fehler auf, den man nicht in kürzester Zeit beheben kann, so sollte man immer nur möglichst wenig im Quelltext ändern und dann eine erneute Formatierung starten. Auf jeden Fall sollte man sich genau merken, was man verändert hat, um die Auswirkungen besser beurteilen zu können.

Sicherheitshalber sollte man die Datei zunächst zweimal kopieren. Mit der ersten Kopie wird dann der Fehler gesucht. Ist das Dokument recht groß, so empfiehlt es sich, die Datei aufzuteilen. Dabei kann es jedoch passieren, daß beide Teile einzeln problemlos formatiert werden können. In diesem Fall sollte man die Trennung an einer anderen Stelle vornehmen. Alternativ kann man auch fehlerfreie Teile des Textes löschen, oder an entsprechender Stelle im Text den Befehl `\end{document}` einsetzen. Tritt der Fehler immer noch auf, so empfiehlt sich eine weitere Kürzung, jedoch sollte man vorher die Datei erneut kopieren. Tritt der Fehler dagegen in der gekürzten Fassung nicht mehr auf, so verwendet man die zweite Kopie und kürzt diese entsprechend weniger.

Mit diesem Verfahren kann man recht schnell die Umgebung oder den Befehl, der den Fehler verursacht, ausfindig machen. In der Regel beruht der Fehler auf komplizierteren Strukturen. Man sollte jetzt versuchen, einzelne Argumente oder Parameter zu verändern, um auf diese Weise den Fehler zu beheben.

Es kann vorkommen, daß beim Formatieren erneut ein Fehler gemeldet wird, obwohl die Eingabedatei korrigiert wurde. Dies liegt in der Regel daran, daß sich der Fehler in eine Hilfsdatei (`.aux`,`.toc`, ...) fortgesetzt hat. Welche dieser Dateien betroffen ist, hängt von der Art des Fehlers ab. Als Lösung sollte entweder die entsprechende Datei editiert oder gelöscht werden. Danach sollte der Fehler nicht mehr auftreten.

Insgesamt sollte versucht werden, einen fehlerfreien Quelltext zu schreiben, der auch möglichst wenig Warnungen produziert.

Anhang B

Mathematische Symbole

In den folgenden Tabellen sind alle Symbole angeführt, die standardmäßig im mathematischen Modus verwendet werden können. Die mit einem * versehenen Symbole sind in LaTeX 2ε nur mit dem Paket `latexsym` verwendbar. Bei vielen Installationen stehen mit den Paketen `amssymb` und `wasysym` weitere Zeichen zur Verfügung.

Tabelle B.1: Griechische Buchstaben

Großbuchstaben

Γ	`\Gamma`	Ξ	`\Xi`	Φ	`\Phi`
Δ	`\Delta`	Π	`\Pi`	Ψ	`\Psi`
Θ	`\Theta`	Σ	`\Sigma`	Ω	`\Omega`
Λ	`\Lambda`	Υ	`\Upsilon`		

Kleinbuchstaben

α	`\alpha`	ι	`\iota`	ϱ	`\varrho`
β	`\beta`	κ	`\kappa`	σ	`\sigma`
γ	`\gamma`	λ	`\lambda`	ς	`\varsigma`
δ	`\delta`	μ	`\mu`	τ	`\tau`
ϵ	`\epsilon`	ν	`\nu`	υ	`\upsilon`
ε	`\varepsilon`	ξ	`\xi`	ϕ	`\phi`
ζ	`\zeta`	o	`o`	φ	`\varphi`
η	`\eta`	π	`\pi`	χ	`\chi`
θ	`\theta`	ϖ	`\varpi`	ψ	`\psi`
ϑ	`\vartheta`	ρ	`\rho`	ω	`\omega`

Tabelle B.2: Mathematische Akzente

$\hat{x}$	`\hat{x}`	$\breve{x}$	`\breve{x}`	$\check{x}$	`\check{x}`
$\tilde{x}$	`\tilde{x}`	$\grave{x}$	`\grave{x}`	$\acute{x}$	`\acute{x}`
$\dot{x}$	`\dot{x}`	$\widehat{abc}$	`\widehat{abc}`	$\bar{x}$	`\bar{x}`
$\vec{x}$	`\vec{x}`	$\ddot{x}$	`\ddot{x}`	$\widetilde{xyz}$	`\widetilde{xyz}`

Tabelle B.3: Klammersymbole

$[$	`[`	$[$	`\lbrack`	$($	`(`
$]$	`]`	$]$	`\rbrack`	$)$	`)`
$\{$	`\{`	$\{$	`\lbrace`	$\lceil$	`\lceil`
$\}$	`\}`	$\}$	`\rbrace`	$\rceil$	`\rceil`
$\langle$	`\langle`	$\rangle$	`\rangle`	$\lfloor$	`\lfloor`

Tabelle B.4: Nicht-mathematische Symbole

†	`\dag`	§	`\S`	©	`\copyright`
‡	`\ddag`	¶	`\P`	£	`\pounds`
$	`\$`	%	`\%`	{	`\{`
}	`\}`	&	`\&`	#	`\#`

Tabelle B.5: Symbole in zwei Größen

$\sum$	`\sum`	$\bigcap$	`\bigcap`	$\bigodot$	`\bigodot`
$\prod$	`\prod`	$\bigcup$	`\bigcup`	$\bigotimes$	`\bigotimes`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`	$\bigoplus$	`\bigoplus`
$\int$	`\int`	$\bigvee$	`\bigvee`	$\biguplus$	`\biguplus`
$\oint$	`\oint`	$\bigwedge$	`\bigwedge`		

Tabelle B.6: Binäre Operationssymbole

$+$	`+`	$-$	`-`	$\pm$	`\pm`
$\cap$	`\cap`	$\vee$	`\vee`	$\mp$	`\mp`
$\cup$	`\cup`	$\wedge$	`\wedge`	$\setminus$	`\setminus`
$\uplus$	`\uplus`	$\oplus$	`\oplus`	$\cdot$	`\cdot`
$\sqcap$	`\sqcap`	$\ominus$	`\ominus`	$\times$	`\times`
$\sqcup$	`\sqcup`	$\otimes$	`\otimes`	$\ast$	`\ast`
$\triangleleft$	`\triangleleft`	$\oslash$	`\oslash`	$\star$	`\star`
$\triangleright$	`\triangleright`	$\odot$	`\odot`	$\diamond$	`\diamond`
$\wr$	`\wr`	$\dagger$	`\dagger`	$\circ$	`\circ`
$\bigcirc$	`\bigcirc`	$\ddagger$	`\ddagger`	$\bullet$	`\bullet`
$\bigtriangleup$	`\bigtriangleup`	$\amalg$	`\amalg`	$\div$	`\div`
$\bigtriangledown$	`\bigtriangledown`				

Tabelle B.7: Mathematische Vergleichssymbole

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq`	$\geq$	`\geq`	$\equiv$	`\equiv`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\ll$	`\ll`	$\gg$	`\gg`	$\asymp$	`\asymp`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni`	$\Join$	`\Join`*
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\smile$	`\smile`	$\mid$	`\mid`	$\doteq$	`\doteq`
$\frown$	`\frown`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\sqsubset$	`sqsubset`*	$\sqsupset$	`sqsupset`*	$\propto$	`\propto`
$\lhd$	`\lhd`*	$\unlhd$	`\unlhd`*	$\rhd$	`\rhd`*
				$\unrhd$	`\unrhd`*

Alle Vergleichssymbole können durch Voranstellen eines `\not` negiert werden: `\not\approx` ergibt $\not\approx$.

Tabelle B.8: Sonstige mathematische Symbole

$\aleph$	`\aleph`	$\prime$	`\prime`	$\forall$	`\forall`
$\hbar$	`\hbar`	$\emptyset$	`\emptyset`	$\exists$	`\exists`
$\imath$	`\imath`	∇	`\nabla`	$\neg$	`\neg`
$\jmath$	`\jmath`	$\surd$	`\surd`	$\flat$	`\flat`
ℓ	`\ell`	$\top$	`\top`	$\natural$	`\natural`
$\wp$	`\wp`	$\bot$	`\bot`	$\sharp$	`\sharp`
$\Re$	`\Re`	$\|$	`|`	$\clubsuit$	`\clubsuit`
$\Im$	`\Im`	$\angle$	`\angle`	$\diamondsuit$	`\diamondsuit`
∂	`\partial`	$\triangle$	`\triangle`	$\heartsuit$	`\heartsuit`
∞	`\infty`	$\backslash$	`\backslash`	$\spadesuit$	`\spadesuit`
$\mho$	`\mho`*	$\Box$	`\Box`*	$\Diamond$	`\Diamond`*

Tabelle B.9: Pfeil- und Zeigersymbole

$\leftarrow$	`\leftarrow`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\rightarrow$	`\rightarrow`	$\longrightarrow$	`\longrightarrow`	$\downarrow$	`\downarrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Downarrow$	`\Downarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleft...`	$\updownarrow$	`\updownarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleft...`	$\Updownarrow$	`\Updownarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`	$\nearrow$	`\nearrow`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`	$\searrow$	`\searrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`	$\nwarrow$	`\nwarrow`
$\rightleftharpoons$	`\rightleftharpoons`	$\leadsto$	`\leadsto`*		

Anhang C

Liste der wichtigsten Befehle

Dieser Abschnitt enthält eine alphabetisch sortierte Liste der wichtigsten Befehle von LaTeX sowie den im Buch beschriebenen Zusatzpaketen. Zu den Befehlen wird jeweils die Syntax erläutert und angegeben, auf welcher Seite der Befehl bzw. das Paket ausführlicher beschrieben ist. Fehlt diese Angabe, so wurde im Text nicht näher auf den Befehl eingegangen. Zusätzlich treten bei einigen Befehlen folgende Kennungen in eckigen Klammern auf:

[M] Der Befehl ist nur im mathematischen Modus erlaubt.
[P] Der Befehl ist nur in der Präambel erlaubt.
[Z] Der Befehl ist zerbrechlich.

Befehlsindex

A

`\AA` erzeugt Å — S. 29

`\aa` erzeugt å — S. 29

`\abovedisplayskip` [M]
Definiert den elastischen, vertikalen Abstand einer *langen* abgesetzten Formel zum vorangehenden Text.

`\abovedisplayshortskip` [M]
Definiert den elastischen, vertikalen Abstand einer *kurzen* abgesetzten Formel zum vorangehenden Text.

`\acute{`*x*`}` [M] — S. 130
Erzeugt ein Akut-Akzent über x: `\acute{x}` $= \acute{x}$

`\addcontentsline{`*Datei*`}{`*Format*`}{`*Eintrag*`}` [Z] — S. 43, 44
Manueller Eintrag in eine Verzeichnisdatei (`.toc`, `.lot`, `lof`). Der Text von *Eintrag* erscheint dabei in der Form *Format*.

`\address{`*Absender*`}` S. 75

Nur in der Klasse `letter` erlaubt. ***Absender*** enthält Namen und Adresse des Absenders. Zeilen werden durch `\\` getrennt.

`\addtocontens{`*Datei*`}{`*Eintrag*`}` [Z]

Erzeugt einen zusätzlichen Eintrag in der angegebenen Verzeichnisdatei (`.toc`, `.lot`, `lof`). Dies kann zum Beispiel einen Seitenumbruch sein. Zerbrechliche Befehle müssen eventuell durch Voranstellen von `\protect` geschützt werden.

`\addtocounter{`*Zähler*`}{`*Betrag*`}` S. 49

Addiert ***Betrag*** zum aktuellen Wert von ***Zähler***.

`\addtolength{`*Längenbefehl*`}{`*Maßangabe*`}` S. 21

Addiert ***Maßangabe*** zum angegebenen ***Längenbefehl***. Die Maßangabe kann feste oder elastische Maßangaben enthalten. Bei elastischen Angaben werden die Werte für *Sollwert*, *Dehnwert* und *Schrumpfwert* jeweils getrennt addiert. Enthielt der Längenbefehl vorher einen festen Wert, so wird bei Addition eines elastischen Wertes der *Sollwert* addiert und die elastischen Anteile hinzugefügt.

Maßangabe kann auch einen anderen Längenbefehl eventuell mit vorangestelltem Vorzeichen und/oder Faktor enthalten.

`\AE` erzeugt Æ S. 29

`\ae` erzeugt æ S. 29

`\afterpage{`*Befehle*`}` S. 69

Bewirkt, daß die Befehle, die ihm als Argument übergeben werden, erst nach der aktuellen Seite ausgeführt werden. Bestandteil des `afterpage`-Paketes.

`\aleph` [M] erzeugt $\aleph$ S. 132

`\allinethicknes{`*Stärke*`}` ???

Bestandteil des `eepic`-Paketes. Der Befehl legt die Strichstärke aller Strichzeichenbefehle fest. Dies betrifft gerade Linien, Kurven, Kreise, Ellipsen, Bögen, Ovale und Splines.

`\Alph{`*Zähler*`}`

Gibt den aktuellen Wert von ***Zähler*** als Großbuchstabe aus.

`\alph{`*Zähler*`}`

Gibt den aktuellen Wert von ***Zähler*** als Kleinbuchstabe aus.

`\alpha` [M] erzeugt α S. 129

`\amalg` [M] erzeugt $\amalg$ S. 130

`\and` S. 41

Trennt die einzelnen Autorennamen auf einer mit `\maketitle` gesetzten Titelseite.

`\angle` [M] erzeugt $\angle$ S. 132

`\approx` [M] erzeugt $\approx$ S. 131

`\arabic{`*`Zähler`*`}`
Gibt den aktuellen Wert von *`Zähler`* als arabische Zahl aus.

`\arc{`*`Durchmesser`*`}{`*`Anfangswinkel`*`}{`*`Endwinkel`*`}` ???
Zeichenbefehl des `eepic`-Paketes. Der Befehl zeichnet einen Kreisbogen mit dem angegebenen *`Durchmesser`*. Sowohl *`Anfangswinkel`*, als auch *`Endwinkel`* werden in Radianten angegeben. Die Bögen werden im Uhrzeigersinn gezeichnet.

`\arc[`*`NSymbol`*`](`*`x`*`,`*`y`*`){`*`Winkel`*`}` ???
Befehl des `curves`-Paketes. Der Befehl zeichnet einen Kreisbogen, der an der aktuellen Position des `\put`-Befehls zentriert ist und bei `(`*`x`*`,`*`y`*`)` beginnt. Der Bogen wird gegen den Uhrzeigersinn gezeichnet. *`Winkel`* wird in Grad angegeben. Das optionale Argument *`NSymbol`* beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

`\arccos` [M] S. 107
Erzeugt den Funktionsnamen „arccos“ in Formeln.

`\arcsin` [M] S. 107
Erzeugt den Funktionsnamen „arcsin“ in Formeln.

`\arctan` [M] S. 107
Erzeugt den Funktionsnamen „arctan“ in Formeln.

`\arg` [M] S. 107
Erzeugt den Funktionsnamen „arg“ in Formeln.

`\arraycolsep` S. 86
Enthält den halben Spaltenabstand in einer `array`-Umgebung. Die Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\arrayrulewidth` S. 86
Enthält die Liniendicke vertikaler und horizontaler Linien in `array`- und `tabular`-Umgebung. Die Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\arraystrech` S. 86
Faktor zur Veränderung des Zeilenabstandes in einer Tabelle. Wertzuweisung durch `\renewcommand{arraystretch}{`*`Faktor`*`}`. (Standard = 1.0)

`\ast` [M] erzeugt $\ast$ S. 130

`\asymp` [M] erzeugt $\asymp$ S. 131

`\atop` [M] S. 109
TEX-Befehl zum Übereinandersetzen von Daten. Syntax: `{`*`oben`*` \atop `*`unten`*`}`. Beispiel: `${a*b \atop a-b}$` erzeugt ${a*b \atop a-b}$

`\author{`*`Autor`*`}` S. 41

Erzeugt die Angabe eines Autorennamens auf einer Titelseite. Die Titelseite wird mit `\maketitle` gesetzt. *`Autor`* kann aus mehreren Namen bestehen, die durch ein `\and` getrennt werden.

B

`\b{`*x*`}` S. 29

Erzeugt einen Unterstreichungsakzent unter *x*: `\b{a}` erzeugt a̱

`\backslash` [M] erzeugt $\backslash$ S. 132

`\bar{`*x*`}` [M] S. 130

erzeugt einen Makron-Akzent über der Variable *x*: `$\bar{b}$` erzeugt $\bar{b}$

`\baselineskip` S. 78

Zeilenabstand innerhalb eines Absatzes. Jeder Zeichensatz hat einen eigenen Zeilenabstand. Wertzuweisung oder -änderung mit `\setlength` oder `\addtolength`.

`\baselinestretch`

Faktor, mit dem `\baselineskip` intern bei der Erzeugung des Zeilenabstandes multipliziert wird (Standard = 1.0). Wertänderung erfolgt mit `\renewcommand{\baselinestretch}{`*`Faktor`*`}` Die Wertänderung wird erst beim nächsten Schriftgrößenbefehl wirksam.

`\begin{`*`Umgebung`*`}` [Z] S. 52

Beginn einer Neuen Umgebung mit dem Namen *`Umgebung`*. Der Befehl erfordert zwingend die Beendigung der Umgebung mit `\end{`*`Umgebung`*`}`.

`\begin{abstract}` [Z]

Beginn der Umgebung `abstract` zur Erzeugung einer Zusammenfassung. In der Dokumentenklasse `article` wird die Zusammenfassung in der Schriftgröße `\small` und der Umgebung `quotation` gesetzt. In den Klassen `report` und `book` erscheint die Zusammenfassung auf einer eigenen Seite. In allen Fällen wird der Text mit einer zentrierten Überschrift **Abstract** bzw. **Zusammenfassung** eingeleitet. Die Überschrift kann mit `\renewcommand{\abstractname}{`*`Neuer Name`*`}` geändert werden.

`\begin{appendix}` [Z]

Beginn der Umgebung `appendix` zur Erzeugung eines Anhangs. Der Befehl bewirkt die Zurücksetzung des obersten Gliederungszählers und dessen Numerierung mit Großbuchstaben.

`\begin{array}[`*`Pos`*`]{`*`Spalten`*`}` [M] [Z] S. 109

Beginn der Umgebung `array` zur Erzeugung mathematischer Felder und Matrizen. Die Deklaration *`Spalten`* bestimmt das Aussehen der einzelnen Spalten. Erlaubte Angaben sind `l` für linksbündige, `c` für zentrierte und `r` für rechtsbündige Spalten. Zusätzlich können auch Striche `|` angegeben werden.

Der optionale Parameter *Pos* legt die Ausrichtung der Umgebung zur laufenden Zeile fest. Ein `t` richtet die oberste Zeile, ein `b` die unterste Zeile zur umgebenden Zeile aus. Ohne Angabe wird die Umgebung vertikal zentriert.

Die Zeilen werden durch `\\`, die Spalten durch `&` getrennt.

`\begin{bundle}{`*Oberknoten*`}` ???

Bestandteil des `ecltre`-Paketes. Beginnt eine neue Umgebung zum Zeichnen eines Baumdiagramms. Das Argument *Oberknoten* gibt den Text an, der über dem obersten Knoten erscheint.

`\begin{center}` [Z] S. 22

Beginn der Umgebung `center`. Der Text der Umgebung wird horizontal zentriert. Zeilenumbrüche erfolgen ohne Silbentrennung automatisch bei einem Leerzeichen oder werden mit `\\` erzwungen.
Siehe auch `\centering`.

`\begin{description}` [Z] S. 53

Beginn der Umgebung `description` zur Erzeugung einer markierten, eingerückten Liste. Die Marke wird den einzelnen `\item`-Befehlen als Option übergeben.

`\begin{displaymath}` [Z] S. 103

Umschaltbefehl zum Wechseln in den mathematischen Modus. Gleichwertig mit `\[`.

`\begin{document}` [Z] S. 8

Befehl zum Beenden der Präambel und zum Beginn des eigentlichen Textteiles eines jeden Dokumentes. Die Umgebung darf in jedem Dokument nur einmal auftreten und bildet die äußerste Umgebung. Der Befehl ist für jedes LaTeX-Dokument zwingend notwendig, ebenso wie sein Gegenstück `\end{document}`, das den gesamten Text beendet; eventuell folgender Text bleibt unberücksichtigt.

`\begin{enumerate}` [Z] S. 50

Beginn der `enumerate` Umgebung zur Erzeugung einer markierten, eingerückten Aufzählung. Die Markierung ist von der Schachtelungstiefe abhängig. Sie besteht in der ersten Ebene aus einer Nummer, die mit jedem `\item`-Befehl erzeugt und um eins erhöht wird.

`\begin{eqnarray}` [Z] S. 110

Befehl zum Umschalten in den mathematischen Modus zur Erzeugung einer abgesetzten Formel in Form einer dreispaltigen Tabelle (Deklaration: `{rcl}`). Die einzelnen Zeilen werden durch `\\`, die Spalten durch `&` getrennt. Jede Zeile erhält eine fortlaufende Nummer, deren Erzeugung mit dem Befehl `\nonumber` unterdrückt werden kann.

`\begin{eqnarray*}` [Z] S. 110
Entspricht dem Befehl `\begin{eqnarray}`. Allerdings werden die Zeilen nicht numeriert.

`\begin{equation}` [Z] S. 104
Umschaltbefehl zum Wechsel in den mathematischen Modus zur Erzeugung einer abgesetzten Formel mit automatischer Numerierung.

`\begin{figure}[`*`Pos`*`]` [Z] S. 67
Beginn der gleitenden `figure`-Umgebung zur Erzeugung einer Abbildung. Der optionale Parameter *`Pos`* legt die gewünschten Gleitmöglichkeiten fest. Erlaubt ist jede Kombination von `t`, `b`, `h` und `p`. Voreinstellung ist `tbp`.

`\begin{figure*}[`*`Pos`*`]` [Z] S. 67
Entspricht `\begin{figure}`, allerdings reicht die Abbildung bei mehrspaltigem Druck über alle Spalten des Textes.

`\begin{flushleft}` [Z] S. 23
Beginn der Umgebung `flushleft`. Der Text wird linksbündig gesetzt. Zeilenumbrüche erfolgen ohne Silbentrennung automatisch oder können mit `\\` erzwungen werden.
Siehe auch `\raggedright`.

`\begin{flushright}` [Z] S. 23
Beginn der Umgebung `flushright`. Der Text wird rechtsbündig gesetzt. Zeilenumbrüche erfolgen ohne Silbentrennung automatisch oder können mit `\\` erzwungen werden.
Siehe auch `\raggedleft`.

`\begin{fussypar}` [Z]
Beginn der Umgebung `fussypar`, innerhalb derer die Absatzformatierung standardmäßig, d.h. mit begrenzter Elastizität der Wortabstände, erfolgt. Gegensatz zu `sloppypar`.

`\begin{itemize}` [Z] S. 53
Beginn der Umgebung `itemize` zur Erzeugung einer markierten, eingerückten Liste. Die Markierung ist von der Schachtelungstiefe abhängig und besteht in der obersten Ebene aus einen •.

`\begin{letter}{`*`Empfänger`*`}` [Z] S. 75
Beginn eines Briefes in der Dokumentenklasse `letter`. Der zwingende Parameter *`Empfänger`* enthält Name und Anschrift des Empfängers. Zeilen werden im Empfängerfeld mit `\\` getrennt.

`\begin{list}{`*`Marke`*`}{`*`Definition`*`}` [Z]
Beginn einer allgemeinen Listenumgebung. Der Parameter *`Marke`* definiert die gewünschte Markierung, *`Definition`* enthält die gewünschten Längenerklärungen für das Aussehen der Liste.

`\begin{math}` [Z]
Befehl zum Umschalten vom Text- in den mathematischen Modus. Gleichwertig mit `\(` und dem ersten Auftreten des `$`-Zeichens.

`\begin{minipage}[Pos]{Breite}` [Z] S. 60
Beginn der Umgebung `minipage` zur Erzeugung einer Teilseite mit der angegebenen Breite *`Breite`*. Der optionale Parameter *`Pos`* definiert die Ausrichtung der Umgebung zur laufenden Zeile. Ein `t` richtet die oberste Zeile, ein `b` die unterste Zeile zur umgebenden Zeile aus. Ohne Angabe wird die Umgebung vertikal zentriert.

`\begin{picture}(X,Y)(dx,dy)` [Z]
Umgebung zur Erzeugung einer Graphik. Die Parameter *`X`* und *`X`* legen die Breite und Höhe des Bildes fest. Mit der optionalen Angabe von *`dx`* und *`dy`* wird der Koordinatenursprung der Graphik zusätzlich horizontal und vertikal verschoben.

`\begin{samepage}` [Z]
Erlaubt einen Seitenumbruch nur an eingeschränkten Stellen, wie z.B. zwischen Absätzen, oder dort, wo er mit `\newpage` oder `\pagebreak` erzwungen wird.
Siehe auch `\samepage`

`\begin{quotation}` [Z] S. 24
Beginn der `quotation`-Umgebung zur Erzeugung von beidseitig eingerücktem Text. Absätze beginnen mit einer zusätzlich eingerückten Zeile. Die Einrücktiefe der Umgebung und der ersten Zeilen eines Absatzes nehmen mit zunehmender Verschachtelung ab.

`\begin{quote}` [Z] S. 24
Beginn der `quote`-Umgebung zur Erzeugung von beidseitig eingerücktem Text. Absätze werden durch einen zusätzlichen vertikalen Zwischenraum getrennt. Die Einrücktiefe der Umgebung nimmt mit zunehmender Verschachtelung ab.

`\begin{sloppypar}` [Z] S. 18
Gegenstück zur `fussypar`-Umgebung. Innerhalb der `sloppypar`-Umgebung sind größere Wortabstände erlaubt. Eignet sich besonders zum Setzen schmaler Zeilen.

`\begin{tabbing}` [Z] S. 56
Beginn der `tabbing`-Umgebung zum Verwenden von Tabulatorstops. Mit `\=` werden beliebige Tabulatorstops festgelegt, zu denen in den einzelnen Zeilen mit `\>` vor- und `\<` zurückgesprungen werden kann. Mit `\+` und `\-` kann der linke Rand um jeweils einen Tabulatorstop nach rechts bzw. links verschoben werden.

`\begin{table}[Pos]` [Z] S. 68
Gleitende Umgebung zur Aufnahme von Tabellen. Der optionale Para-

meter *Pos* legt die gewünschten Gleitmöglichkeiten der Umgebung fest. Erlaubt ist jede Kombination von `t`, `b`, `h` und `p`. Voreinstellung ist `tbp`.

`\begin{table*}[`*Pos*`]` [Z] S. 68

Entspricht der `table`-Umgebung, allerdings reicht die Abbildung über mehrspaltigem Druck über alle Spalten des Textes.

`\begin{tabular}[`*Pos*`]{`*Spalten*`}` [Z] S. 83

Beginn der `tabular`-Umgebung zur Erzeugung von Tabellen. Die Deklaration *Spalten* bestimmt das Aussehen der einzelnen Spalten. Erlaubte Angaben sind `l` für linksbündige, `c` für zentrierte und `r` für rechtsbündige Spalten. Zusätzlich kann neben Strichen `|` über die gesamte Tabellenhöhe auch noch die Option `\p{`*Maß*`}` für eine Spalte mit der Breite *Maß* angegeben werden. Mit `\@{Text}` kann ein zusätzlicher Eintrag zwischen zwei Spalten erfolgen.

Der optionale Parameter *Pos* legt die Ausrichtung der Umgebung zur laufenden Zeile fest. Ein `t` richtet die oberste Zeile, ein `b` die unterste Zeile zur umgebenden Zeile aus. Ohne Angabe wird die Umgebung vertikal zentriert.

Die Zeilen der Tabelle werden durch `\\`, die Spalten durch `&` getrennt.

`\begin{tabular*}{`*Breite*`}[`*Pos*`]{`*Spalten*`}` [Z] S. 83

Entspricht der `tabular`-Umgebung. Allerdings muß mit *Breite* die Gesamtbreite der Tabelle angegeben werden. Dabei wird entweder die letzte Spalte soweit verbreitert, bis die Breite erreicht ist, oder man erreicht mit der Angabe von `@{\extracolsep\fill}` am Anfang der Spaltendeklaration (`{`*Spalten*`}`) einen variablen Spaltenzwischenraum.

`\begin{thebibliography}{`*Mustermarke*`}` [Z] S. 44

Beginnt die Umgebung zur Erzeugung eines Literaturverzeichnisses. Der Parameter *Mustermarke* bestimmt die Einrücktiefe der Einträge. Jeder Eintrag beginnt mit einem `\bibitem`-Befehl.

`\begin{theindex}` [Z]

Umgebung zur Erzeugung eines Indexes. Der Index wird zweispaltig gedruckt. Einträge erfolgen mit `\item`, `\subitem` und `\subsubitem`. Vertikalen Zwischenraum erreicht man mit `\indexspace`. Steht das Programm `makeindex` zur Verfügung, so kann die Erstellung automatisiert werden.

`\begin{`*Theorem-Name*`}[`*Zusatz*`]` [Z] S. 56

Beginnt eine anwendereigene Theorem-Umgebung. *Theorem-Name* muß vorher mit einem `\newtheorem`-Befehl definiert werden. Der optionale Parameter *Zusatz* erscheint in runden Klammern hinter der fortlaufenden Nummer des Theorembegriffs.
Siehe auch `\newtheorem`

`\begin{titlepage}` [Z] S. 41
Umgebung zur freien Gestaltung einer Titelseite für ein Dokument. Die Seite erhält keine Seitennummer.

`\begin{trivlist}` [Z]
Umgebung zur Erzeugung einer trivialen Liste. Die Listeneinträge enthalten keine Markierung. Die Werte für `\leftmargin`, `\labelwidth` und `\itemsep` betragen jeweils 0pt und `\listparindent` = `\parindent`, `\parsep` = `\parskip`.

`\begin{verbatim}` [Z] S. 57
Beginnt eine Umgebung zum Setzen von Text in der Form der Originaleingabe in Schreibmaschinenschrift. Alle Zeichen werden so ausgegeben, wie sie in der Datei stehen, d.h. keinerlei Befehle werden bearbeitet.
Siehe auch `\verb`.

`\begin{verbatim*}` [Z] S. 57
Ähnlich der `verbatim`-Umgebung. Allerdings werden Leerzeichen als ␣ ausgegeben.
Siehe auch `\verb*`.

`\begin{verse}` [Z] S. 24
Umgebung zum beidseitig eingerückten Setzen von Gedichten, Versen, Reimen u.ä. Die Strophen werden durch Leerzeilen getrennt. Die einzelnen Zeilen werden durch `\\` getrennt. Ist eine Zeile länger als die aktuelle Zeilenlänge (`\hsize`), so wird diese ohne Silbentrennung umgebrochen und stärker eingerückt.

`\belowdisplayskip` [M]
Definiert den elastischen vertikalen Abstand einer *langen* abgesetzten Formel zum nachfolgenden Text.

`\belowdisplayshortskip` [M]
Definiert den elastischen vertikalen Abstand einer *kurzen* abgesetzten Formel zum nachfolgenden Text.

`\beta` [M] erzeugt β S. 129

`\bfseries` S. 30
Schaltet auch **fettgedruckte** Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\bibitem[`*`Marke`*`]{`*`Schlüssel`*`}` *`Text`* S. 44
Erzeugt einen Eintrag in einem Literaturverzeichnis (`thebibliography`-Umgebung). Der optionale Parameter *`Marke`* wird dabei vor dem Eintrag in eckigen Klammern ausgegeben. Wird der Parameter weggelassen, so erscheint eine fortlaufende Nummer. Auf das mit *`Schlüssel`* definierte Bezugswort kann im Text mit `\cite{`*`Schlüssel`*`}` verwiesen werden.

`\bibliography{`*Datei*`}`
: Befehl im Zusammenhang mit dem Programm BibTeX. Der Befehl definiert die `.bib`-Datei (ohne Angabe der Dateiendung), aus der das Literaturverzeichnis erstellt werden soll.

`\bibliographystyle{`*Stil*`}`
: Legt den Formatierungsstil fest, nach dem das von BibTeX erzeugte Literaturverzeichnis gesetzt werden soll.

`\big`*Symbol* [M]
: Erzeugt ein größeres Klammersymbol als normal, jedoch kleiner als mit `\Big`. Beispiel: `\big[`

`\Big`*Symbol* [M]
: Erzeugt ein größeres Klammersymbol als mit `\big`, jedoch kleiner als mit `\bigg`. Beispiel: `\Big)`

`\bigcap` [M] erzeugt $\bigcap$ S. 132

`\bigcirc` [M] erzeugt $\bigcirc$ S. 132

`\bigcircle[`*NSymbol*`]{`*Durckmesser*`}` ???
: Befehl des `curves`-Paketes. Zeichnet einen Kreis mit dem angegebenen Durchmesser in Einheiten von `\unitlength`. *NSymbol* beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

`\bigcup` [M] erzeugt $\bigcup$ S. 132

`\bigg`*Symbol* [M]
: Größeres Klammersymbol als mit `\Big`, jedoch kleiner als `\Bigg`. Beispiel: `\bigg(`

`\Bigg`*Symbol* [M]
: Größtes Klammersymbol. Beispiel: `\Bigg)`

`\biggl`*Symbol* [M]
: Entspricht `\bigg` mit gleichzeitig funktionell öffnender Klammer.

`\Biggl`*Symbol* [M]
: Entspricht `\Bigg` mit gleichzeitig funktionell öffnender Klammer.

`\biggm`*Symbol* [M]
: Entspricht `\bigg` mit einem größeren horizontalen Abstand.

`\Biggm`*Symbol* [M]
: Entspricht `\Bigg` mit einem größeren horizontalen Abstand.

`\biggr`*Symbol* [M]
: Entspricht `\bigg` mit gleichzeitig funktionell schließender Klammer.

`\Biggr`*Symbol* [M]
: Entspricht `\Bigg` mit gleichzeitig funktionell schließender Klammer.

`\bigl`*Symbol* [M]
Entspricht `\big` mit gleichzeitig funktionell öffnender Klammer.

`\Bigl`*Symbol* [M]
Entspricht `\Big` mit gleichzeitig funktionell öffnender Klammer.

`\bigm`*Symbol* [M]
Entspricht `\big` mit einem größeren horizontalen Abstand.

`\Bigm`*Symbol* [M]
Entspricht `\Big` mit einem größeren horizontalen Abstand.

`\bigodot` [M] erzeugt $\bigodot$ S. 132

`\bigoplus` [M] erzeugt $\bigoplus$ S. 132

`\bigotimes` [M] erzeugt $\bigotimes$ S. 132

`\bigr`*Symbol* [M]
Entspricht `\big` mit gleichzeitig funktionell schließender Klammer.

`\Bigr`*Symbol* [M]
Entspricht `\Big` mit gleichzeitig funktionell schließender Klammer.

`\bigtriangledown` [M] erzeugt $\bigtriangledown$ S. 130

`\bigtriangleup` [M] erzeugt $\bigtriangleup$ S. 130

`\bigskip` [Z] S. 20
Erzeugt einen großen vertikalen Zwischenraum von der Größe `\bigskipamount`.

`\bigskipamount`
Standardwert für den durch `\bigskip` erzeugten vertikalen Zwischenraum. Wertzuweisung mit erfolgt `\setlength` oder `\addtolength`

`\bigsqcup` [M] erzeugt $\bigsqcup$ S. 132

`\biguplus` [M] erzeugt $\biguplus$ S. 132

`\bigvee` [M] erzeugt $\bigvee$ S. 132

`\bigwedge` [M] erzeugt $\bigwedge$ S. 132

`\bmod` [M] S. 106
Befehl zur Erzeugung des Funktionsnamens „mod“. Beispiel: `a \bmod b` erzeugt $a \bmod b$

`\boldmath` [Z]
Befehl zum Umschalten auf Fettdruck in mathematischen Formeln. Der Befehl muß im Textmodus stehen. Für eine Teilformel muß vorübergehend mit `\mbox{\boldmath$...$}` in den Textmodus zurückgeschaltet werden.

`\bot` [M] erzeugt $\bot$ S. 132

`\bottomfraction`
Maximaler Anteil einer Seite, der von Gleitobjekten am Fuß der Seite eingenommen werden darf. Voreinstellung ist 0.3. Eine Wertänderung erfolgt mit `\renewcommand`.

Die maximale Anzahl von Gleitobjekten, die am Fuß der Seite gesetzt werden dürfen, wird mit `bottomnumber` definiert. Voreinstellung ist 1. Der Wert wird mit `\setcounter` geändert.

`\bowtie` [M] erzeugt $\bowtie$ S. 131

`\Box` [M] erzeugt $\Box$ (wird vom Paket `latexsym` bereitgestellt) S. 132

`\branch{`*`Steilheit`*`}{`*`Text`*`}{`*`ID`*`}:`*`TocherA`*`,`*`TochterB`*`.` ???
Befehl des `trees`-Paket zum Zeichnen einer Verzweigung in einem binären Baum. Der Parameter *`Steilheit`* legt den Anstieg der Zweige fest. Er wird durch eine ganze Zahl zwischen 0 und 3 angegeben. *`Text`* wird über dem entsprechenden Punkt eingefügt. *`ID`* bezeichnet die gegenwärtige Verzweigung. *`TochterA`* und *`TochterB`* bezeichnen die beiden Tochterknoten. Doppelpunkt, Komma und Punkt sind obligatorisch.

`\branchlabels{`*`LabelA`*`}{`*`LabelB`*`}{`*`LabelC`*`}` ???
Befehl des `trees`-Paketes. Legt die Label fest, die für die Zweige des Baumes verwendet werden. Die Voreinstellung ist 0 für den ersten, 1 für den zweiten und 2 für den dritten Zweig.

`\breve{`*`x`*`}` [M] S. 130
Breve-Akzent über *`x`*. `$\breve{c}$` erzeugt $\breve{c}$

`\bullet` [M] erzeugt $\bullet$ S. 130

C

`\c{`*`x`*`}` S. 29
Erzeugt eine Cedille unter *`x`*: `\c{C}` erzeugt Ç

`\cap` [M] erzeugt $\cap$ S. 130

`\caption[{`*`Kurzform`*`]{`*`Überschrift`*`}` [Z] S. 44, 46
Erzeugt eine numerierte Über- oder Unterschrift innerhalb einer Gleitumgebung. Mit dem optionalen Parameter *`Kurzform`* kann eine verkürzte Form zum Eintrag in das Abbildungs- oder Tabellenverzeichnis verwendet werden.

`\cc{`*`Liste`*`}` S. 76
Befehl der Dokumentenklasse `letter`, zur Erzeugung einer Verteilerliste (*`Liste`*) am Ende des Briefes.

`\cdot` [M] erzeugt $\cdot$ S. 106

`\cdots` [M] erzeugt $\cdots$ S. 106

`\centering` S. 91
Setzt allen nachfolgenden Text bis zum Ende der aktuellen Umgebung horizontal zentriert. Zeilenumbrüche erfolgen ohne Silbentrennung automatisch oder werden mit `\\` erzwungen.
Siehe auch `\begin{center}`.

`\centerline{`*Text*`}` S. 23
Setzt den übergebenen *Text* in einer eigenen Zeile horizontal zentriert.

`\cfoot[`*MF-gerade*`]{`*MF-ungerade*`}` S. 16
Definiert den mittleren Teil der Fußzeile beim Seitenstil `fancy`. Der optionale Parameter *MF-gerade* kann genutzt werden, um bei zweiseitigem Druck die Ausgabe auf geraden Seiten zu ändern.
Siehe auch `\chead`, `\lhead`, `\rhead`, `\lfoot`, `\rfoot`

`\chapter[`*Kurzform*`]{`*Überschrift*`}` [Z] S. 40
Gliederungsbefehl, der ein neues Kapitel mit einer neuen Seite startet. Das Kapitel wird mit einer fortlaufenden Nummer versehen. Wenn der optionale Parameter *Kurzform* angegeben wird, wird dieser für den Eintrag ins Inhaltsverzeichnis und die Kopfzeile der Seiten verwendet.

`\chapter*{`*Überschrift*`}` [Z] S. 40
Gliederungsbefehl, der ein neues Kapitel mit einer neuen Seite startet. Allerdings wird keine Kapitelnummer ausgegeben und der interne Zähler wird nicht erhöht. Außerdem wird kein Eintrag ins Inhaltsverzeichnis vorgenommen.

`\chead[`*MK-gerade*`]{`*MK-ungerade*`}` S. 16
Definiert den mittleren Seitenkopf beim Seitenstil `fancy`. Der optionale Parameter *MK-gerade* kann genutzt werden, um bei zweiseitigem Druck die Ausgabe auf geraden Seiten zu ändern.
Siehe auch `\lhead`, `\rhead`, `\cfoot`, `\lfoot`, `\rfoot`

`\check{x}` [M] S. 108
Erzeugt einen Háček-Akzent über *x*. Beispiel: `\check{c}` erzeugt $\check{c}$

`\chi` [M] erzeugt χ S. 129

`\choose` [M] S. 109
Mathematischer TEX-Befehl zur Erzeugung eines Binominalkoeffizienten. Syntax: `{`*oben* `\choose` *unten*`}` erzeugt $\binom{\text{oben}}{\text{unten}}$

`\chunk[`*Kantentext* `{`*Knotentext*`}]` ???
Befehl des `ecltree`-Paketes. Legt einen neuen Knoten des Baumes fest. *Knotentext* ist der Label für diesen Knoten. Mit *Kantentext* kann optional ein Text angegeben werden, der neben der Kante erscheint.

`\circ` [M] erzeugt $\circ$ S. 130

`\circle{`*Durchmesser*`}` ???

Bildbefehl der `picture`-Umgebung, mit dem ein Kreis mit dem angegebenen *Durchmesser* erzeugt wird. Der Befehl wird einem `\put`- oder `\multiput`-Befehl als Argument übergeben. Die Koordinaten des Positionierungsbefehls beziehen sich auf den Kreismittelpunkt.

`\circle*{`*Durchmesser*`}` ???

Wie `\circle`, jedoch wird der Kreis mit Farbe gefüllt.

`\cite[`*Zusatz*`]{`*Schlüssel*`}` [Z] S. 45

Befehl zur Erzeugung eines Literaturverweises innerhalb eines Textes. Mit dem Parameter *Schlüssel* wird ein im Literaturverzeichnis definierter Eintrag referenziert. Mit dem optionalen Parameter *Zusatz* kann ein zusätzlicher Text angegeben werden, der direkt hinter der Markierung im Text erscheint.

`\cleardoublepage` [Z] S. 68

Beendet die laufende Seite und erzwingt die Ausgabe aller bis dahin nocht nicht bearbeiteten Gleitobjekte. Der Befehl beginnt den weiteren Text mit einer ungraden (rechten) Seite.

`\clearpage` S. 68

Beendet die laufende Seite und erzwingt die Ausgabe aller bis dahin nocht nicht bearbeiteten Gleitobjekte. Der folgende Text wird mit einer neuen Seite fortgesetzt.

`\cline{`*a-b*`}` S. 85

Befehl innerhalb einer `tabular`-Umgebung zur Erzeugung einer horizontalen Linie von Spalte *a* bis Spalte *b* (beide einschließlich).

`\closecurve[`*NSymbol*`](`x_1`,`y_1`,`x_2`,`y_2`,...,`x_n`,`y_n`)` ???

Befehl des `curves`-Paketes. Der Befehl erzeugt eine geschlossene Kurve. Es müssen dazu mindestens sechs Punkte angegeben werden. Das optionale Argument *NSymbol* beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

`\closing{`*Gruß*`}` S. 76

Befehl innerhalb einer `letter`-Umgebung zur Beendigung des eigentlichen Brieftextes. *Gruß* steht für die gewünschte Grußformel und darf aus mehreren Zeilen bestehen.

`\clubsuit` [M] erzeugt ♣ S. 132

`\color{`*Farbe*`}` ???

Befehl des `color`-Paketes. Druckt sämtlichen nachfolgenden Text bis zum Ende der aktuellen Umgebung in *Farbe*. *Farbe* muß dabei, wie auch in allen nachfolgenden Befehlen, ein vorher definierter Farbname sein.

`\color[`*Modell*`]{`*Definition*`}` ???

Befehl des `color`-Paketes. Analog zum obigen Befehl. Allerdings wird die Farbe direkt definiert.

`\colorbox{Farbe}{Text}` ???

Befehl des `color`-Paketes. Erzeugt eine rahmenlose Box mit der Hintergrundfarbe *Farbe*. *Text* wird in der aktuellen Vordergrundfarbe besetzt.

`\columnsep` S. 14

Definition des Spaltenabstandes bei mehrspaltigem Druck. Wertzuweisung mit `\setlength` oder `\addtolength`.

`\columnseprule` S. 14

Definition der Strichdicke für den vertikalen Trennstrich bei mehrspaltigem Druck. Wertzuweisung mit `\setlength` oder `\addtolength`.

`\cong` [M] erzeugt $\cong$ S. 131

`\contensline{Typ}{\numberline{Nummer} Titeltext}{Seite}`

Interner Befehl, der durch den Befehl `\tableofcontents` in die Inhaltsverzeichnisdatei (`.toc`) geschrieben wird. Bei Bedarf kann der Befehl von Hand verändert werden. *Typ* steht dabei für den Namen einer Gliederungsebene, *Nummer* ist die Gliederungsnummer und *Seite* die zugehörige Seite.

`\coprod` [M] erzeugt $\coprod$ S. 132

`\copyright` erzeugt © S. 131

`\cornersize{Faktor}` S. 61

Bestandteil des `fancybox`-Paketes. Legt den Radius der Ecken bei einer `\ovalbox` fest. Der Befehl multipliziert *Faktor* mit dem Minimum von Breite und Höhe der Box. Voreinstellung ist 0.5.

`\cos` [M] S. 107

Befehl zur Erzeugung des Funktionsnamens „cos" in Formeln.

`\cosh` [M] S. 107

Befehl zur Erzeugung des Funktionsnamens „cosh" in Formeln.

`\cot` [M] S. 107

Befehl zur Erzeugung des Funktionsnamens „cot" in Formeln.

`\coth` [M] S. 107

Befehl zur Erzeugung des Funktionsnamens „coth" in Formeln.

`\csc` [M] S. 107

Befehl zur Erzeugung des Funktionsnamens „csc" in Formeln.

`\cup` [M] erzeugt $\cup$ S. 130

`\curve[NSymbol]`$(x_1,y_1,x_2,y_2,\ldots,x_n,y_n)$???

Befehl des `curves`-Paketes. Der Befehl zeichnet eine Kurve durch die angegebenen Punkte. Zwei Punkte ergeben eine Gerade, drei eine Parabel. Das optionale Argument *NSymbol* beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

D

`\d{x}` erzeugt einen Unterpunkt-Akzent: Ọ S. 29

`\dag` erzeugt † S. 131

`\dagger` [M] erzeugt † S. 131

`\dashbox{Strichlänge}(Breite,Höhe)[Position{Inhalt}]` ???

Rahmenbox der `picture`-Umgebung mit der angegebenen *Breite* und *Höhe*, bei der der Rahmen gestrichelt wird. Die Länge der Striche muß mit *Strichlänge* festgelegt werden. Ohne Angabe von *Pos* erscheint der Inhalt horizontal und vertikal zentriert in der Box. Der optionale Parameter *Pos* läßt mit `l`, `r`, `t`, `b` oder einer Paarkomibation (z.B. `rt`) eine Änderung zu.

`\dashline[Dehn]{Länge}[Lücke]`$(x_1,y_1)(x_2,y_2)\ldots(x_n,y_n)$???

Zeichenbefehl des `epic`-Paketes. Verbindet die angegebenen Punkte mit einer gestrichelten Linie. Der optionale Parameter *Dehn* legt fest, wie stark die Striche gedehnt oder geschrumpft werden. Der Parameter muß zwischen -100 und $+\infty$ liegen. *Länge* legt die Länge der verwendeten Striche fest, *Lücke* definiert den Abstand zwischen den Punkten, die für einen Strich verwendet werden. Standardmäßig werden durchgezogene Striche verwendet.

`\dashv` [M] erzeugt ⊣ S. 131

`\date{Datum}` S. 41

Der Befehl `\maketitle` erzeugt standardmäßig eine Datumsangabe auf einer Titelseite. Mit diesem Befehl kann die Ausgabe verändert oder unterdrückt werden (`\datum{}`).

`\dblfigrule`

Befehl zum Erzeugen einer Trennlinie zwischen Gleitobjekten am Seitenanfang und dem nachfolgenden zweispaltigen Text. Die Linie wird direkt vor dem Abstand `\dbltextfloatsep` zwischen Text und Gleitobjekt eingefügt. Ebenso wie `\footnoterule` darf sie keinen vertikalen Raum einnehmen.

`\dblfloatpagefraction`

Anteil der Seitenfläche bei zweispaltigem Druck, der von zweispaltigen Gleitobjekten eingenommen werden darf. Voreinstellung ist 0.5. Eine Wertänderung erfolgt mit `\renewcommand`.

`\dblfloatsep`

Gummilänge, die den vertikalen Abstand zwischen zweispaltigen Gleitobjekten am Kopf oder Fuß einer Seite angibt. Der Wert ist voreingestellt aus `12pt plus 2pt minus 2pt` für Dokumente mit dem Schriftgrad 10pt oder 11pt und `14pt plus 2pt minus 4pt` für Dokumente mit Schriftgrad 12pt. Änderung erfolgt mit `\setlengt` oder `\addtolength`.

`\dbltextfloatsep`
Gummilänge, die der vertikalen Abstand zwischen einem zweispaltigen Gleitobjekt am Kopf oder Fuß einer Seite und dem Text angibt. Voreingestellt auf `20pt plus 2pt minus 4pt`. Wird mit `\setlength` oder `\addtolength` geändert.

`\dbltopfraction`
Der Anteil einer zweispaltigen Seite, der von zweispaltigen Gleitobjekten eingenommen werden darf. Voreingestellt ist 0.7. Änderung erfolgt mit `\renewcommand`.

`\ddag` erzeugt ‡ S. 131

`\ddagger` [M] erzeugt ‡ S. 131

`\ddot{`*x*`}` [M] S. 130
Erzeugt einen Doppelpunkt-Akzent im mathematischen Formeln:
`\ddot{o}` = $\ddot{o}$

`\ddots` [M] erzeugt $\ddots$ S. 106

`\deg` [M] S. 107
Befehl zur Erzeugung des Funktionsnamens „deg“ in mathematischen Formeln.

`\Delta` [M] erzeugt Δ S. 129

`\delta` [M] erzeugt δ S. 129

`\det` [M] S. 107
Befehl zur Erzeugung des Funktionsnames „det“ in mathematischen Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine Grenzangabe.

`\Diamond` [M] erzeugt $\Diamond$ S. 132

`\diamond` [M] erzeugt $\diamond$ S. 132

`\diamondsuit` [M] erzeugt $\diamondsuit$ S. 132

`\dim` [M] S. 107
Befehl zur Erzeugung des Funktionsnamens „dim“ in mathematischen Formlen.

`\discretionary{`*Vor*`}{`*Nach*`}{`*Ohne*`}` S. 26
Definiert die Trennart bei Worten, deren Schreibweise sich beim Trennen ändert. *Vor* und *Nach* geben die Schreibweise beim Trennen vor bzw. nach der Trennung an, *Ohne* legt fest, wie das Wort ungetrennt geschrieben wird. Beispiel `Dru\discretionary{k-}{k}{ck}er`

`\div` [M] erzeugt $\div$

`\documentclass[`*Optionen*`]{`*Klasse*`}` [P] S. 9

Normalerweise die erste Zeile eines LaTeX 2_ε-Dokumentes. Mit *Klasse* muß genau eine der definierten Dokumentenklassen gewählt werden. *Optionen* enthält Optionen für die gewählte Klasse oder spätere geladene Pakete.

`\dot{`*x*`}` [M] S. 130

Erzeugt einen Punkt-Akzent: `\dot{x}` $= \dot{x}$

`\doteq` [M] erzeugt $\doteq$ S. 131

`\dotfill` S. 20

Füllt einen gegebenen Raum, z.B. eine Zeile, mit Punkten auf: ....

`\dots` erzeugt ... S. 28

`\dottedline[`*Zeichen*`]{`*Lücke*`}(`x_1`,`y_1`)(`x_2`,`y_2`)...(`x_n`,`y_n`)` ???

Zeichenbefehl des `epic`-Paketes. Erzeugt eine gepunktete Linie zwischen den angegebenen Punkten. Mit dem optionalen Parameter *Zeichen* kann ein bestimmtes Zeichen für die Punkte festgelegt werden. Der Leerraum zwischen den Punkten wird mit *Lücke* eingestellt.

`\doublerulesep` S. 86

Definiert den Abstand von Doppellinien innerhalb einer `array`- oder `tabular`-Umgebung. Änderung erfolgt außerhalb der Umgebung mit `\setlength`.

`\doublebox{`*Inhalt*`}` S. 61

Erzeugt eine $\boxed{\text{Doppelbox}}$ um *Inhalt*. Bestandteil des `fancybox`-Paketes.

`\Downarrow` [M] erzeugt $\Downarrow$ S. 132

`\downarrow` [M] erzeugt $\downarrow$ S. 132

`\drawline[`*Dehn*`](`x_1`,`y_1`)(`x_2`,`y_2`)...(`x_n`,`y_n`)` ???

Zeichenbefehl des `epic`-Paketes. Verbindet die angegebenen Punkte mit einer durchgezogenen Linie. Mit dem optionalen Parameter *Dehn* kann die Anzahl der Segmente, aus denen die Linie zusammengesetzt wird, vergrößert werden. Dadurch entsteht eine weniger gezackte Linie.

`\drawwith{`*Zeichenbefehl*`}` ???

Befehl des `ecltree`-Paketes. Dient dazu, das Aussehen der Kanten zu verändern.

E

`\ell` [M] erzeugt ℓ S. 132

`\ellipse{`*x-Durchmesser*`}{`*y-Durchmesser*`}` ???

Zeichenbefehl des `eepic`-Paketes. Zeichnet eine Ellipse mit den angegebenen *x-Durchmesser* und *y-Durchmesser*.

`\ellipse*{`*x-Durchmesser*`}{`*y-Durchmesser*`}` ???

Zeichenbefehl des `eepic`-Paketes. Zeichnet eine mit Farbe gefüllte Ellipse mit den angegebenen *x-Durchmesser* und *y-Durchmesser*.

`\emph` S. 53, 30

Umschaltbefehl auf die *hervorgehobene* Schriftart.

`\emptyset` [M] erzeugt $\emptyset$ S. 132

`\encl{`*Anlagen*`}` S. 76

Befehl der Dokumentenklasse `letter` zur Ausgabe von „encl.:“ oder „Anlagen:“, gefolgt von einer Liste der angegebenen *Anlagen*.

`\end{`*Umgebung*`}` [Z] S. 9

Beendigungsbefehl für eine mit `\begin{`*Umgebung*`}` eröffneten Umgebung.

`\enlargethispage{`*Maß*`}` [Z] S. 18

Verlängert die aktuelle Seite um das angegebene *Maß*.

`\epsilon` [M] erzeugt ϵ S. 129

`\epsfig{`*Optionen*`}` ???

Befehl des `epsfig`-Paketes. Bindet die mit der Option `file=`*Dateiname* gewählte eps-Datei in das Dokument ein. Die weiteren Optionen werden in Abschnitt 9.3.1 auf Seite 164 beschrieben.

`\equiv` [M] erzeugt $\equiv$ S. 131

`\eta` [M] erzeugt η S. 129

`\evensidemargin` [P] S. 14

Linker Rand bei geraden Seiten. Wird nur wirksam bei zweiseitigem Druck (Klasse `book` oder Option `twoside`). Wertänderung erfolgt mit `\setlength`.

`\exists` [M] erzeugt $\exists$ S. 132

`\exp` [M] S. 107

Befehl zur Erzeugung des Funktionsnamens „exp“ in mathematischen Formeln.

`\extracolsep{`*Breite*`}` S. 89

Befehl zur Definition einer zusätzlichen Breite bei allen nachfolgenden Spalten einer Tabelle. Der Befehl wird als @-Ausdruck der Spaltendeklaration übergeben: `\begin{tabular}{l@{\extracolsep{1mm}}llrc}`

`\extrarowheight` S. 89

Zusätzlicher Wert, der zur normalen Höhe einer Tabellenzeile addiert wird. Bestandteil des Paketes `array`.

F

`\fbox{`*Text*`}` erzeugt einen Rahmen um $\boxed{\textit{Text}}$ S. 58

`\fboxrule` S. 61
Definiert die Liniendicke für Rahmenboxen. Wertänderungen erfolgen mit `\setlength` oder `\addtolength`.

`\fboxsep` S. 60
Definiert den Abstand zwischen Rahmen und Text in einer Rahmenbox. Wertänderungen erfolgen mit `\setlength` oder `\addtolength`.

`\fcolorbox{`*Rahmen*`}{`*Farbe*`}{`*Text*`}` ???
Befehl des `color`-Paketes. Erzeugt eine Rahmenbox mit der Rahmenfarbe *Rahmen* und der Hintergrundfarbe *Farbe*. *Text* wird in der aktuellen Vordergrundfarbe besetzt.

`\fill`
Elastisches Maß, das die natürliche Länge „Null Einheiten" hat und auf jede beliebige Länge gedehnt werden kann. Wird z.B. in Platzbefehlen verwendet: `\vspace{\fill}` oder `\rule{2mm}{\fill}`

`\filltype{`*Fülltyp*`}` ???
Befehl des `eepic`-Paketes. Der Befehl legt die Bereichsfüllung von `\circle*`- und `\ellipse*`-Befehlen fest. Erlaubte Werte sind: `black` (Standardwert), `white` und `shade`.

`\flat` [M] erzeugt $\flat$ S. 132

`\floatpagefraction`
Mindestanteil einer Gleitobjektseite, der von Gleitobjekten eingenommen werden muß (voreingestellt auf 0.5). Damit wird der Freiraum auf der Seite begrenzt. Änderungen erfolgen mit `\renewcommand`.

`\floatsep`
Elastischer vertikaler Abstand zwischen zwei Gleiobjekten auf einer Seite. Wertänderungen erfolgen mit `\setlength` oder `\addtolength`.

`\flq`
Erzeugt halbe öffnende französische Anführungszeichen: ‹

`\flqq`
Erzeugt öffnende französische Anführungszeichen: «

`\flushbottom`
Der Befehl veranlaßt LaTeX dazu, die Absätze auf einer Seite so zu setzen, daß auf allen Seiten die ersten und letzten Zeilen jeweils in einer Höhe sind. Standard bei der Dokumentenklasse `book` und der Option `twoside`.

`\fontencoding{`*Kodierung*`}` S. 32

Definiert die Zeichnsatzkodierung bei der Auswahl einer Schrift. Zur Auswahl stehen: `T1`, `OT1`, `OML`, `OMS`, `OMX`, `U`. Der Wert ist abhängig von dem verwendeten Zeichensatz. Die gewählte Schrift muß mit `\selectfont` aktiviert werden.

`\fontfamily{`*Familie*`}` S. 32

Definiert die Schriftfamilie. *Familie* bezeichnet den Namen, wie er in den internen Tabellen deklariert ist (z.B. `cmr` für die Computer Modern Roman Schriftfamilie). Der Name der Familie wird auch für den Namen der Zeichensatzdatei verwendet. Die gewählte Schrift muß mit `\selectfont` aktiviert werden.

`\fontseries{`*Serie*`}` S. 32

Definiert die Schriftserie. Diese legt die Schriftbreite und die Schriftstärke fest. Diese Attribute können also nicht getrennt verändert werden. Ein Wechsel z.B. von normaler Schrift auf Fettdruck ändert somit auch die verwendete Breite der Schrift, was der Praxis des Buchdesigns entspricht. Welche Werte erlaubt sind, hängt von der gewählten Schriftfamilie ab. Die gewählte Schrift muß mit `\selectfont` aktiviert werden.

`\fontshape{`*Form*`}` S. 32

Definiert die Schriftform („normal", „kursiv", ...). Besteht aus einem oder zwei Buchstaben. Die gewählte Schrift muß mit `\selectfont` aktiviert werden.

`\fontsize{Größe}{`*Durchschuß*`}` S. 32

Definiert die Schriftgröße einer gewählten Schrift. *Größe* gibt dabei die gewünschte Größe an, *Durchschuß* wird für den Zeilenabstand verwendet. Beide Werte müssen eine Einheit haben. Wird diese weggelassen, so wird automatisch `pt` angenommen. *Größe* muß eine in der zum Zeichensatz gehörenden Schriftdefinition (`fd`-Datei) erlaubten Größe sein. Bei skalierbaren Zeichensätzen wie z.B. PostScript kann jeder beliebige Wert eingesetzt werden. Die gewählte Schrift muß mit `\selectfont` aktiviert werden.

`\footheight` [P]

Definiert die Höhe der Fußzeile. Wertänderung mit `\setlength` oder `\addtolength`.

`\footnote[`*Nummer*`]{`*Text*`}` [Z] S. 48

Erzeugt eine Fußnote mit dem übergebenen *Text*. Wird zusätzlich der optionale Parameter *Nummer* angegeben, so wird die automatische Numerierung unterdrückt und statt dessen der angegebene Wert verwendet. Der Zähler für Fußnoten wird dabei nicht verändert.

`\footnotemark[`*Nummer*`]` [Z] S. 49

Erzeugt eine Fußnotenmarkierung im Text. Wird der optionale Parameter *Nummer* angegeben, so wird dieser verwendet und die automatische Numerierung der Fußnote unterdrückt. Mit diesem Befehl ist es zusammen mit `\footnotetext` möglich, auch in Strukturen, in denen der Befehl `\footnote` verboten ist (z.B. LR-Boxen, Tabellen, math. Formeln, ...) Fußnoten zu verwenden.

`\footnoterule` [Z]
Interner Befehl, zur Erzeugung eines horizontalen Striches zwischen dem Seitentext und den Fußnoten. Eine Änderung erfolgt mit `\renewcommand`.

`\footnotesep` [Z]
Höhe einer Linie ohne horizontaler Ausdehnung, die am Anfang jeder Fußnote plaziert wird. Ist diese höher als der Zeilenabstand, so erscheinen die Fußnoten mit einem vertikalen Zwischenraum. Wertänderung erfolgt mit `\renewcommand{\footnotesep}{`*`Neue Höhe`*`}`

`\footnotesize` [Z] S. 31
Schriftgrößenbefehl, der auf die in Fußnoten verwendete Schriftgröße umschaltet.

`\footnotetext[`*`Nummer`*`]{`*`Fußnotentext`*`}` S. 49
Erzeugt eine Fußnote, ohne eine Markierung im laufenden Text zu setzen. Die Fußnote erhält den aktuellen Wert des Fußnotenzählers, der jedoch nicht verändert wird. Bei Angabe des optionalen Parameters *`Nummer`* wird dieser anstelle des aktuellen Werten verwendet. Mit diesem Befehl ist es zusammen mit `\footnotemark` möglich, auch in Strukturen, in denen der Befehl `\footnote` verboten ist (z.B. LR-Boxen, Tabellen, math. Formeln, ...) Fußnoten zu verwenden.

`\footrulewidth{`*`Breite`*`}` S. 17
Legt bei Verwendung des `fancyheadings`-Paketes die Dicke der Linie im Seitenfuß fest. Voreinstellung ist 0pt.
Siehe auch `\headrulewidth`.

`\footskip` [P] S. 14
Abstand zwischen der Unterkante des Seitenrumpfes zur Unterkante der Fußzeile. Wertänderung mit `\setlength` oder `\addtolength`.

`\forall` [M] erzeugt $\forall$ S. 132

`\frac{`*`Zähler`*`}{`*`Nenner`*`}` [M] S. 106
Erzeugt einen mathmatischen Bruch.

`\frakfamily` S. 34
Schaltet bei Verwendung des `oldgerm`-Paketes auf Fraktur-Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\frame{`*`Text`*`}` [Z] S. 59
Erzeugt einen Rahmen ohne Zwischenraum um *`Text`*

`\framebox[`*`Breite`*`][`*`Pos`*`]{`*`Text`*`}` S. 58
Erzeugt eine Rahmenbox der angegeben Breite. Wird dieser Parameter weggelassen, so wird die Breite automatisch berechnet. Der optionale

Parameter *Pos* legt die Ausrichtung des Textes innerhalb des Rahmens fest. Es kann `l` für linksbündigen Text oder `r` für rechtsbündigen Text angegeben werden. Außerdem kann mit `s` festgelegt werden, daß der Text innerhalb der Box soweit gedehnt wird, daß er die gesamte Box Platz ausfüllt.

`\framebox(`*Breite* `,`*Höhe*`[`*Pos*`]{`*Inhalt*`})` ???

Rahmenbox der `picture`-Umgebung mit der angegebenen *Breite* und *Höhe*. Ohne Angabe von *Pos* erscheint der Inhalt horizontal und vertikal zentriert in der Box. Der optionale Parameter *Pos* läßt mit `l`, `r`, `t`, `b` oder einer Paarkomibation (z.B. `rt`) eine Änderung zu.

`\frown` [M] erzeugt $\frown$ S. 131

`\frq`

Erzeugt halbe schließende französische Anführungszeichen: ›

`\frqq`

Erzeugt ganze schließende französische Anführungszeichen: »

`\fussy`

Schaltet auf die normale Formatierung zurück, wenn vorübergehend mit `\sloppy` größere Wortabstände erlaubt worden sind.

G

`\Gamma` [M] erzeugt Γ S. 129

`\gamma` [M] erzeugt γ S. 129

`\gcd` [M] S. 107

Erzeugt den Funktionsnamen „gcd“ in mathematischen Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine untere Grenzangabe.

`\ge` [M] erzeugt $\geq$ S. 131

`\geq` [M] erzeugt $\geq$ S. 131

`\gets` [M] erzeugt $\leftarrow$

`\gg` [M] erzeugt $\gg$ S. 131

`\glossary{`*Eintrag*`}` [Z]

Schreibt einen `\glossaryentry` Befehl in eine `.glo`-Datei.

`\glossaryentry{`*Eintrag*`}{`*Seitenzahl*`}`

Wird von einem `\glossary`-Befehl erzeugt.

`\glq`

Erzeugt halbe öffnende deutsche Anführungszeichen (unten): ‚

`\glqq`
Erzeugt ganze öffnende deutsche Anführungszeichen (unten): „
(Gleichwertig mit `"‘`).

`\gothfamily` S. 34
Schaltet bei Verwendung des `oldgerm`-Paketes auf gotische-Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\grave{`*a*`}` [M] S. 130
Erzeugt einen Gravis-Akzent über *a*: $\grave{a}$

`\grid(`*Breite*`,`*Höhe*`)(`Δ*Breite*`,`Δ*Höhe*`)[`*X-Start*`,`*Y-Start*`]` ???
Zeichenbefehl des `epic`-Paketes. Erzeugt ein Raster mit der angegebenen *Breite* und *Höhe*, in dem horizontale Linien im Abstand Δ*Höhe* und vertikale im Abstand Δ*Breite* gezeichnet werden. Werden die optionalen Parameter *X-Start* und *Y-Start* angegeben, so werden die Linien zusätzlich mit Nummern versehen.

`\grq`
Erzeugt halbe schließende deutsche Anführungszeichen (oben): ‘

`\grqq`
Erzeugt ganze schließende deutsche Anführungszeichen (oben): “
(Gleichwertig mit `"’`)

H

`\H{`*o*`}` S. 29
Erzeugt einen ungarischen Doppelpunktakzent: `\H{o}` = ő

`\hat{`*c*`}` [M] S. 130
Erzeugt ein Cirkumflex über *c*: $\hat{c}$

`\hbar` [M] erzeugt $\hbar$ S. 132

`\headheight` [P] S. 14
Definiert die Höhe der Kopfzeile. Wertänderung erfolgt mit `\setlength` oder `\addtolength`.

`\headrulewidth{`*Breite*`}` S. 17
Legt bei Verwendung des `fancyheadings`-Paketes die Dicke der Linie im Seitenkopf fest. Voreinstellung ist 0.4pt.
Siehe auch `\footrulewidth`.

`\headsep` [P] S. 14
Definiert den Abstand zwischen der Unterkante der Kopfzeile und der Oberkante des Seitentextes. Wertänderung erfolgt mit `\setlength` oder `\addtolength`.

`\heartsuit` [M] erzeugt $\heartsuit$ S. 132

`\hfill` S. 19
Erzeugt einen horizontalen Zwischenraum beliebiger Länge innerhalb einer Textzeile. Der Befehl ist eine Abkürzung für `\hspace{\fill}`.

`\hline` S. 85
Erzeugt eine horizontale Linie über die gesamte Tabellenbreite in Tabellenumgebungen.

`\hhline{`*`Deklaration`*`}` S. 85
Erweitert die Möglichkeiten, horizontale Linien in Tabellen zu definieren. Bestandteil des `hhline`-Paketes.

`\hom` [M] S. 107
Erzeugt den Funktionsnamen „hom“ in mathematischen Funktionen.

`\hoffset` [P] S. 14
Definiert den linken Bezugsrand der Seitenformatierung. Ist standardmäßig auf 0pt eingestellt. Der Wert dient der Korrektur der Druckerzuführung.

`\hookleftarrow` [M] erzeugt $\hookleftarrow$ S. 132

`\hookrightarrow` [M] erzeugt $\hookrightarrow$ S. 132

`\hrulefill` S. 20
Füllt Rest einer Zeile bis zur vollen Zeilenlänge mit einem horizontalen Strich aus.

`\hspace{`*`Breite`*`}` S. 19
Erzeugt einen horizontalen Zwischenraum der angegebenen *`Breite`*. Zwischenräume am Anfang oder Ende einer Zeile werden unterdrückt.

`\hspace*{`*`Breite`*`}` S. 19
Erzeugt einen horizontalen Zwischenraum der angegebenen *`Breite`*, auch wenn dieser am Anfang oder Ende einer Seite steht.

`\Huge` [Z] S. 31
Schaltet auf die größte verfügbare Schriftgröße.

`\huge` [Z] S. 31
Schaltet auf die Schriftgröße `huge`; größer als `LARGE` aber kleiner als `Huge`.

`\hyphenation{`*`Wortliste`*`}` [P] S. 26
Legt ein Trennverzeichnis an. Die in *`Wortliste`* übergebenen Worte können im Dokument nur an den durch `-` gekennzeichneten Stellen getrennt werden.

I

`\i` erzeugt ı S. 29

`\iff` [M] erzeugt $\iff$ S. 132

`\Im` [M] erzeugt $\Im$ S. 132

`\imath` [M] erzeugt $\imath$ S. 132

`\in` [M] erzeugt $\in$ S. 131

`\include{`*Dateiname*`}` [Z]
: Fügt den Inhalt der Datei mit dem Grundnamen *Dateiname* an Stelle des Befehls ein. Die eingelesene Datei beginnt immer mit einer neuen Seite (identisch mit `\clearpage \input{`*Dateiname*`} \clearpage`).

`\includeonly{`*Dateiliste*`}` [P]
: Legt (durch Kommata getrennt) fest, welche Dateien im Textteil durch `\include`-Befehle eingelesen werden dürfen. Soll mit `\include` eine Datei eingelesen werden, die nicht in der *Dateiliste* steht, so ist der `\include`-Befehl identisch mit `\clearpage`.

`\indent`
: Rückt die erste Zeile des *nächsten* Absatzes ein.

`\index{`*Eintrag*`}` [Z]
: Schreibt einen `\indexentry`-Befehl in die `.idx`-Datei. Verwendet man das Programm MakeIndex, so sind folgende Eintragsformen möglich:
`\index{`*Haupteintrag*`}`
`\index{`*Haupteintrag*`!`*Untereintrag*`}`
`\index{`*Haupteintrag*`!`*Untereintrag*`!`*Unter-untereintrag*`}`
MakeIndex erzeugt daraus eine lexikalisch geordnete `theindex`-Umgebung mit `\item`-, `\subitem`- und `\subsubitem`-Einträgen.

`\indexentry{`*Eintrag*`}{`*Seitenzahl*`}`
: Wird von `\index` automatisch in die `.idx`-Datei geschrieben.

`\indexspace`
: Erzeugt eine Leerzeile innerhalb einer `theindex`-Umgebung.

`\initfloatingfigs` S. 70
: Befehl des `floatfig`-Paketes. Ermöglicht die korrekte Bearbeitung von `floatingfigure`-Umgebungen in einem Dokument. Der Befehl muß vor der ersten `floatingfigure`-Umgebungen auftreten.

`\inf` [M] S. 107
: Erzeugt den Funktionsnamen „inf“ in mathematischen Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine untere Grenzangabe.

`\infty` [M] erzeugt ∞ S. 132

`\input{Dateiname}` [Z]
Fügt den Inhalt der Datei mit dem angegebenen Grundnamen *Dateiname* an Stelles des Befehls in den laufenden Text ein. Eine mit `\input` eingelesene Datei darf weitere `\input`-Befehle enthalten.

`\int` [M] erzeugt $\int$ S. 106

`\intextsep` S. 71
Definiert den elastischen vertikalen Abstand zwischen Gleitobjekten und dem umgebenden Text. Eine Wertänderung erfolgt mit `\setlength` oder `\addtolength`.

`\iota` [M] erzeugt ι S. 129

`\item[Marke]` [Z]
Erzeugt eine Markierung in einer listenartigen Umgebung. Ohne Angabe des Parameters *Marke* erfolgt die Markierung in der der Umgebung entsprechenden Art, andernfalls wird *Marke* verwendet.

`\item{Eintrag}`
Erzeugt einen Haupteintrag in einer `theindex`-Umgebung.

`\itemindent`
Definiert den Betrag, um den die Markierung und der nachfolgende Text der ersten Zeile nach jedem `\item`-Befehl eingerückt wird. Die Voreinstellung beträgt 0pt. Wertänderung mit `\setlength` oder `\addtolength`.

`\itemsep`
Zusätzlich zu `\parsep` eingefügter elastischer vertikaler Abstand zwischen dem vorangehenden Listentext und der nächsten Markierung. Wertänderung mit `\setlength` oder `\addtolength`.

`\itshape` S. 30
Schaltet auf *italic*-Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

J

`\j` erzeugt ȷ S. 29

`\jmath` [M] erzeugt $\jmath$ S. 132

`\Join` [M] erzeugt $\bowtie$ (wird vom Paket `latexsym` bereitgestellt) S. 131

`\jot` [M]
Vertikaler Zwischenraum zwischen den Zeilen einer mit `eqnarray` erzeugten Formelgruppe. Voreingestellt ist 3pt. Wertänderung mit `\setlength` oder `\addtolength`.

`\jput(`*x*`,`*y*`){`*Objekt*`}` ???

Zeichenbefehl des `epic`-Paketes, der in einer der drei Umgebungen `dottedjoin`, `dashjoin` und `dashjoin` verwendet wird. Der Befehl erzeugt das angegebene *Objekt* am Punkte `(`*x*`,`*y*`)` und verbindet die Punkte mit einer der Umgebung entsprechenden Linie.

K

`\kappa` [M] erzeugt κ S. 129

`\ker` [M] S. 107

Erzeugt den Funktionsnamen „ker“ in mathematischen Formeln.

`\kill` S. 57

Verhindert das Setzen einer zur Erzeugung der Tabulatorstopps benutzten Musterzeile innerhalb einer `tabbing`- Umgebung.

L

`\L` erzeugt Ł S. 29

`\l` erzeugt ł S. 29

`\label{`*Marke*`}` S. 45

Bringt eine unsichtbare Marke *Marke* im Text an, auf die mit Befehlen wie `\ref{`*Marke*`}` verwiesen werden kann.

`\labelenum`*n* S. 55

Dient zur Definition der Standardmarkierungen der `enumerate`-Umgebung für alle Schachtelungstiefen mit *n* als `i`, `ii`, `iii` und `iv`. Die Neudefinition erfolgt mit `\renewcommand{\labelenumi}{\arabic}`

`\labelitem`*n* S. 55

Dient zur Definition der Standardmarkierungen der `itemize`-Umgebung für alle Schachtelungstiefen mit *n* als `i`, `ii`, `iii` und `iv`. Die Neudefinition erfolgt mit `\renewcommand{\labelitemi}{\arabic}`

`\labelsep`

Definiert innerhalb einer `list`-Umgebung den Abstand zwischen dem Markierungsfeld und dem Listentext. Eine Wertänderung erfolgt mit `\setlength` oder `\addtolength`.

`\labelwidth`

Breite des Markierungsfeldes in einer `list`-Umgebung. Wertänderung erfolgt mit `\setlength` oder `\addtolength`.

`\Lambda` [M] erzeugt Λ S. 129

`\lambda` [M] erzeugt λ S. 129

`\langle` [M] erzeugt $\langle$ S. 130

`\LARGE` [Z] S. 31
Umschaltbefehl auf die Schriftgröße `\LARGE`

`\Large` [Z] S. 31
Umschaltbefehl auf die Schriftgröße `\Large`

`\large` [Z] S. 31
Umschaltbefehl auf die Schriftgröße `\large`

`\LaTeX` erzeugt LaTeX

`\LaTeXe` erzeugt LaTeX 2$_\varepsilon$

`\layout` S. 14
Erzeugt ein im Verhältnis 1:2 verkleinertes Bild des aktuellen Seitenformates mit der Ausgabe der wichtigsten Seitenmaße. Bestandteil des `layout`-Paketes.

`\lceil` [M] erzeugt $\lceil$ S. 130

`\ldots` [M] erzeugt $\ldots$ S. 106

`\le` [M] erzeugt $\le$

`\leadsto` [M] erzeugt $\leadsto$ S. 132

`\leaf{`*Kopftext*`}{`*Seitentext*`}{`*BlattID*`}` ???
Befehl des `trees`-Paketes. Die Parameter *Kopftext Seitentext* werden über bzw. neben dem aktuellen Blatt ausgegeben, das den internen Bezeichner *BlattID* hat.

`\left`*klammer* [M]
Erzeugt ein linkes Klammernsymbol, das an die durch `\left` und `\right`-Paare eingeschlossene Teilformel in der Größe angepaßt ist. Soll ein Klammernsymbol nicht gepaart auftreten, so kann mit `\left.` bzw. `\right.` eine unsichtbare Klammer erzeugt werden.

`\Leftarrow` [M] erzeugt $\Leftarrow$ S. 132

`\leftarrow` [M] erzeugt $\leftarrow$ S. 132

`\lefteqn` [M]
Befehl innerhalb einer `eqnarray`-Umgebung, nach dem die zweite und alle weitere Formelzeilen leicht eingerückt werden.

`\leftharpoondown` [M] erzeugt $\leftharpoondown$ S. 132

`\leftharpoonup` [M] erzeugt $\leftharpoonup$ S. 132

`\leftline{`*Text*`}`
: TeX-Befehl, der den angegebenen *Text* in einer eigenen Zeile linksbündig setzt.

`\leftmargin`
: Die Einrücktiefe des linken Randes innerhalb einer `list`-Umgebung. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`. Bei verschachtelten Umgebungen können die verschiedenen Schachtelungstiefen durch Anhängen von `i ... iv` gekennzeichnet werden.

`\Leftrightarrow` [M] erzeugt $\Leftrightarrow$ S. 132

`\leftrightarrow` [M] erzeugt $\leftrightarrow$ S. 132

`\leq` [M] erzeugt $\leq$ S. 131

`\lfloor` [M] erzeugt $\lfloor$ S. 130

`\lfoot[`*LF-gerade*`]{`*LF-ungerade*`}` S. 16
: Definiert den linken Teil der Fußzeile beim Seitenstil `fancy`. Der optionale Parameter *LF-gerade* kann genutzt werden, um bei zweiseitigem Druck die Ausgabe auf geraden Seiten zu ändern.
Siehe auch `\chead`, `\lhead`, `\rhead`, `\cfoot`, `\rfoot`

`\lg` [M] erzeugt den Funktionsnamen „lg" in Formeln. S. 107

`\lhd` [M] erzeugt $\lhd$ S. 131

`\lhead[`*LK-gerade*`]{`*LK-ungerade*`}` S. 16
: Definiert den linken Seitenkopf beim Seitenstil `fancy`. Der optionale Parameter *LK-gerade* kann genutzt werden, um bei zweiseitigem Druck die Ausgabe auf geraden Seiten zu ändern.
Siehe auch `\chead`, `\rhead`, `\cfoot`, `\lfoot`, `\rfoot`

`\lim` [M] S. 107
: Befehl zur Erzeugung des Funktionsnamen „lim" in Formeln. Kann mit unterer Grenzangabe kombiniert werden.

`\liminf` [M] S. 107
: Befehl zur Erzeugung des Funktionsnamen „lim inf" in Formeln. Kann mit unterer Grenzangabe kombiniert werden.

`\limits` [M] S. 107
: Setzt die obere und untere Grenze über bzw. unter den zugeordneten Symbolen an, wenn diese standardmäßig hinter den Symbolen gesetzt werden.

`\limsup` [M] S. 107
: Befehl zur Erzeugung des Funktionsnamen „lim sup" in Formeln. Kann mit unterer Grenzangabe kombiniert werden.

`\line(`Δx`,`Δy`){`*Länge*`}`
: Bildbefehl innerhalb einer `picture`-Umgebung zur Erzeugung von Linien. Es stehen dabei standardmäßig nur eine begrenzte Anzahl von Steigungen zur Verfügung. Bei horizontalen und vertikalen Linien stellt *Länge* die tatsächliche Länge dar, bei geneigten Linien die Projektion der Linienlänge auf die x-Achse. Die Neigung wird durch die beiden Werte Δx und Δy bestimmt, wobei standardmäßig nur ganzzahlige Werte zwischen -6 und +6 erlaubt sind.

`\linebreak[`*n*`]` S. 18
: Empfehlung an LaTeX, eine Zeile an der Stelle des Befehls beidbündig zu brechen. Die Dringlichkeit der Empfehlung wird mit dem optionalen Parameter *n* gesteuert. Der Parameter kann Werte zwischen 0 und 4 annehmen, wobei die 4 gleichbedeutend mit zwingend ist. Wird der Parameter weggelassen, so ist der Befehl ebenfalls zwingend.

`\linespread{`*Faktor*`}` S. 20
: Vergrößert die Zeilenabstände um *Faktor*. Für „eineinhalbzeilige“ Ausgabe beträgt *Faktor* 1.3, für „zweizeilige“ Ausgabe 1.6.

`\linethickness{`*Stärke*`}` ???
: Legt die Stärke für horizontale und vertikale Striche in einer `picture`-Umgebung fest. Für *Stärke* ist jede positive, feste Längenangabe erlaubt. Die Wirkung des Befehls endet, sobald entweder ein weiterer `\linethickness`-Befehl ein andere Strichstärke auswählt oder einer der beiden Strichstärkenbefehle auftritt.

  Einige von Zusatzpaketen bereitgestellten Zeichenfunktionen beachten den Befehl ebenfalls.

`\listoffigures` S. 43
: Erzeugt das Abbildungsverzeichnis.

`\listoftables` S. 43
: Erzeugt das Tabellenverzeichnis.

`\listparindent`
: Einrücktiefe der ersten Zeile eines Absatzes in einer `list`-Umgebung. Wertänderung erfolgt mit `\setlength` und `\addtolength`.

`\ll` [m] erzeugt $\ll$ S. 130

`\ln` [M] erzeugt den Funktionsnamen „ln“ in Formeln. S. 107

`\log` [M] erzeugt den Funktionsnamen „log“ in Formeln. S. 107

`\Longleftarrow` [M] erzeugt $\Longleftarrow$ S. 132

`\longleftarrow` [M] erzeugt $\longleftarrow$ S. 132

`\Longleftrightarrow` [M] erzeugt $\Longleftrightarrow$ S. 132

`\longleftrightarrow` [M] erzeugt $\longleftrightarrow$ S. 132

`\longmapsto` [M] erzeugt $\longmapsto$ S. 132

`\Longrightarrow` [M] erzeugt $\Longrightarrow$ S. 132

`\longrightarrow` [M] erzeugt $\longrightarrow$ S. 132

`\lq` erzeugt ‘, identisch mit der Taste `‘`.

M

`\makebox[`*`Breite`*`][`*`Pos`*`]{`*`Text`*`}` S. 58

Erzeugt eine rahmenlose Box der angegeben Breite. Wird dieser Parameter weggelassen, so wird die Breite automatisch berechnet. Der optionale Parameter *`Pos`* legt die Ausrichtung des Textes innerhalb des Rahmens fest. Es kann `l` für linksbündigen Text oder `r` für rechtsbündigen Text angegeben werden. Außerdem kann mit `s` festgelegt werden, daß der Text innerhalb der Box soweit gedehnt wird, daß er die gesamte Box ausfüllt.

`\makebox(`*`Breite`*`,`*`Höhe`*`[`*`Pos`*`]{`*`Inhalt`*`})` ???

Bildbefehl zur Erzeugung einer rahmenlosen Box mit der angegebenen *`Breite`* und *`Höhe`*. Ohne Angabe von *`Pos`* erscheint der Inhalt horizontal und vertikal zentriert in der Box. Der optionale Parameter *`Pos`* läßt mit `l`, `r`, `t`, `b` oder einer Paarkomibation (z.B. `rt`) eine Änderung zu.

`\makeglossary` [P]

Befehl zur Aktivierung der `\glossary`-Befehle im Textteil.

`\makeindex` [P]

Befehl zu Aktivierung der `\index`-Befehle im Textteil.

`\makelabels` [P]

Befehl der Dokumentenklasse `letter`. Druckt Addressen in einer 4.25 Zoll breiten und 2 Zoll hohen Box aus. Werden mit dem Dokument mehrere Briefe geschrieben, so werden alle Addressen auf einer eigenen Seite in zwei Spalten nebeneinander gesetzt.

`\maketitle` S. 42

Erzeugt die Ausgabe eines Titels mit den Angaben der `\author`, `\title` und ggf. `\date` und `\thanks`-Befehle. Bei Verwendung der Dokumentenklasse `article` erscheint der Titel in der oberen Hälfte der ersten Seite, bei `report` und `book` wird eine eigene Seite erzeugt. Bei `arictle` kann mit der Klassenoption `titlepage` die Ausgabe einer eigenen Titelseite erreicht werden.

`\mapsto` [M] erzeugt $\mapsto$ S. 132

`\marginpar[`*`linke Notiz`*`]{`*`rechte Notiz`*`}` S. 50

Erzeugt eine Randnotiz rechts vom Text mit dem Inhalt *`rechte Notiz`*. Bei doppelseitigem Layout kann mit Angabe des optionalen Parameters *linke Notiz* erreicht werden, daß auf geraden Seiten dieser Text auf der linken Seite erscheint.

`\marginparpush` S. 14
Der minimale vertikale Abstand zwischen zwei Randnotizen. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\marginparsep` S. 14
Der Abstand zwischen dem Textrand und einer Randnotiz. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\marginparwidth` S. 14
Die Boxbreite einer Randnotiz. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\markboth{`*linker Kopf*`}{`*rechter Kopf*`}` S. 14
Definiert den Textinhalt einer Kopfzeile bei doppelseitigem Druck für die linke und rechte Seite für die Seitenstile `myheadings` und `headings`.

`\markright{`*Kopf*`}` S. 14
Definiert den Textinhalt einer Kopfzeile für die Seitenstile `myheadings` und `headings`. Bei doppelseitigem Layout kann hiermit der Kopf der rechten Seite definiert werden.

`\matrixput`$(x,y)(\Delta x_1,\Delta y_1)\{n_1\}(\Delta x_2,\Delta y_2)\{n_2\}$`{`*Objekt*`}` ???
Positionierungsbefehl des `epic`-Paketes. Er entspricht einem zweidimensionalen `\multiput`-Befehl.

`\max` [M] S. 107
Erzeugt den Funktionsnamen „max“ in Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine untere Grenzangabe.

`\maxovaldiam` ???
Bestandteil des `eepic`-Paketes. Legt den maximalen Durchmesser der Vierteilkreise an den Ecken von Ovalen fest. Voreinstellung: 40pt

`\mbox{`*Text*`}` erzeugt eine LR-Box um *Text* S. 58

`\mdseries` S. 30
Schaltet die Schriftstärke auf medium um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\medskip` [Z] S. 20
Mittlerer vertikaler Abstand vom Betrag `\medskipamount`.

`\medskipamount`
Standardwert für den mit `\medskip` erzeugten vertikalen Zwischenraum.

`\mho` [M] erzeugt $\mho$ S. 132

`\mid` [M] erzeugt $\mid$ S. 131

`\min` [M] S. 107
Erzeugt den Funktionsnamen „min“ in Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine untere Grenzangabe.

`\models` [M] erzeugt $\models$ S. 131

`\mp` [M] erzeugt $\mp$ S. 130

`\mu` [M] erzeugt μ S. 129

`\multicolumn{`*n*`}{`*col*`}{`*Text*`}` S. 85
Faßt die nächsten *n* Spalten einer Tabelle zusammen. Der Parameter *col* definiert die Formatierung des Textes *Text* innerhalb der Zelle.

`\multiput(`*x*`,`*y*`)(`Δx`,`Δy`){`*N*`}{`*Bildobjekt*`}` ???
Erzeugt das angegebene *Bildobjekt* *N*-mal nacheinander bei (*x*,*y*), (*x* + Δx,*y* + Δy), ...

`\multiputlist(`*x*`,`*y*`)(`Δx`,`Δy`)[`*Pos*`]{`*Objekt1,Objekt2,...*`}` ???
Befehl des `epic`-Paketes. Die Verwendung ist ähnlich dem `\multiput`-Befehl. Allerdings werden nacheinander die in der Objektliste angegebenen Objekte gesetzt. Die Objekte werden intern in eine `\makebox(0,0)[`*Pos*`]` gesetzt. Der optionale Parameter *Pos* des `\multiputlist`-Befehls wird an die `\makebox` weitergereicht.

N

`\nabla` [M] erzeugt ∇ S. 132

`\name{`*Autor*`}` S. 76
Definiert in der Dokumentenklasse `letter` den Names des Briefautors.

`\natural` [M] erzeugt $\natural$ S. 132

`\nearrow` [M] erzeugt $\nearrow$ S. 132

`\NeedsTeXFormat{LaTeX2e}[`*Version*`]` [P] S. 9
Zeigt an, daß die verwendete Datei LaTeX 2$_\varepsilon$ ab der angegebenen *Verion* benötigt. Der Befehl *muß* vor `\documentclass` stehen.

`\neg` [M] erzeugt $\neg$ S. 132

`\neq` [M] erzeugt $\neq$ S. 131

`\newcolumntype{`*Name*`}[`*Num*`]{`*Deklaration*`}` S. 92
Erzeugt einen neuen Spaltentyp. *Name* legt dabei einen einzelnen Buchstaben fest, unter dem der Spaltentyp verfügbar sein soll. *Num* gibt die Anzahl der Parameter an, die dem Spaltentyp übergeben werden. *Deklaration* schließlich definiert das Aussehen der Spalte.

`\newcommand{\`*Name*`}[`*N*`]{`*Definition*`}`
Erklärung eines benutzereigenen Befehls mit dem Namen `\`*Name*. Der optionale Parameter *N* legt die Anzahl der Argumente fest, die dem neuen Befehl übergeben werden müssen (*N* $\leq$ 9) und auf die in *Definition* mit `#1` bis `#9` verwiesen werden darf.

`\newcounter{`*Zählername*`}[`*Rücksetzer*`]`
Legt einen neuen Zähler mit dem Namen `\`*Zählername* an, der optional mit jeder Erhöhung des existierenden Zählers *Rücksetzer* wieder auf Null zurückgesetzt wird.

`\newenvironment{`*Name*`}[`*N*`]{`*Beginndef*`}{`*Enddef*`}`
Definiert eine neue Umgebung mit dem Namen *Name* und der `\begin`-Definition *Beginndef* und der `\end`-Definition *Enddef*. Der optionale Parameter *N* (*N* $\leq$ 9) bestimmt die Anzahl der Parameter für die Umgebung, auf die in *Beginndef* mit `#1` bis `#9` verwiesen werden darf.

`\newlength{\`*Längenname*`}`
Legt einen neuen Längenbefehl mit dem Namen `\`*Längenname* an und initialisiert ihn mit der Länge 0pt. Wertzuweisung erfolgt wie bei Längenbefehlen üblich mit `\setlength` oder `\addtolength`.

`\newline` [Z] S. 18
Erzwingt einen Zeilenumbruch *ohne* Randausgleich.

`\newpage` S. 18
Erzwingt einen Seitenumbruch. Der Rest der Seite bleibt leer.

`\newsavebox{\`*Boxname*`}` S. 59
Richtet einen Speicher `\`*Boxname* ein, in den LR-Boxen mit dem Befehl `\savebox` abgespeichert werden können. Die Ausgabe des Boxinhaltes erfolgt mit `\usebox`.

`\newtheorem{`*Name*`}[`*Zähler*`]{`*Satz*`}[`*Zusatzzähler*`]` S. 56
Erzeugt eine neue Umgebung *Name*, mit der eine Theoremaussage gesetzt werden kann, der der fettgedruckte Begriff *Satz*, gefolgt von einer fortlaufenden Nummer, vorangestellt wird. Der optionale Parameter *Zähler* ist der Name einer anderen existierenden Theoremumgebung, die gemeinsam mit dieser gezählt wird. *Zusatzzähler* ist der Name eines Gliederungszählers, dessen momentaner Wert der Numerierung von *Satz* vorangeht und innerhalb der Gliederung die Numerierung mit 1 beginnen läßt.
Siehe auch `\begin{Theorem-Name}`

`\ni` [M] erzeugt $\ni$ S. 131

`\nocite{`*Schlüssel*`}`
Fügt einen Eintrag ins Literaturverzeichnis mit dem Bezug *Schlüssel* auf eine Datenbank, ohne daß im Text darauf verwiesen wird.
Siehe auch `\cite`

`\nofiles` [P]
: Unterdrückt die Erzeugung der Hilfsdateien `.aux`, `.glo`, `.idx`, `.lof`, `.lot` und `.toc`.

`\noindent` S. 18
: Verhindert den horizontalen Einzug der ersten Zeile des *nächsten* Absatzes.

`\nolimits` [M]
: Ordnet die obere und untere Grenzen hinter den zugeordneten Symbolen an, wenn diese standardmäßig über bzw. unter den Symbolen gesetzt werden.

`\nolinebreak[N]` [Z] S. 18
: Empfehlung an LaTeX, die Zeile an dieser Stelle *nicht* umzubrechen. Die Dringlichkeit kann mit dem optionalen Parameter *N* festgelegt werden. *N* kann Werte zwischen 0 und 4 annehmen, wobei 4 gleichbedeutend mit zwingend ist. Wird der Parameter weggelassen, so ist der Umbruch ebenfalls zwingend.

`\nonumber` [M] S. 110
: Die vorangegangene Formel einer `eqnarray`-Umgebung erhält keine Nummer, der interne Zähler wird nicht hochgesetzt.

`\nopagebreak[N]` [Z] S. 18
: Empfehlung an LaTeX, die Seite an dieser Stelle *nicht* umzubrechen. Die Dringlichkeit kann mit dem optionalen Parameter *N* festgelegt werden. *N* kann Werte zwischen 0 und 4 annehmen, wobei 4 gleichbedeutend mit zwingend ist. Wird der Parameter weggelassen, so ist der Umbruch ebenfalls zwingend.

`\normalem` S. 25
: Schaltet nach einem `\ULem`-Befehl wieder zurück auf die normalen Funktionen zum Unterstreichen. Bestandteil des `ulem`-Paketes.

`\normalfont` S. 30
: Schaltet auf die aktuelle Basis-Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\normalsize` [Z] S. 31
: Stellt auf die Schriftgröße `\normalsize` um.

`\not` [M] S. 131
: Negiert das folgende Vergleichssymbol durch Anbringen eines Striches: `\not\approx` = $\not\approx$.

`\notin` [M] erzeugt $\notin$ S. 131

`\nu` [M] erzeugt ν S. 131

`\nwarrow` [M] erzeugt $\nwarrow$ S. 132

O

`\O` erzeugt ffi S. 29

`\o` erzeugt ff S. 29

`\oddsidemargin` [P] S. 14
Linker Rand für ungerade Seiten bei doppelseitigem Druck, bzw. linker Rand für alle Seiten bei einseitigem Druck. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\odot` [M] erzeugt $\odot$ S. 130

`\OE` erzeugt Œ S. 29

`\oe` erzeugt œ S. 29

`\oint` [M] erzeugt $\oint$ S. 130

`\oldstylenums{`*`Jahreszahl`*`}`
Setzt die angegebene *`Jahreszahl`* bei Verwendung des `oldgerm`-Paketes in sog. Mediävalziffern: `\oldstylenums{1972}` = 1972

`\Omega` [M] erzeugt Ω S. 129

`\omega` [M] erzeugt ω S. 129

`\ominus` [M] erzeugt $\ominus$ S. 130

`\onecolumn` [Z] S. 64
Beginnt eine neue Seite und schaltet von zweispaltiger Formatierung auf einspaltige Formatierung um.

`\opening{`*`Anrede`*`}` S. 76
Befehl in der `letter`-Umgebung zum Setzen der Anrede.

`\oplus` [M] erzeugt $\oplus$ S. 130

`\oslash` [M] erzeugt $\oslash$ S. 130

`\otimes` [M] erzeugt $\otimes$ S. 130

`\Ovalbox{`*`Inhalt`*`}` S. 61
Erzeugt eine $\boxed{\text{ovale}}$ Box um *`Inhalt`*. Bestandteil des `fancybox`-Paketes.

`\ovalbox{`*`Inhalt`*`}` S. 61
Erzeugt eine $\boxed{\text{ovale}}$ Box um *`Inhalt`*. Bestandteil des `fancybox`-Paketes.

`\overbrace{`*`Formelteil`*`}` [M] S. 105
Erzeugt eine horizontale Klammer über dem angegebenen Formelteil.

`\overline{`*`Formelteil`*`}` [M] S. 105
Erzeugt eine horizontale Linie über dem angegebenen Formelteil.

P

`\P` erzeugt ¶ S. 131

`\pagebreak[n]` S. 18
Empfehlung, eine Seite an der Stelle des Befehls umzubrechen. Die Dringlichkeit der Empfehlung wird mit dem optionalen Parameter *n* gesteuert. Der Parameter kann Werte zwischen 0 und 4 annehmen, wobei die 4 gleichbedeutend mit zwingend ist. Wird der Parameter weggelassen, so ist der Befehl ebenfalls zwingend.

`\pagecolor{Farbe}` ???
Befehl des `color`-Paketes. Stellt die Hintergrundfarbe der aktuellen und aller nachfolgender Seiten auf *Farbe*. Der Befehl ist global, das bedeutet, er wirkt auch über das Ende der aktuellen Umgebun hinaus und kann nur durch einen weiteren `\pagecolor`-Befehl aufgehoben werden.

`\pagenumbering{Stil}` S. 16
Bestimmt den Stil der Seitennumerierung. Für *Stil* stehen `arabic`, `roman`, `Roman`, `alph` und `Alph` zur Verfügung. Ein Umschalten der Seitennumerierung innerhalb des Textes setzt den Zähler wieder auf 1 zurück.

`\pageref{Marke}` [Z] S. 46
Erzeugt die Seitennummer derjenigen Seite, auf der mit `\label{Marke}` eine unsichtbare Markierung angebracht wurde.

`\pagestyle{Stil}` [P] S. 14
Legt den Seitenstil des Dokumentes fest. Standardmäßig stehen dafür die Stilarten `plain` (Standard), `empty`, `headings` und `myheadings` zur Verfügung. Weitere Stilarten können über Zusatzpakete angeboten werden.

`\par`
Beendet den laufenden Absatz und beginnt einen neuen. Wirkt wie eine Leerzeile.

`\paragraph[Kurzform]{Überschrift}` S. 40
Vorletzter Gliederungsbefehl. Der optionale Parameter *Kurzform* kann verwendet werden, um im Inhaltsverzeichnis ein anderen Text zu setzen.

`\paragraph*{Überschrift}` S. 40
Beginnt einen neuen Abschnitt, ohne ihn zu numerieren. Es erfolgt kein Eintrag ins Inhaltverzeichnis, der interne Zähler bleibt unverändert.

`\parallel` [M] erzeugt ‖ S. 131

`\parbox[Pos]{Breite}{Text}` S. 60
Erzeugt eine vertikale Box der angegebenen Breite, in der der Text entsprechend der Boxbreite beidbündig umgebrochen wird. Der optionale

Parameter *Pos* gibt an, ob die Box mit ihrer obersten Zeile (`t`), oder mit der untersten Zeile (`b`) auf die laufende Zeile ausgerichtet werden soll. Standardmäßig wird die Box vertikal zentriert.

`\parindent`
Einrücktiefe der ersten Zeile eines Absatzes. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\parsep`
In einer `list`-Umgebung der vertikale Abstand zwischen zwei Absätzen. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\parskip`
Elastischer Abstand zwischen zwei Absätzen. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\part[`*Kurzform*`]{`*Überschrift*`}` S. 40
Oberste Gliederungsebene. Er beginnt einen neuen „Teil“ mit einer laufenden Nummer. Die Numerierung der nachfolgenden Gliederungsebenen wird nicht beeinflußt.

`\part*{`*Überschrift*`}` S. 40
Wie `\part`, jedoch erfolgt keine Numerierung und kein Eintrag ins Inhaltsverzeichnis.

`\partial` [M] erzeugt ∂ S. 132

`\partopsep`
Zusätzlicher elastischer Zwischenraum am Beginn uder Ende einer Listenumgebung, wenn dieser eine Leerzeile vorangeht oder folgt. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\path(`x_1`,`y_1`)(`x_2`,`y_2`)...(`x_n`,`y_n`)` ???
Befehl des `eepic`-Paketes. Verbindet die angegebenen Punkte durch eine durchgezogene Linie.

`\perp` [M] erzeugt $\perp$ S. 131

`\Phi` [M] erzeugt Φ S. 129

`\phi` [M] erzeugt ϕ S. 129

`\Pi` [M] erzeugt Π S. 129

`\pi` [M] erzeugt π S. 129

`\pm` [M] erzeugt $\pm$ S. 130

`\pmod{`*Arg*`}` S. 106
Befehl zur Erzeugung des Funktionsnamens „mod“ in Formeln.

`\poptabs`
Innerhalb einer `tabbing`-Umgebung wird hiermit der vorige Satz von Tabulatoren eingestellt. Rückschaltbefehl von `\pushtabs`.

`\pounds` erzeugt £ S. 131

`\Pr` [M]
Befehl zur Erzeugung des Funktionsnames „Pr“ in mathematischen Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine untere Grenzangabe.

`\prec` [M] erzeugt $\prec$ S. 131

`\preceq` [M] erzeugt $\preceq$ S. 131

`\prime` [M] erzeugt $\prime$

`\printindex`
Erzeugt bei der Verwendung von `makeidx` die Ausgabe eines Indexes.

`\prod` [M] erzeugt $\prod$ S. 132

`\propto` [M] erzeugt $\propto$ S. 131

`\protect` S. 124
Schützt zerbrechliche Befehle in wandernden Argumenten.

`\ps` S. 77
Befehl innerhalb der `letter`-Umgebung, um auch nach dem Befehl `\closing` noch Text setzen zu können.

`\psdraft` ???
Befehl des `graphics`-Paketes, das als Basispaket von allen Paketen geladen wird, die PostScript-Fähigkeiten bieten. Der Befehl schaltet den Draft-Modus ein. In diesem Modus wird zum Beispiel die Ausgabe aller mit `\epsfig` eingebundenen Abbildungen unterdrückt. Stattdessen wird ein Rahmen in der Größe der Abbildung gezogen, in dem der Dateiname steht. Der Befehl wirkt solange, bis der Draft-Modus mit dem Befehl `\psfull` ausgeschaltet wird.

`\psfull` ???
Befehl des `graphics`-Paketes, das als Basispaket von allen Paketen geladen wird, die PostScript-Fähigkeiten bieten. Der Befehl schaltet den mit `\psdraft` eingeschalteten Draft-Modus wieder aus.

`\Psi` [M] erzeugt Ψ S. 129

`\psi` [M] erzeugt ψ S. 129

`\pushtabs`
Innerhalb einer `tabbing`-Umgebung wird hiermit der aktuelle Satz von Tabulatoren gespeichert. Er kann mit `\poptabs` wieder aktiviert werden.

`\put(`*x*`,`*y*`){`*Bildobjekt*`}` ???
Bildbefehl der `picture`-Umgebung. Erzeugt das angegebene *Bildobjekt* an den Koordinaten `(`*x*`,`*y*`)`.

Q

`\quad` S. 19
: Horizontaler Abstand der Größe 1em.

`\qquad` S. 19
: Horizontaler Abstand der Größe 2em.

R

`\raggedbottom`
: Standardformatierung in den Dokumentenklassen `article`, `report` und `letter` bei einseitigem Druck. Die unterste Zeile variiert von Seite zu Seite. Gegenbefehl zu `\flushbottom`.

`\raggedleft` S. 23, 91
: Bewirkt, daß sämtlicher nachfolgender Text rechtsbündig formatiert wird. Ein Zeilenumbruch erfolgt ohne Silbentrennung. Die einzelnen Zeilen können durch \\ getrennt werden.
Siehe auch `\begin{flushright}`.

`\raggedright` S. 23, 91
: Bewirkt, daß sämtlicher nachfolgender Text linksbündig formatiert wird. Ein Zeilenumbruch erfolgt ohne Silbentrennung. Die einzelnen Zeilen können durch \\ getrennt werden.
Siehe auch `\begin{flushleft}`.

`\raisebox{`*`Lift`*`}[`*`Oberlänge`*`][`*`Unterlänge`*`]{Text}` S. 63
: Erzeugt eine LR-Box, deren Inhalt vertikal um *`Lift`* gegenüber der Grundlinie der aktuellen Zeile verschoben wurde. Die optionalen Parameter bewirken, daß unabhängig von den tatsächlichen Ausmaßen der Box angenommen wird, daß die Box um *`Oberlänge`* über und um *`Unterlänge`* unter die laufende Grundlinie reicht. Diese Werte werdeb bei der Bestimmung des Zeilenabstandes beachtet.

`\rangle` [M] erzeugt $\rangle$ S. 130

`\rceil` [M] erzeugt $\rceil$ S. 130

`\Re` [M] erzeugt $\Re$ S. 132

`\ref{`*`Marke`*`}` S. 46
: Erzeugt eine Gliederungs-, Gleichungs, Bild- oder Tabellennummer, je nachdem, wo mit `\label{`*`Marke`*`}` eine Markierung angebracht wurde.

`\reflectbox{`*Text*`}` ???

Befehl des `graphics`-Paketes. Spiegelt *Text* um eine vertikale Achse. Entspricht `\scalebox{-1}[1]{`*Text*`}`.

`\refstepcounter{`*Zähler*`}`

Erhöht den angegebenen Zähler ebenso wie `\stepcounter`. Gleichzeitig wird mit `\label`-`\ref`-Paaren auf den aktuellen Wert von *Zähler* Bezug genommen.

`\renewcommand{\`*Befehl*`}[`*N*`]{`*Definition*`}`

Definiert den bestehenden Befehl `\`*Befehl* neu. Der optionale Parameter *N* legt die Anzahl der Argumente fest, die dem neuen Befehl übergeben werden müssen (*N* $\leq$ 9) und auf die in *Definition* mit `#1` bis `#9` verwiesen werden darf.

`\renewenvironment`

Definiert die existierende Umgebung mit dem Namen *Name* mit der `\begin`-Definition *Beginndef* und der `\end`-Definition *Enddef* neu. Der optionale Parameter *N* (*N* ≤ 9) bestimmt die Anzahl der Parameter für die Umgebung, auf die in *Beginndef* mit `#1` bis `#9` verwiesen werden darf.

`\resizebox*{`*Breite*`}{`*Höhe*`}{`*Text*`}` ???

Befehl des `graphics`-Paketes. Skaliert *Text* auf die angegebene absolute *Breite* und *Höhe*. Wird für eine der Längenangaben `!` angegeben, so wird für beide Werte der gleiche Skalierungsfaktor verwendet, so daß das Längen-Breiten-Verhältnis erhalten bleibt.

`\reversemarginpar`

Ordnet alle nachfolgenden Randnotizen auf der linken bzw. inneren Seite des Textes an. Der Rückschaltbefehl lautet: `\normalmarginpar`.

`\rfloor` [M] erzeugt $\rfloor$ S. 130

`\rfoot[`*RF-gerade*`]{`*RF-ungerade*`}` S. 16

Definiert den linken Teil der Fußzeile beim Seitenstil `fancy`. Der optionale Parameter *LF-gerade* kann genutzt werden, um bei zweiseitigem Layout die Ausgabe auf geraden Seiten zu ändern.
Siehe auch `\chead`, `\lhead`, `\rhead`, `\cfoot`, `\lfoot`

`\rhd` [M] erzeugt $\rhd$ S. 131

`\rhead[`*RK-gerade*`]{`*RK-ungerade*`}` S. 16

Definiert den rechten Seitenkopf beim Seitenstil `fancy`. Der optionale Parameter *RK-gerade* kann genutzt werden, um bei zweiseitigem Layout die Ausgabe auf geraden Seiten zu ändern.
Siehe auch `\chead`, `\lhead`, `\cfoot`, `\lfoot`, `\rfoot`

`\rho` [M] erzeugt ρ S. 129

\right*klammer* [M]
Erzeugt ein rechtes Klammernsymbol, das an die durch \left und \right-Paare eingeschlossene Teilformel in der Größe angepaßt ist. Soll ein Klammernsymbol nicht gepaart auftreten, so kann mit \left. bzw. \right. eine unsichtbare Klammer erzeugt werden.

\Rightarrow [M] erzeugt $\Rightarrow$ S. 132

\rightarrow [M] erzeugt $\rightarrow$ S. 132

\rightharpoondown [M] erzeugt $\rightharpoondown$ S. 132

\rightharpoonup [M] erzeugt $\rightharpoonup$ S. 132

\rightleftharpoons [M] erzeugt $\rightleftharpoons$ S. 132

\rightline{*Text*}
TEX-Befehl, der den angegebenen *Text* in einer eigenen Zeile rechtsbündig setzt.

\rightmargin
Der Abstand des rechten Randes einer list-Umgebung zum rechten Rand des äußeren Textes. Wertzuweisung erfolgt mit \setlength oder \addtolength. Bei verschachtelten Umgebungen können die verschiedenen Schachtelungstiefen durch Anhängen von i ... iv gekennzeichnet werden.

\rmfamily S. 30
Schaltet auf die Serifenschrift um. Üblicherweise ist dies die Standardschrift. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

\Roman{*Zähler*}
Druckt den aktuellen Wert von *Zähler* als große römische Zahl.

\roman{*Zähler*}
Druckt den aktuellen Wert von *Zähler* als kleine römische Zahl.

\root(*X*,*Y*) *WurzelID*. ???
Befehl des trees-Paketes. Definiert die absoluten Koordinaten der Wurzel. Die Klammern, das Leerzeichen vor *WurzelID* und der Punkt sind obligatorisch.

\rotcaption{*Überschrift*} ???
Befehl des rotating-Paketes. Dreht die Überschrift eines Gleitobjektes um 90°. Der Befehl wird verwendet, wenn der Inhalt eines Gleitobjektes gedreht werden soll, hierzu aber nicht eine der Umgebungen sidewaysfigure oder sidewaystable verwendet wird.

\rq erzeugt ', identisch mit der '-Taste.

\rule[*Lift*]{*Breite*}{*Höhe*} S. 62
Erzeugt ein schwarzes Recheck der angegebenen *Breite* und *Höhe*, das um den optionalen Parameter *Lift* vertikal gegenüber der aktuellen Grundlinie verschoben werden kann.

S

`\S` erzeugt § S. 131

`\samepage`
Ab diesem Befehl ist ein Seitenumbruch nur an eingeschränkten Stellen möglich, z.B. zwischen Absätzen, sofern er nicht ausdrücklich mit `\newpage` oder `\pagebreak` erzwungen wird.
Siehe auch `\begin{samepage}`

`\savebox{\`*Boxname*`[`*Breite*`][`*Pos*`]{`*Text*`}` S. 59
Speichert *Text* in der durch `\newsavebox` eingerichteten Box ab. Der Inhalt kann jederzeit mit `\usebox{\`*Boxname*`}` wieder ausgegeben werden.

`\sbox{\`*Boxname*`}{`*Text*`}` S. 59
Speichert *Text* in der durch `\newsavebox` eingerichteten Box als LR-Box ab. Der Inhalt kann jederzeit mit `\usebox{\`*Boxname*`}` wieder ausgegeben werden.

`\scalebox{`*Faktor-Breite*`}[`*Faktor-Höhe*`]{`*Text*`}` ???
Befehl des `graphics`-Paketes. Skaliert *Text* um den angegebenen Faktor *Faktor-Breite*. Wird zusätzlich das optionale Argument *Faktor-Höhe* angegeben, so wird der Text vertikal anders skaliert als horizontal und der Text demnach verzerrt.

`\scaleput(`*x*`,`*y*`){`*Bildobjekt*`}` ???
Befehl des `curves`-Paketes. Der Befehl zeichnet *Bildobjekt* an der Position `(`*x*`,`*y*`)`. Gleichzeitig nimmt er eine Parallelprojektion oder Drehung vor, die durch die vier Faktoren `\xscale`, `\xscaley`, `\yscale` und `\yscalex` bestimmt wird. Das optionale Argument *NSymbol* beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

`\scriptscriptstyle` [M]
Umschaltbefehl auf die Schriftgröße `\scriptscriptstyle` innerhalb einer Formel.

`\scriptsize` S. 31
Umschaltbefehl auf die Schriftgröße `\scriptsize`

`\scriptstyle` [M]
Umschaltbefehl auf die Schriftgröße `\scriptstyle` innerhalb einer Formel.

`\scshape` S. 30
Schaltet auf eine KAPITÄLCHEN-SCHRIFT um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\searrow` [M] erzeugt $\searrow$ S. 132

`\sec` [M] S. 107
Erzeugt den Funktionsnamen „sec“ in Formeln.

`\section[`*Kurzform*`]{`*Überschrift*`}` S. 40
Beginnt einen neuen numerierten Abschnitt und trägt ihn in das Inhaltsverzeichnis ein. Der optionale Parameter *Kurzform* kann verwendet werden, um im Inhaltsverzeichnis ein anderen Text zu setzen.

`\section*{`*Überschrift*`}` S. 40
Beginnt einen neuen Abschnitt, ohne ihn zu numerieren. Es erfolgt kein Eintrag ins Inhaltverzeichnis, der interne Zähler bleibt unverändert.

`\see`
Befehl bei Verwendung des `makeidx`-Paketes, der einen Verweis innerhalb des Indexes ermöglicht.
Der Aufruf erfolgt in der Form `\index{`*Eintrag*`|see{`*Verweis*`}`.

`\selectfont` S. 32
Aktiviert die zuvor mit den `\font`*xxx*-Befehlen definierte Schrift.

`\setcounter{`*Zähler*`}{`*Wert*`}` S. 16
Setzt den Zähler mit dem Namen *Zähler* auf *Wert*. Es sind nur ganze Zahlen zugelassen.

`\setlength{\`*Längenbefehl*`}{`*Maßangabe*`}` S. 21
Weist dem Längenbefehl mit Namen `\`*Längenbefehl* den festen oder elastischen Wert *Maßangabe* zu. *Maßangabe* darf dabei auch ein anderer Längenbefehl, ggf. mit einem vorangestellten Vorzeichen und/oder Faktor sein.

`\setminus` [M] erzeugt $\setminus$ S. 130

`\settowidth{\Längenbefehl}{`*Text*`}`
Weist dem Längenbefehl mit Namen `\`*Längenbefehl* die Länge zu, die von *Text* eingenommen wird.

`\sffamily` S. 30
Schaltet auf eine serifenlose Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\shadowbox{`*Inhalt*`}` S. 61
Erzeugt eine schattierte Box um *Inhalt*. Bestandteil des `fancybox`-Paketes.

`\sharp` [M] erzeugt $\sharp$ S. 132

`\shortstack[`*Pos*`]{`*Text*`}` [Z]
Schmale, einspaltige Formatierung, bei der die Zeichen durch `\\` betrennt und so eng wie möglich übereinander gesetzt werden. Optionale kann mit `l` oder `r` für *Pos* eine links- oder rechtsbündige Position erreicht werden.

`\showcols` [P] S. 92
Bewirkt die Ausgabe aller neuen Spaltendeklarationen.

`\Sigma` [M] erzeugt Σ S. 129

`\sigma` [M] erzeugt σ S. 129

`\signature{`*Unterschrift*`}` S. 76
Definiert in einer `letter`-Umgebung die Unterschrift, wenn diese von der Namensangabe aus `\name{Author}` abweichen soll.

`\sim` [M] erzeugt $\sim$ S. 131

`\simeq` [M] erzeugt $\simeq$ S. 131

`\sin` [M] S. 107
Erzeugt den Funktionsnamen „sin“ in Formeln.

`\sinh` S. 107
Erzeugt den Funktionsnamen „sinh“ in Formeln.

`\sloppy`
Erwirkt eine Formatierung mit großzügigeren Wortabständen der nachfolgenden Absätze. Rückschaltbefehl `\fussy`
Siehe auch `\begin{sloppypar}`.

`\slshape` S. 30
Schaltet auf eine *schräggestellte Schrift* um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\small` [Z] S. 31
Schaltet auf die Schriftgröße `\small` um.

`\smallskip` [Z] S. 20
Kleiner vertikaler Abstand vom Betrag `\smallskipamount`.

`\smallskipamount`
Betrag des durch `\smallskip` erzeugten vertikalen Abstandes. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\smile` [M] erzeugt $\smile$ S. 131

`\sout{`*Text*`}` S. 25
Streicht *Text* ~~durch~~. Bestandteil des `ulem`-Paketes.

`\spadesuit` [M] erzeugt $\spadesuit$ S. 132

`\spline(`x_1`,`y_1`)(`x_2`,`y_2`)...(`x_n`,`y_n`)` ???
Befehl des `eepic`-Paketes. Zeichnet eine Chaikins-Kurve, die durch den ersten und letzten Punkt geht. Die übrigen Punkte sind nur Kontrollpunkte.

`\sqcap` [M] erzeugt $\sqcap$ S. 130

`\sqrt[N]{Formelteil}` [M] [Z] S. 104
Befehl zur Erzeugung einer Wurzel. Höhe und Länge des Zeichens wird an den Inhalt von *Formelteil* angepaßt. Der optionale Parameter *N* setzt den Wurzelgrad.

`\sqsubset` [M] erzeugt $\sqsubset$ S. 130

`\sqsubseteq` [M] erzeugt $\sqsubseteq$ S. 130

`\ss` erzeugt ß

`\stackrel{Oben}{Unten}` [M] S. 110
Setzt die beiden Symbole *Oben* und *Unten* übereinander. Das obere Symbol erscheint dabei in kleinerer Schriftgröße.

`\star` [M] erzeugt $\star$ S. 130

`\stepcounter{Zähler}` S. 49
Erhöht *Zähler* um eins.

`\subitem{Untereintrag}`
Befehl innerhalb einer `theindex`-Umgebung zur Erzeugung eines Untereintrages.

`\subparagraph[Kurzform]{Überschrift}` S. 40
Unterster Gliederungsbefehl. Formatiert *Überschrift*, ggf. mit einer 5- oder 6-gliedrigen Nummer. Der optionale Parameter *Kurzform* kann für den Eintrag ins Inhaltsverzeichnis verwendet werden.

`\subparagraph*{Überschrift}` S. 40
Unterster Gliederungsbefehl, ohne Numerierung und ohne Eintrag ins Inhaltsverzeichnis.

`\subsection[Kurzform]{Überschrift}` S. 40
Gliederungsbefehl zwischen `\section` und `\subsubsection`. Formatiert *Überschrift*, ggf. mit einer 2- oder 3-gliedrigen Nummer. Der optionale Parameter *Kurzform* kann für den Eintrag ins Inhaltsverzeichnis verwendet werden.

`\subsection*{Überschrift}` S. 40
Wie `\subsection`, allerdings ohne Numerierung und ohne Eintrag ins Inhaltsverzeichnis.

`\subsubitem{Unter-Unter-Eintrag}`
Erzeugt einen Unteruntereintrag innerhalb einer `theindex`-Umgebung.

`\subsubsection[Kurzform]{Überschrift}` S. 40
Gliederungsbefehl zwischen `\subsection` und `\paragraph`. Formatiert *Überschrift*, ggf. mit einer 3- oder 4-gliedrigen Nummer. Der optionale Parameter *Kurzform* kann für den Eintrag ins Inhaltsverzeichnis verwendet werden.

`\subsubsection*{`*Überschrift*`}` S. 40
: Wie `\subsubsection`, allerdings ohne Numerierung und ohne Eintrag ins Inhaltsverzeichnis.

`\subset` [M] erzeugt $\subset$ S. 131

`\subseteq` [M] erzeugt $\subseteq$ S. 131

`\succ` [M] erzeugt $\succ$ S. 131

`\succeq` [M] erzeugt $\succeq$ S. 131

`\sum` [M] erzeugt $\sum$ S. 132

`\sup` [M] S. 107
: Erzeugt den Funktionsnamen „sup" in Formeln. Der Befehl kann kombiniert werden mit einer Tieferstellung für eine untere Grenzangabe.

`\supset` [M] erzeugt $\supset$ S. 131

`\supseteq` [M] erzeugt $\supseteq$ S. 131

`\surd` [M] erzeugt $\surd$ S. 132

`\swabfamily` S. 34
: Schaltet bei Verwendung des `oldgerm`-Paketes auf Schwabacher-Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\swarrow` [M] erzeugt $\swarrow$ S. 132

`\symbol{`*N*`}`
: Erzeugt im gerade aktiven Zeichensatz das Zeichen mit der internen Kodierung *N* mit $0 \leq N \leq 127$.

T

`\t{`*xy*`}` S. 29
: Erzeugt einen Akzent über zwei Buchstaben: `\t{ac}` = a͡c

`\tabcolsep` S. 86
: Definiert den halben Spaltenabstand in einer `tabular`-Umgebung. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\tableofcontents` S. 42
: Erzeugt das Inhaltsverzeichnis mit den Angaben aus den Gliederungsbefehlen und event. Zusatzeinträgen.

`\tagcurve[`*NSymbol*`](`x_1`,`y_1`,`x_2`,`y_2`,...,`x_n`,`y_n`)` ???
: Befehl des `curves`-Paket. Der Befehl zeichnet eine Kurve ohne das erste und letzte Segment. Werden nur drei Punkte angegeben, so wird das letzte Segment gezeichnet. Das optionale Argument *NSymbol* beeinflußt die Anzahl der zu zeichnenden Kurvensegmente.

`\tan` [M] S. 107
Erzeugt den Funktionsnamen „tan" in Formeln.

`\tanh` S. 107
Erzeugt den Funktionsnamen „tanh" in Formeln.

`\tau` [M] erzeugt τ S. 129

`\TeX` erzeugt TEX

`\textbf{`*`Text`*`}` S. 30
Schreibt *`Text`* in **Fettdruck**.

`\textcolor{`*`Farbe`*`}{`*`Text`*`}` ???
Befehl des `color`-Paketes. Druckt *`Text`* in *`Farbe`*.

`\textcolor[`*`Modell`*`]{`*`Definition`*`}{`*`Text`*`}` ???
Befehl des `color`-Paketes. Analog zum obigen Befehl, allerdings wird die Farbe direkt definiert.

`\textfloatsep`
Der vertikale Abstand zwischen Gleitobjekten oben auf der Seite und dem nachfolgenden Text, bzw. zwischen dem Text und unten folgenden Gleitobjekten. Elastische Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\textfraction`
Der Anteil einer Seite mit Text und Gleidobjekten, der für den Text mindestens zur Verfügung steht. Wertzuweisung erfolgt mit `\renewcommand`.

`\textfrak{`*`Text`*`}` S. 34
Setzt *`Text`* bei Verwendung des `oldgerm`-Paketes in Fraktur-Schrift.

`\textgoth{`*`Text`*`}` S. 34
Setzt *`Text`* bei Verwendung des `oldgerm`-Paketes in gotischer Schrift.

`\textheight` [P] S. 14
Gesamthöhe für den Seitentext. Eine Wertzuweisung erfolgt wie gewohnt mit `\setlength` oder `\addtolength`.

`\textit{`*`Text`*`}` S. 30
Schreibt *`Text`* in *kursiver Schrift*.

`\textmd{`*`Text`*`}` S. 30
Schreibt *`Text`* in der Schriftstärke medium.

`\textnormal{`*`Text`*`}` S. 30
Schreibt *`Text`* in der aktuell eingestellten Basis-Schrift.

`\textrm{`*`Text`*`}` S. 30
Schreibt *`Text`* in der aktuell eingestellten Serifenschrift. Üblicherweise ist dies die Standardschrift.

`\textsc{Text}` S. 30
Schreibt *Text* in Kapitälchen.

`\textsf{Text}` S. 30
Schreibt *Text* in serifenloser Schrift.

`\textsl{Text}` S. 30
Schreibt *Text* in *schräggestellter Schrift*.

`\textswab{Text}` S. 34
Setzt *Text* bei Verwendung des `oldgerm`-Paketes in Schwabacher-Schrift.

`\textstyle` [M]
Schaltet auf die Schriftgröße `\textstyle` innerhalb von Formeln um.

`\texttt{Text}` S. 30
Schreibt *Text* in `Typewriter-Schrift`.

`\textup{Text}` S. 30
Schreibt *Text* in senkrechter Schrift.

`\textwidth` [M] S. 14
Die Breite des Textes. Die Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\thanks{Fußnotentext}` S. 42
Ermöglicht eine Fußnote auf einer Titelseite.

`\theZähler`
Interner Befehl, mit dem der aktuelle Wert des Zählers ausgegeben wird.

`\TheSbox` S. 61
Befehl aus dem `fancybox`-Paket, das eine mit der `Sbox` definierten Umgebung aufruft.

`\Theta` [M] erzeugt Θ S. 129

`\theta` [M] erzeugt θ S. 129

`\Thicklines` ???
Befehl des `eepic`-Paketes. Der Befehl stellt die Strichstärke auf das 1.5-fache des Wertes von `\thicklines`.

`\thicklines` ???
Nach diesem Befehl erscheinen alle Linien innerhalb einer `picture`-Umgebung in dicker Strichstärke.

`\thinlines` ???
Nach diesem Befehl erscheinen alle Linien innerhalb einer `picture`-Umgebung in dünner Strichstärke.

`\thispagestyle{Stil}` S. 15
Ändert den Seitenstil der laufenden Seite auf *Stil*.

`\tilde{`*x*`}` [M] S. 130
Erzeugt eine Tilde über der Variable *x*.

`\tiny` [Z] S. 31
Schaltet auf die kleinste Schriftgröße um.

`\times` [M] erzeugt $\times$ S. 130

`\title{`*Titel*`}` S. 41
Legt den Titel auf einer durch `\maketitle` erzeugten Titelseite fest.

`\to` [M] erzeugt $\to$ S. 132

`\today` S. 7
Erzeugt das aktuelle Datum.

`\top` [M] erzeugt $\top$ S. 132

`\topfraction` [P] S. 14
Anteil einer Seite, bis zu dem Gleitobjekte am oberen Seitenende gesetzt werden dürfen. Wertänderung erfolgt mit `\renewcommand`.

`\topnumber`
Maximale Anzahl von Gleitobjekten, die am oberen Seitenende gesetzt werden dürfen. Wertänderung erfolgt mit `\setcounter`.

`\topsep`
Vertikaler Zwischenraum zusätzlich zu `\parsep` vor und nach Listenumgebungen und abgesetzten Formeln. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\topskip` [P] S. 14
Abstand zwischen der Oberkante des Seitenrumpfes und der Grundlinie der ersten Textzeile. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\totalnumber`
Anzahl der Gleitobjekte, die unabhängig von ihrer Größe maximal auf einer Seite gesetzt werden dürfen.

`\triangle` [M] erzeugt $\triangle$ S. 132

`\triangleleft` [M] erzeugt $\triangleleft$ S. 130

`\triangleright` [M] erzeugt $\triangleright$ S. 130

`\ttfamily` S. 30
Schaltet auf eine `Typewriter-Schrift` um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\twocolumn[`*Text*`]` [Z] S. 64
Beginnt eine neue Seite mit zweispaltiger Formatierung. Der optionale *Text* wird zu Beginn der Seite gesetzt und reicht über beide Spalten.

U

`\u{`*Zeichen*`}` S. 29
: Erzeugt einen „Breve-Akzent“ über *Zeichen*: `\u{x}` = x̆

`\ULforem` S. 25
: Schaltet auf die Unterstreichungsfunktionen des `ulem`-Paketes um.

`\uline{`*Text*`}` S. 25
: Unterstreicht *Text*. Bestandteil des `ulem`-Paketes.

`\unboldmath` [Z]
: Rückschaltbefehl zu `\boldmath`.

`\underbrace{`*Teilformel*`}` [M] S. 105
: Erzeugt eine horizontale geschweifte Klammer unter der angegebenen *Teilformel*.

`\underline{`*Text*`}` [Z] S. 105
: Erzeugt eine horizontale Linie unter *Text*.

`\unitlength`
: Definiert die Einheitslänge in einer `picture`-Umgebung. Wertzuweisung erfolgt mit `\setlength`. Durch Änderung des Wertes kann ein Bild sehr leicht vergrößert oder verkleinert werden.

`\unlhd` [M] erzeugt $\unlhd$ S. 131

`\unrhd` [M] erzeugt $\unrhd$ S. 131

`\Uparrow` [M] erzeugt $\Uparrow$ S. 132

`\uparrow` [M] erzeugt $\uparrow$ S. 132

`\Updownarrow` [M] erzeugt $\Updownarrow$ S. 132

`\updownarrow` [M] erzeugt $\updownarrow$ S. 132

`\uplus` [M] erzeugt $\uplus$ S. 130

`\upshape` S. 30
: Schaltet auf eine senkrechte Schrift um. Der Befehl wirkt bis zum Ende der aktuellen Umgebung.

`\Upsilon` [M] erzeugt Υ S. 129

`\upsilon` [M] erzeugt υ S. 129

`\usebox{\`*Boxname*`}` S. 59
: Gibt den Inhalt von `\`*Boxname* aus. Der Name der Box muß zuvor mit `\newsavebox{\`*Boxname*`}` eingerichtet worden sein. Gefüllt wird die Box mit einem `\sbox`- oder `\savebox`-Befehl.

`\usepackage[Optionen]{Pakete}` S. 11
: Lädt die durch Komma getrennten Paketdateien. Die *Optionen* werden an alle mit einem Aufruf geladenen Paketen weitergeleiten und müssen von diesen unterstützt werden. Alternativ können die Optionen auch im `\documentclass`-Befehl übergeben werden.

`\uwave{Text}` S. 25
: Unterstreicht *Text* mit einer Schlangenlinie.

V

`\v{Zeichen}` S. 29
: Erzeugt einen Háček-Akzent über *Zeichen*: `\v{c}` = č

`\value{Zähler}`
: Liefert den aktuellen Wert von *Zähler*. Er kann nun einem anderen Zähler übergeben werden: `\setcounter{Zähler1}{\valueZähler2}`

`\varepsilon` [M] erzeugt ε S. 129

`\varpi` [M] erzeugt ϖ S. 129

`\varrho` [M] erzeugt ϱ S. 129

`\varsigma` [M] erzeugt ς S. 129

`\vartheta` [M] erzeugt ϑ S. 129

`\vdash` [M] erzeugt $\vdash$ S. 131

`\vdots` [M] erzeugt $\vdots$ S. 106

`\vec{x}` [M] S. 108
: Erzeugt das Vektor-Symbol über *x*: `\vec{s}` = $\vec{s}$

`\vector(Δx,Δy){Länge}`
: Bildbefehl innerhalb einer `picture`-Umgebung zur Erzeugung von Pfeillinien. Es stehen dabei standardmäßig nur eine begrenzte Anzahl von Steigungen zur Verfügung. Bei horizontalen und vertikalen Linien stellt *Länge* die tatsächliche Länge dar, bei geneigten Linien die Projektion der Linienlänge auf die x-Achse. Die Neigung wird durch die beiden Werte Δx und Δy bestimmt, wobei nur ganzahlige Werte zwischen -4 und +4 erlaubt sind.

`\vee` [M] erzeugt $\vee$ S. 130

`\verb|original Text|` S. 57
: Der angegebene Text wird in Schreibmaschinenschrift genau so gesetzt, wie er eingegeben wird. Für `|` darf jedes Zeichen außer `*` und Leerzeichen verwendet werden, das in *original Text* nicht auftritt.
Siehe auch `\begin{verbatim}`

`\verb*|original Text|` S. 57
: Bewirkt das gleiche wie `\verb| |`. Allerdings werden Leerzeichen als ␣ ausgegeben.
Siehe auch `\begin{verbatim*}`

`\vfill` S. 20
: Erzeugt einen vertikalen Zwischenraum beliebiger Länge. Der Befehl ist eine Abkürzung für `\vspace{\fill}`.

`\vline` S. 85
: Erzeugt einen vertikalen Zwischenstrich innerhalb eines Spalteneintrages in einer Tabelle.

`\voffset` [P]
: Definiert den oberen Bezugsrand der Seitenformatierung. Wertzuweisung erfolgt mit `\setlength` oder `\addtolength`.

`\vpageref[Diese Seite][Andere Seite]{Marke}` [Z] S. 47
: Bestandteil des `varioref`-Paketes. Erzeugt die Seitennummer derjenigen Seite, auf der mit `\label{Marke}` eine unsichtbare Markierung angebracht wurde. Der optionale Parameter *Diese Seite* wird verwendet, wenn die Marke auf der gleichen Seite liegt, *Andere Seite* wird verwendet, wenn die Marke auf einer anderen Seite liegt.

`\vref{Marke}` S. 46
: Bestandteil des `varioref`-Paketes. Erzeugt eine Gliederungs-, Gleichungs-, Bild- oder Tabellennummer, je nachdem, wo mit `\label{Marke}` eine Markierung angebracht wurde. Der Text hängt davon ab, wo sich die Marke genau befindet. Liegt die Marke z.B. auf der nächsten Seite, so ergibt der Befehl nicht die Seitennummer, sondern den Text „auf der nächsten Seite“.

`\vspace{Abstand}` S. 21
: Erzeugt einen vertikalen Zwischenraum der Länge *Abstand*. Am Anfang oder Ende einer Seite wird der Zwischenraum nicht gesetzt.

`\vspace*{Abstand}` S. 21
: Wie `\vspace`, allerdings wird der Abstand auch am Anfang oder Ende einer Seite gesetzt.

W

`\wedge` [M] erzeugt $\wedge$ S. 130

`\widehat{Arg}` [M] S. 108
: Erzeugt ein breites `\hat` über *Arg*: `\widehat{abc}` = $\widehat{abc}$

`\widetilde` [M] S. 108
: Erzeugt ein breites Tildesymbol über *Arg*: `\widetilde{abc}` = $\widetilde{abc}$

X

Z

Anhang D

Layout dieses Handbuches

An dieser Stelle soll noch ein wenig über das in diesem Buch verwendete Layout gesprochen werden. Dazu werden Ausschnitte aus der Hauptdatei des Buches aufgeführt und erläutert.

Das Buch verwendet die Dokumentenklasse `book`. Außer ein paar Anpassungen an den deutschen Sprachraum wurde die Klassendatei nicht weiter verändert.

Leicht erkennbar ist, daß die Kopf- und Fußzeile ein deutliches Stück über den Textrand nach außen ragen. Auf diese Weise ist mehr Platz für Randnotizen entstanden und die vielen Beispiele, die ja ebenfalls die Platz der Randnotizen einnehmen, sind leichter erkennbar. Verwendet wurde hierzu das Paket `fancyheadings` (siehe Abschnitt 3.1.3 auf Seite 16).

```
\setlength{\textwidth}{451pt}
\usepackage{fancyheadings}
\pagestyle{fancy}
\lhead[\fancyplain{}{\slshape \thepage}]
    {\fancyplain{}{\slshape \rightmark}}
\rhead[\fancyplain{}{\slshape \leftmark}]
    {\fancyplain{}{\slshape \thepage}}
\cfoot{}
\setlength{\footrulewidth}{0.5pt}
\setlength{\headrulewidth}{0.5pt}
```

Wie man hier sieht, wird die Textbreite zunächst auf 451pt gesetzt. Diese Breite übernimmt das `fancybox`-Paket in dem Moment, in dem der Seitenstil auf `fancy` festgelegt wird. Spätere Änderungen an der Textbreite haben keinerlei Einfluß auf die gesetzten Kopf- und Fußzeilen.

Anschließend wird das eigentliche Seitenlayout festgelegt. Hier muß vor allem dafür gesorgt werden, daß der Seitenrand für die geraden Seiten entsprechend vergrößert wird, um die kürzeren Textzeilen auszugleichen. Das Ziel ist es, daß die Linien der Kopf- und Fußzeilen auf den Seiten exakt übereinander liegen. Ebenfalls festgelegt wird hier die Breite für die `\marginpar`-Befehle.

```
\setlength{\oddsidemargin}{71pt}
\setlength{\evensidemargin}{125pt}
\setlength{\topmargin}{80pt}
\setlength{\textwidth}{369pt}
\setlength{\textheight}{597pt}
\setlength{\marginparwidth}{75pt}
\addtolength{\headheight}{2pt}
```

Nun kann damit begonnen werden, die Befehle für die Beispiele zu erzeugen. Da sehr viele Beispiele verwendet werden, lohnt es sich auf jeden Fall, eigene, leicht zu bedienende Befehle zu erzeugen.

Zunächst muß ein Weg gefunden werden, die Beispiele in Boxen zu setzen, die breiter als der normale Text sind und immer *nach außen* ragen. Hierzu wird die Umgebung `WideBox` erzeugt. Der bei der Definition verwendete Befehl `Sbox` stammt aus dem `fancybox`-Paket. Mit ihm ist es möglich, eine Box mit `\begin` und `\end` zu definieren.

```
\newenvironment{WideBox}
    {\nopagebreak \medskip \begin{Sbox}
      \begin{minipage}{451pt}}
    {\end{minipage} \medskip \end{Sbox} \medskip
      \noindent
      \ifodd\thepage
        \makebox[\textwidth][l]
        {\makebox[451pt]{\TheSbox}}
      \else
        \makebox[\textwidth][r]
       {\makebox[451pt]{\TheSbox}}
     \fi}

\newenvironment{WideBoxR}
    {\nopagebreak \begin{Sbox}
      \begin{minipage}{451pt}}
    {\end{minipage} \medskip \end{Sbox}
       \noindent
       \makebox[\textwidth][r]
       {\makebox[451pt]{\TheSbox}}}

\newenvironment{WideBoxL}
    {\nopagebreak \begin{Sbox}
      \begin{minipage}{451pt}}
    {\end{minipage} \medskip \end{Sbox}
       \noindent
       \makebox[\textwidth][l]
      {\makebox[451pt]{\TheSbox}}}
```

Die Abfrage, ob die Box nach links oder rechts hinausragen muß, erfolgt mit dem Befehl `\ifodd`. Ist die Seite eine gerade Seite, so wird eine Box der Breite `\textwidth` erzeugt, deren Inhalt rechtsbündig gesetzt wird. Da der Inhalt aus einer breiteren Box besteht, ragt er nach links heraus. Bei den ungeraden Seiten ist es genau umgekehrt.

Die beiden zusätzlichen Umgebungen `WideBoxR` und `WideBoxL` werden dort verwendet, wo die automatische Abfrage nicht funktioniert. Ist ist immer dann der Fall, wenn die Box auf die nächste Seite rutscht. In diesem Fall ist also ein manueller Eingriff nötig.

Nun wurden also die Boxen für die Beispiele erzeugt. Die Beispiele selbst enthalten jedoch immer zwei Teile: Auf der linken Seite steht das Ergebnis und auf der rechten die LaTeX-Befehle, mit denen das Beispiel erzeugt worden ist. Zu diesem Zweck muß eine `minipage`-Umgebung innerhalb der `WideBox` erzeugt werden. Zudem soll vor und nach dem Beispiel jeweils eine horizontale Linie als Trennung erscheinen. Die `WideBox`-Umgebung muß also erweitert werden.

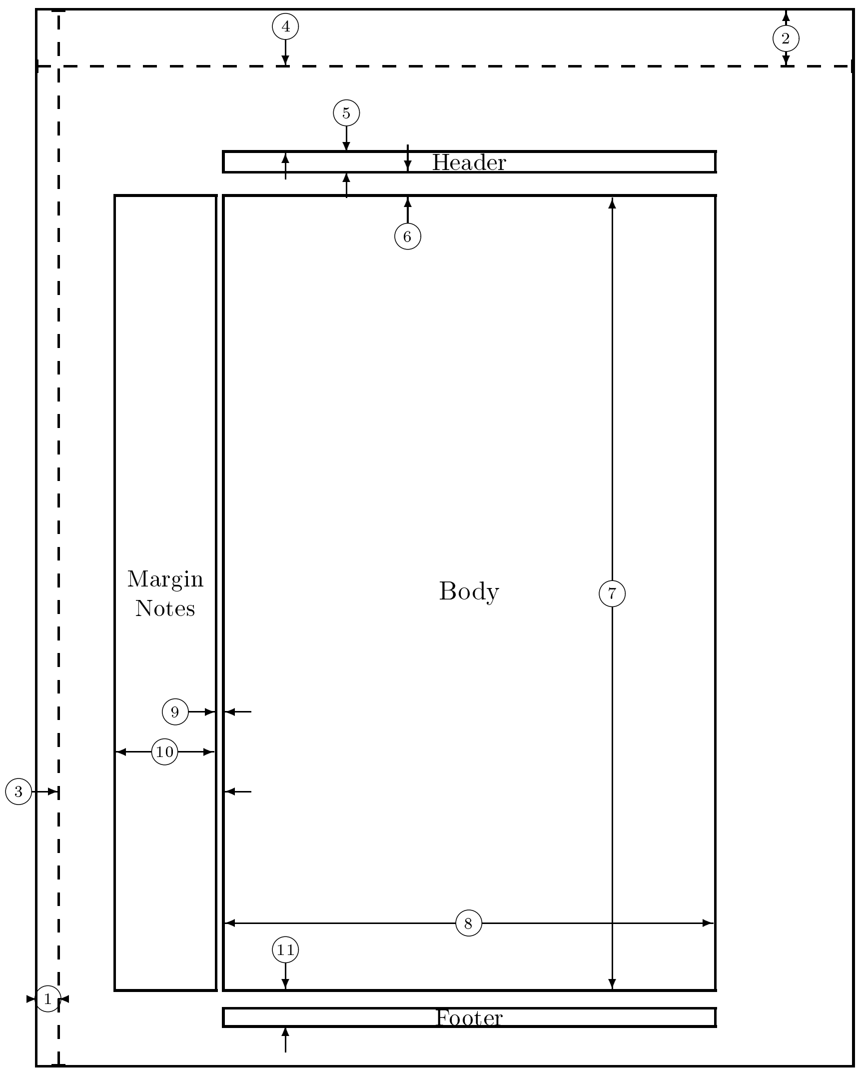
```
\newenvironment{RuleExample}
    {\begin{WideBox}\medskip\hrule\medskip
      \begin{flushleft} \smallskip \nopagebreak
      \begin{minipage}[t]{75mm}\sloppy}
    {\end{minipage}\end{flushleft}
      \hrule \medskip \end{WideBox}}
```

Auch diese Umgebung gibt es in zwei weiteren Versionen, die auf der `WideBoxL` bzw. der `WideBoxR` basieren. Außerdem existieren auch Versionen ohne eine horizontale Linie.

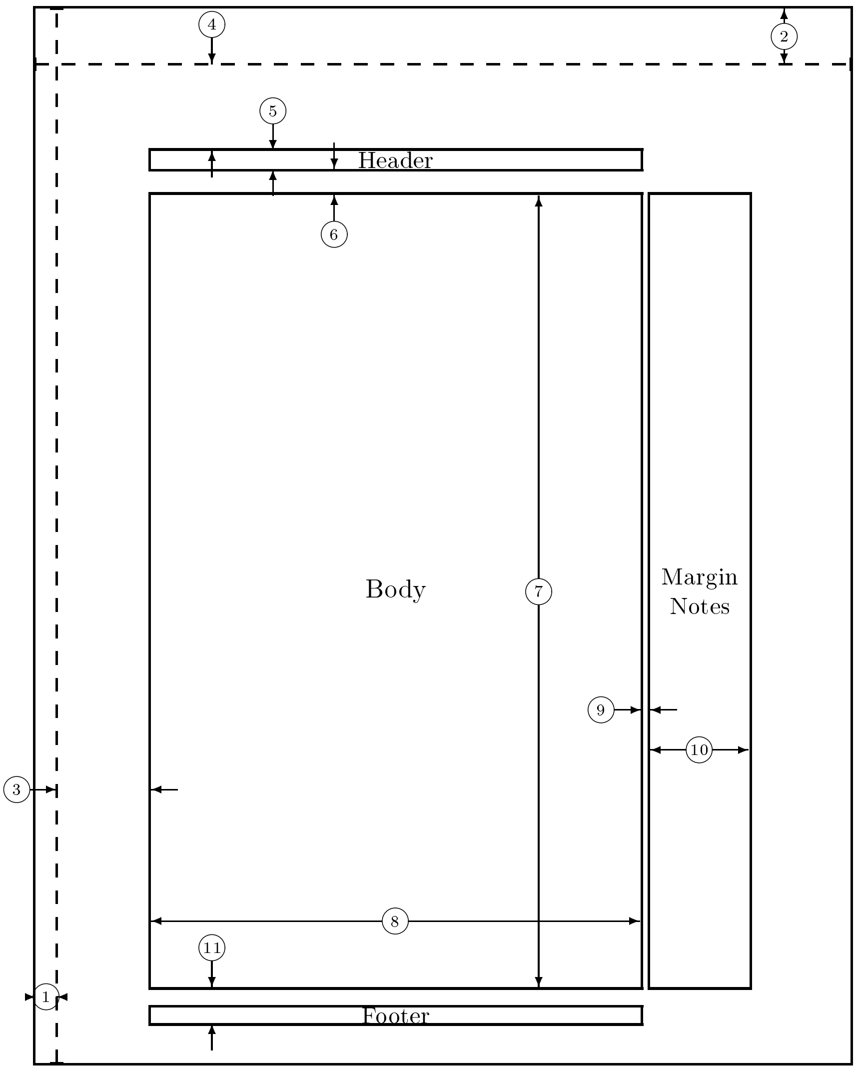
Wer sich diese Definition genau ansieht und sie mit den Beispielen vergleicht, wird feststellen, daß noch ein wenig fehlt. Die Beispiele bestehen ja aus zwei nebeneinander liegenden Teilen. In der Definition taucht jedoch nur eine `minipage`-Umgebung auf. Gelöst wird das Problem über einen weiteren Befehl, der die erste `minipage` beendet, und eine zweite beginnt.

```
\newcommand{\Mitte}
    {\end{minipage}\hspace*{\fill}
      \begin{minipage}[t]{75mm}\sloppy}
```

Auf der folgenden Doppelseiten ist das Layout dieses Handbuches dargestellt. Die Ausgabe wurde mit dem Befehl `\layout` aus dem gleichnamigen Paket erstellt. Im Vergleich kann man sehr gut die Unterschiede erkennen, die LaTeX zwischen einer rechten und einer linken Textseite vornimmt.



1	one inch + \hoffset	2	one inch + \voffset
3	\evensidemargin = 125pt	4	\topmargin = 65pt
5	\headheight = 14pt	6	\headsep = 19pt
7	\textheight = 597pt	8	\textwidth = 369pt
9	\marginparsep = 7pt	10	\marginparwidth = 75pt
11	\footskip = 27pt		\marginparpush = 5pt (not shown)
	\hoffset = -56pt		\voffset = -30pt
	\paperwidth = 614pt		\paperheight = 794pt



1	`one inch + \hoffset`	2	`one inch + \voffset`
3	`\oddsidemargin = 71pt`	4	`\topmargin = 65pt`
5	`\headheight = 14pt`	6	`\headsep = 19pt`
7	`\textheight = 597pt`	8	`\textwidth = 369pt`
9	`\marginparsep = 7pt`	10	`\marginparwidth = 75pt`
11	`\footskip = 27pt`		`\marginparpush = 5pt (not shown)`
	`\hoffset = -56pt`		`\voffset = -30pt`
	`\paperwidth = 614pt`		`\paperheight = 794pt`

Literaturverzeichnis

LATEX3 PROJECT TEAM (1995*a*): LATEX 2_ε font selection.

LATEX3 PROJECT TEAM (1995*b*): LATEX 2_ε for authors.

AMERICAN MATHEMATICAL SOCIETY (1997): User's Guide to AMSTEX.

BARROCA, L. (1995): A style option for rotated object in LATEX. Dokumentation des Paketes.

CARLISLE, D. (1995): The `afterpage` package. Dokumentation des Paketes.

CARLISLE, D. (1998*a*): The colortbl package. Dokumentation des Paketes.

CARLISLE, D. (1998*b*): Packages in the 'graphics' bundle. Dokumentation des Paketes.

CLARK, M. (192): Portable graphics in TEX. *TUGboat*, 13. Jg., Nr. 3, S. 253 – 260.

GOOSSENS, M., MITTELBACH, F., & SAMARIN, A. (1994): *Der LATEX Begleiter* Addison Wesley, Bonn, Paris.

GOOSSENS, M., RAHTZ, S., & MITTELBACH, F. (1997): *The LATEX Graphics Companion.* Addison Wesley.

KNESER, T. (1990): *LATEX-Paragraphs Floating around Figures.* Gesellschaft für wissenschaftliche Datenverarbeitung Göttingen, Göttingen.

KNUTH, D. E. (1986): *The TEXbook* In: Computers and Typesetting., Band Band A. Addison-Wesley, Reading.

KOPKA, H. (1994): *LATEX Einführung, Band 1.* Addison Wesley, Bonn, Paris.

KOPKA, H. (1995): *LATEX Ergänzungen – mit einer Einführung in METAFONT, Band 2.* Addison Wesley, Bonn, Paris.

LAMPORT, L. (1994): *LaTeX – A Docoment Prepartion System – User's Guide and Reference Manual.* 2. Aufl, Addison-Wesley, Reading.

LINGNAU, A. (1995): An Improved Environment for Floats. Dokumentation des Paketes.

MACLAINE-CROSS, I. (1995): *Curves in LaTeX Pictures. A Manual for* **CURVES.STY** *and* **CURVESLS.STY***.* School of Mechanical and Manufacturing Engineering. The University of New South Wales., Sydney.

MITTELBACH, F. (1998*a*): An environment for multicolumn output. Dokumentation des Paketes.

MITTELBACH, F. (1998*b*): *Footnotes in a multi-column layout.* Electronic Data Systems GmbH, Rüsselsheim.

MITTELBACH, F. (1998*c*): The `varioref` package. Dokumentation des Paketes.

MITTELBACH, F. & CARLISLE, D. (1998): A new implementation of LaTeX's `tabular` and `array` environment Dokumentation des Paketes.

OOSTRUM, P. V. (1995): *Page headers and footers in LaTeX.* Dept. of Computer Science. Utrecht University, Utrecht.

PODAR, S. (1986): Enhancements to the Picture Environment of LaTeX. (Version 1.2). Technical Report 86-17, Dept. of Computer Science S. U. N. Y. at Stony Brook.

RATHZ, S. & BARROCA, L. (1994): Test of 'rotating' package Dokumentation des Paketes.

SCHWARZ, N. (1991): *Einführung in TeX.* 3. überarbeitete Aufl, Addison Wesley, Bonn, München.

VANROOSE, P. (1990): trees.sty: A Macro for Drawing Binary of Ternary Trees. Dokumentation des Paketes.

WHITE, J. V. (1990): *Color for the Electronic Age.* Watson-Guptil Publications, New York.

Index

Symbole

A

C

M

Q

R

S

U

V